Biyani's Think Tank

Concept based notes

Object Oriented Programming (Java and Visual Basic)

(BCA Part-II)

Shipra Rastogi
Revised By: Kritika Saxena
Deptt. of Information Technology
Biyani Girls College, Jaipur

Published by :
Think Tanks
Biyani Group of Colleges

Sector-3, Vidhyadhar Nagar,
Jaipur-302 023 (Rajasthan)
Ph : 0141-2338371, 2338591-95 • Fax : 0141-2338007
E-mail : acad@biyanicolleges.org
Website :www.gurukpo.com; www.biyanicolleges.org

ISBN : 978-93-81254-44-8

Edition : 2011
Price :

Leaser Type Setted by :
Biyani College Printing Department

Preface

I am glad to present this book, especially designed to serve the needs of the students. The book has been written keeping in mind the general weakness in understanding the fundamental concepts of the topics. The book is self-explanatory and adopts the "Teach Yourself" style. It is based on question-answer pattern. The language of book is quite easy and understandable based on scientific approach.

This book covers basic concepts related to the microbial understandings about diversity, structure, economic aspects, bacterial and viral reproduction etc.

Any further improvement in the contents of the book by making corrections, omission and inclusion is keen to be achieved based on suggestions from the readers for which the author shall be obliged.

I acknowledge special thanks to Mr. Rajeev Biyani, *Chairman* & Dr. Sanjay Biyani, *Director* (*Acad.*) Biyani Group of Colleges, who are the backbones and main concept provider and also have been constant source of motivation throughout this Endeavour. They played an active role in coordinating the various stages of this Endeavour and spearheaded the publishing work.

I look forward to receiving valuable suggestions from professors of various educational institutions, other faculty members and students for improvement of the quality of the book. The reader may feel free to send in their comments and suggestions to the under mentioned address.

Author

Syllabus

B.C.A. Part-II

Object Oriented Programming through Java

Constants, Variables, Data Types, Arithmetic Operations, Relational Operators, Logical Operators, Assignment Operators, Increment and Decrement Operator, Conditional Operator, Bit-wise Operator, Arithmetic Expression, Type Conversion in Expressions, Mathematical Functions, Decision Control Structure, Loop Control Structure, Classes, Objects and Methods, Boolean Methods, Void Methods, Overloading, Nesting of Methods, Constructors, Class Invariants, Composition, Recursive Classes, Extending a Class, Overriding Method, Inheritance versus Compositions, Class Hierarchies, Arrays and Vector, String Arrays, Wrapper (Classes), Defining, Extending and Implementing Interfaces, Accessing Interface Variables, Graphics, Managing Layouts, Event Driven Programming, Applets, Thread and Exceptions, Managing Input Output Files, Reusable Classes, Searching, Sorting and Recursive Algorithims.

Object Oriented Programming through Visual Basic

Object Model, Visual Basic Environment, Visual Basic Code Statements, Controls, Coding for the Controls, Variables, Constants and Calculations, Decision Control Structure, Loop Control Structure, Nested Ifs Statement, Input Validations, Calling Event Procedures, Menus, Sub Procedures and Sub Functions, Multiple Forms, Variables and Constants in Multiple Form Projects, List Boxes and Combo Boxes, Using Mfg Box and String Function, Arrays, Using List Boxes and Arrays, Multidirectional Arrays, Classes, Initializing and Terminating Events, Collections, Using the Object Browser, Data Files, Sequential File Organisation, Random Data Files, Accessing Database Files, Navigation the Database in Code, Displaying Data in Grids, Validation and Error Trapping, Dragging and Dropping Multiple Objects, Graphics, Layering, Simple Animation, Active X, Dynamic Link Libraries, Object Linking and Optimizing VB Code, OLE Automation and VBA, Automating Word, Excel and Outlook 98.

□ □ □

Content

Part-I (Java)

Part-II (Visual Basic)

S.No.	Name of Topic
1.	**Integrated Development Environment**
	1.1 Visual Basic Environment
2.	**Basic Programming Concepts**
	2.1 Controls
	2.2 Properties
	2.3 Variables Constants
3.	**Array Declaration**
	3.1 Arrays
	3.2 Multidimensional Arrays
4.	**Procedural Programming**
	4.1 Objects and Modules
	4.2 Procedures (Sub Procedures and Sub Functions)
	4.3 Functions
5.	**Branching and Looping Constructs**
	5.1 Loop Control Structures
	5.2 Decision Control Procedures
6.	**Forms**
	6.1 Multiple Forms

□ □ □

Chapter-1

Basics of Object Oriented Programming

Q.1 What is the difference between Procedural and Object Oriented Programming?

Ans.: The different languages reflect the different styles of programming. Procedural programming decomposes a program into various different functional units, each of which can gather and manipulate data as needed. Object-oriented programming, on the other hand, decomposes a program into various different data-oriented units or other conceptual units; each unit contains data and various operations that may be performed on that data.

Procedural programming forced developers to write highly interdependent code. A minor change in any part of the code may end up with a bigger modification of large pieces of code. This way of programming is not up to the mark when the user interactivity is higher.Using OOP, a developer can break down any real world item into a collection of objects. These individual objects can be used separately and collectively to accomplish the requirements.

We can summarize the differences as follows :

- **Procedural Programming**
  - top down design
  - create functions to do small tasks
  - communicate by parameters and return values
- **Object Oriented Programming**

- design and represent objects
- determine relationships between objects
- determine attributes each object has
- determine behaviours each object will respond to
- create objects and send messages to them to use or manipulate their attributes

Q.2 What are the basic features of Object Oriented Programming?

Ans.: **Object Oriented Programming (OOP)** is a programming paradigm that uses "objects" and their interactions to design applications and computer programs. It is based on several techniques, including encapsulation, modularity, polymorphism, and inheritance.

- **Inheritance :** 'Subclasses' are more specialized versions of a class, which *inherit* attributes and behaviors from their parent classes, and can introduce their own. Inheritance is inheriting a class into another class.
- **Encapsulation :** Encapsulation conceals the functional details of a class from objects that send messages to it. This is hiding the internal details.
- **Abstraction :** Abstraction is simplifying complex reality by modelling classes appropriate to the problem, and working at the most appropriate level of inheritance for a given aspect of the problem. Building class is an abstraction process.
- **Polymorphism :** Multiple objects exhibiting similar features in different ways is known as polymorphism. Polymorphism is the process of using an operator or function in different ways for different set of inputs given.

Q.3 What are the basic differences between C++ and Java?

Ans.: C++ and Java both are Object Oriented Languages but some of the features of both languages are different from each other. Some of these features are :

- All stand-alone C++ programs require a function named **main** and can have numerous other functions, including both stand-alone functions and functions, which are members of a class. There are no stand-alone functions in Java. Instead, there are only functions that are members of a class, usually called **methods**. Global functions and global data are not allowed in Java.
- All classes in Java ultimately inherit from the **Object** class. This is significantly different from C++ where it is possible to create inheritance trees that are completely unrelated to one another.
- Java does not support **multiple inheritance**. To some extent, the *interface* feature provides the desirable features of multiple inheritance to a Java program without some of the underlying problems.
- Java does not support **operator overloading**.

□ □ □

Chapter-2

Applets

Q.1 What are the differences between Applications and Applets?

OR

Explain Java Applets?

Ans.: Java is a general-purpose, object-oriented programming language. We can develop two types of Java programs :

Stand alone Applications : Programs written in Java to carry out certain tasks on a stand alone local computer. Executing stand alone java programs involves two steps :

(a) Compiling through **javac** compiler.

(b) Executing the byte code using **java** interpreter.

Web Applets : Programs that are developed for Internet based applications. An applet located on a distant computer (Server) can be downloaded via Internet and executed on a local computer (Client) using a java capable browser.

From the user's point of view, Java applications can be compared to compiled C programs since they are started from a command line just like any compiled program would be started. However, there is a major

difference: Java applications, as well as applets, are interpreted. Applications are started on the command-line by calling the Java Interpreter with the name of the application as an argument. Applets, in contrary to applications, are small programs which can be included in web pages and run inside the user's browser. Alternatively, applets can be run in the applet viewer that comes with the JDK, which has advantages for debugging.

□ □ □

Chapter-3

Java Interpreter and Runtime Environment

Q.1 What is JVM?

Ans.: A **Java Virtual Machine** (**JVM**) is a set of computer software programs and data structures which use a virtual machine model for the execution of other computer programs and scripts. The model used by a JVM accepts a form of computer intermediate language commonly referred to as Java bytecode. This language conceptually represents the instruction set of a stack-oriented, capability architecture.

Java Virtual Machines operate on Java bytecode, which is normally (but not necessarily) generated from Java source code; a JVM can also be used to implement programming languages other than Java. For example, Ada source code can be compiled to Java bytecode, which may then be executed by a JVM. JVMs can also be released by other companies besides Sun (the developer of Java) -- JVMs using the "Java" trademark may be developed by other companies as long as they adhere to the JVM specification published by Sun (and related contractual obligations).

The JVM is a crucial component of the Java Platform. Because JVMs are available for many hardware and software platforms, Java can be both middleware and a platform in its own right — hence the expression "write once, run anywhere." The use of the same bytecode for all platforms allows Java to be described as "compile once, run anywhere", as opposed to "write once, compile anywhere", which describes cross-platform compiled languages. The JVM also enables such unique features as

Automated Exception Handling which provides 'root-cause' debugging information for every software error (exception) independent of the source code.

Q.2 What is Automatic Memory Management?

Ans.: Automatic memory management, also known as automatic garbage collection, is the practice of allowing the language implementation to keep track of used and unused memory, freeing the programmer from this burden.

In languages without automatic memory management, the programmer must reserve memory when it is required and mark it as free once its contents no longer have effect in the running program. For example, in the C programming language (probably the most commonly used of those without automatic memory management) this is achieved by using the functions *malloc*, *calloc*, *realloc*, *alloc* and *free*.

On the other hand, languages (or implementations) with automatic memory management automatically reserve memory when it is required for storing new values and reclaim it (free it) when their contents can no longer affect future computations

(1) Java automatically does the Garbage collecting and probably does a better job than most developers on the down side.

(2) By auto Garbage collecting it takes away an opportunity for developers to optimise garbage collecting and auto Garbage collecting can be seen as a constraint not a benefit.

In the most part it is a benefit because it simplifies programming in Java and is probably more efficient at garbage collecting than self regulated Garbage collecting, unless people spent time researching the best way to garbage collect.

Garbage collection concerns objects with dynamic extent (allocated with a new statement in these languages). Java uses various techniques to automatically discover when these objects are no longer referred to, and to reclaim them. In contrast, C++ programmers manually specify where an object with dynamic extent is to be reclaimed by coding a delete statement.

- Java classes can have a finalize function.
- Java uses run-time processing to discover dead objects (which are eligible for reclamation).

Q.3 What is JRE?

OR

What is Java Runtime Environment?

Ans.: The Java Runtime Environment (JRE) provides the libraries, the Java Virtual Machine, and other components to run applets and applications written in the Java programming language. In addition, two key deployment technologies are part of the JRE: Java Plug-in, which enables applets to run in popular browsers; and Java Web Start, which deploys standalone applications over a network.

(a) **Java Plug-in :** Java Plug-in technology, included as part of the Java Runtime Environment, Standard Edition (Java SE), establishes a connection between popular browsers and the Java platform. This connection enables applets on Web sites to be run within a browser on the desktop.

(b) **Java Web Start :** Using Java Web Start technology, standalone Java software applications can be deployed with a single click over the network. Java Web Start ensures the most current version of the application will be deployed, as well as the correct version of the Java Runtime Environment (JRE).

□ □ □

Chapter-4

Basic Features of Java Language

Q.1 What are Command Line Arguments?

Ans.: Command line arguments are parameters that are supplied to the application program at the time of invoking it for execution. We can write Java programs that can receive and use the arguments provided in the command line. *e.g. public static void main (String args[]).* args is declared as an array of strings and arguments provided in the command line are passed to the array args as its elements.

A Java application can accept any number of arguments from the command line. This allows the user to specify configuration information when the application is launched.

The user enters command-line arguments when invoking the application and specifies them after the name of the class to be run. *For example,* suppose a Java application called *Sort.java* sorts lines in a file. To sort the data in a file named *friends.txt*, a user would enter :

java Sort friends.txt

When an application is launched, the runtime system passes the command-line arguments to the application's main method via an array of Strings. In the previous example, the command-line arguments passed to the Sort application in an array that contains a single String: "friends.txt".

The **Echo** example displays each of its command-line arguments on a line by itself :

```
public class Echo {
    public static void main (String[] args) {
        for (String s: args) {
            System.out.println(s);
        }
    }
}
```

The following example shows how a user might run Echo. User input is in italics.

java Echo Drink Hot Java
Drink
Hot
Java

Note that the application displays each word — Drink, Hot, and Java — on a line by itself. This is because the space character separates command-line arguments. To have Drink, Hot, and Java interpreted as a single argument, the user would join them by enclosing them within quotation marks.

java Echo "Drink Hot Java"
Drink Hot Java

Q.2 Explain Type Casting?

OR

What is the process of Automatic Type Conversion?

Ans.: It is sometimes necessary to convert a data item of one type to another type. For example when it is necessary to perform some arithmetic using data items of different types (so called mixed mode arithmetic). Under certain circumstances Type conversion can be carried out automatically, in other cases it must be "forced" manually (explicitly).

Automatic Conversion : In Java type conversions are performed automatically when the type of the expression on the right hand side of an assignment operation can be safely promoted to the type of the variable on the left hand side of the assignment. Thus we can safely assign: byte -> short -> int -> long -> float -> double.

For example :

```
//64 bit long integer
long myLongInteger;
//32 bit long integer
int myInteger;
myLongInteger=myInteger;
```

The extra storage associated with the long integer, in the above example, will simply be padded with extra zeros.

Explicit Conversion (Casting) : The above will not work the other way round. For example we cannot automatically convert a long to an int because the first requires more storage than the second and consequently information may be lost. To force such a conversion we must carry out an explicit conversion (assuming of course that the long integer will fit into a standard integer). This is done using a process known as a type cast :

```
myInteger = (int) myLongInteger
```

This tells the compiler that the type of myLongInteger must be temporarily changed to a int when the given assignment statement is processed. Thus, the cast only lasts for the duration of the assignment. Java type casts have the following form: (T) N where T is the name of a numeric type and N is a data item of another numeric type. The result is of type T.

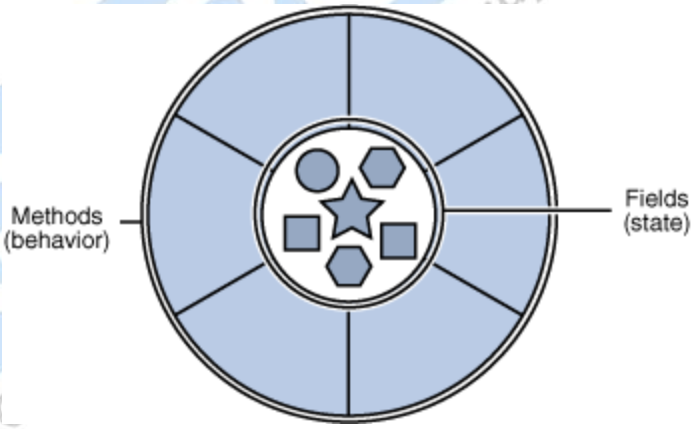
Q.3 What is an Object?

Ans.: Objects are key to understand *object-oriented* technology. Look around right now and you'll find many examples of real-world objects: your dog, your desk, your television set, your bicycle.

So, *anything that exists in real world is an object.* **In other words an object is a real life entity.**

Real-world objects share two characteristics: They all have ***identity, state*** and ***behavior***. Dogs have state (name, color, breed, hungry) and behavior (barking, fetching, wagging tail). Bicycles also have state (current gear, current pedal cadence, current speed) and behavior (changing gear, changing pedal cadence, applying brakes). Identifying the state and behavior for real-world objects is a great way to begin thinking in terms of object-oriented programming.

Real-world objects vary in complexity. e.g. your desktop lamp may have only two possible states (on and off) and two possible behaviors (turn on, turn off), but your desktop radio might have additional states (on, off, current volume, current station) and behavior (turn on, turn off, increase volume, decrease volume, seek, scan, and tune). Some objects, in turn, will also contain other objects. These real-world observations all translate into the world of object-oriented programming.

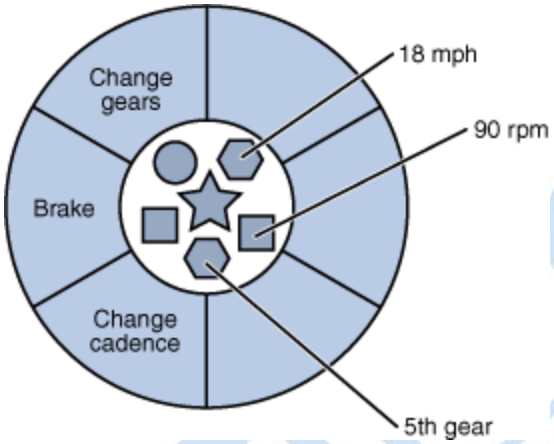


A software Object.

Software objects are conceptually similar to real-world objects: they too consist of state and related behavior. An object stores its state in ***fields*** (variables in some programming languages) and exposes its behavior through ***methods*** (functions in some programming languages). Methods operate on an object's internal state and serve as the primary mechanism

for object-to-object communication. Hiding internal state and requiring all interaction to be performed through an object's methods is known as *data encapsulation* — a fundamental principle of object-oriented programming.

Consider a bicycle, for example:



A Bicycle Modeled as a Software Object

By attributing state (current speed, current pedal cadence, and current gear) and providing methods for changing that state, the object remains in control of how the outside world is allowed to use it. For example, if the bicycle only has 6 gears, a method to change gears could reject any value that is less than 1 or greater than 6.

Q.4 What is a Class?

Ans.: In the real world, you'll often find many individual objects all of the same kind.

All the objects that have similar properties and similar behavior are grouped together to form a class.

In other words we can say *that a class is a user defined data type and objects are the instance variables of class.*

There may be thousands of other bicycles in existence, all of the same make and model. Each bicycle was built from the same set of blueprints and therefore contains the same components. In object-oriented terms, we

say that your bicycle is an *instance* of the *class of objects* known as bicycles. A *class* is the blueprint from which individual objects are created.

The following **Bicycle** class is one possible implementation of a bicycle :

```
class Bicycle {
        int cadence = 0;
        int speed = 0;
         int gear = 1;
         void changeCadence(int newValue) {
                cadence = newValue;
        }
        void changeGear(int newValue) {
                gear = newValue;
        }
        void printStates() {
         System.out.println("cadence:"+cadence+"speed:"+speed+" gear:"+gear);
        }
}
```

The fields cadence, speed, and gear represent the object's state, and the methods (changeCadence, changeGear, speedUp etc.) define its interaction with the outside world.

You may have noticed that the Bicycle class does not contain a main method. That's because it's not a complete application; it's just the blueprint for bicycles that might be *used* in an application.

Here's a BicycleDemo class that creates two separate Bicycle objects and invokes their methods :

```
class BicycleDemo {
        public static void main(String[] args) {
```

```
        // Create two different Bicycle objects
        Bicycle bike1 = new Bicycle();
        Bicycle bike2 = new Bicycle();
        // Invoke methods on those objects
        bike1.changeCadence(50);
        bike1.changeGear(2);
        bike1.printStates();
        bike2.changeCadence(50);
        bike2.changeGear(2);
        bike2.changeCadence(40);
        bike2.changeGear(3);
        bike2.printStates();
    }
}
```

The output of this test prints the ending pedal cadence, speed, and gear for the two bicycles :

Cadence : 50 speed : 10 gear : 2

Cadence : 40 speed : 20 gear : 3

Q.5 How many types of variables are available in Java?

Ans.: A variable is a place in memory where we can store a value. The Java programming language defines the following kinds of variables :

- **Instance Variables (Non-Static Fields) :** Technically speaking, objects store their individual states in "non-static fields", that is, fields declared without the static keyword. Non-static fields are also known as *instance variables* because their values are unique to each *instance* of a class (to each object, in other words); the

currentSpeed of one bicycle is independent from the currentSpeed of another.

- **Class Variables (Static Fields) :** A *class variable* is any field declared with the static modifier; this tells the compiler that there is exactly one copy of this variable in existence, regardless of how many times the class has been instantiated. A field defining the number of gears for a particular kind of bicycle could be marked as static since conceptually the same number of gears will apply to all instances. The code *static int numGears = 6*; would create such a static field. Additionally, the keyword *final* could be added to indicate that the number of gears will never change.
- **Local Variables :** Similar to how an object stores its state in fields, a method will often store its temporary state in *local variables*. The syntax for declaring a local variable is similar to declaring a field (for example, int count = 0;). There is no special keyword designating a variable as local; that determination comes entirely from the location in which the variable is declared — which is between the opening and closing braces of a method. As such, local variables are only visible to the methods in which they are declared; they are not accessible from the rest of the class.

Q.6 Explain Data Types in Java?

OR

How many Primitive Data Types are there in Java?

Ans.: The Java programming language is strongly-typed, which means that all variables must first be declared before they can be used. This involves stating the variable's type and name, as you've already seen :

int gear = 1;

Doing so tells your program that a field named "gear" exists, holds numerical data, and has an initial value of "1". A variable's data type determines the values it may contain, plus the operations that may be

performed on it. In addition to int, the Java programming language supports seven other *primitive data types*. A primitive type is predefined by the language and is named by a reserved keyword. Primitive values do not share state with other primitive values. The eight primitive data types supported by the Java programming language are :

- **byte** : The byte data type is an 8-bit signed two's complement integer. It has a minimum value of -128 and a maximum value of 127 (inclusive).
- **short :** The short data type is a 16-bit signed two's complement integer. It has a minimum value of -32,768 and a maximum value of 32,767 (inclusive). As with
- **int :** The int data type is a 32-bit signed two's complement integer. It has a minimum value of -2,147,483,648 and a maximum value of 2,147,483,647 (inclusive). For integral values, this data type is generally the default choice unless there is a reason (like the above) to choose something else
- **long :** The long data type is a 64-bit signed two's complement integer. It has a minimum value of -9,223,372,036,854,775,808 and a maximum value of 9,223,372,036,854,775,807 (inclusive). Use this data type when you need a range of values wider than those provided by int.
- **float :** The float data type is a single-precision 32-bit IEEE 754 floating point As with the recommendations for byte and short, use a float (instead of double) if you need to save memory in large arrays of floating point numbers. This data type should never be used for precise values, such as currency.
- **double :** The double data type is a double-precision 64-bit IEEE 754 floating point. For decimal values, this data type is generally the default choice. As mentioned above, this data type should never be used for precise values, such as currency.
- **boolean** : The boolean data type has only two possible values: *true* and *false*. Use this data type for simple flags that track true/false

conditions. This data type represents one bit of information, but its "size" isn't something that's precisely defined.

- **char** : The char data type is a single 16-bit Unicode character. It has a minimum value of '\u0000' (or 0) and a maximum value of '\uffff' (or 65,535 inclusive).

In addition to the eight primitive data types listed above, the Java programming language also provides special support for character strings via the java.lang.String class.

Q.7 What are Default Values?

Ans.: It's not always necessary to assign a value when a field is declared. Fields that are declared but not initialized will be set to a reasonable default by the compiler. Generally speaking, this default will be zero or null, depending on the data type

The following chart summarizes the default values for the above data types.

Data Type	Default Value (for fields)
byte	0
short	0
int	0
long	0L
float	0.0f
double	0.0d
char	'\u0000'
string (or any object)	null
boolean	false

Local variables are slightly different; the compiler never assigns a default value to an uninitialized local variable. If you cannot initialize your local variable where it is declared, make sure to assign it a value before you attempt to use it. Accessing an uninitialized local variable will result in a compile-time error.

Q.8 Mention the different types of operators in Java?

Ans.: Operators are special symbols that perform specific operations on one, two, or three *operands*, and then return a result.

The operators in the following table are listed according to precedence order Operators with higher precedence are evaluated before operators with relatively lower precedence. Operators on the same line have equal precedence. When operators of equal precedence appear in the same expression, a rule must govern which is evaluated first. All binary operators except for the assignment operators are evaluated from left to right; assignment operators are evaluated right to left.

Operator Precedence

Operators	Precedence
Postfix	*expr*++ *expr*--
Unary	++*expr* --*expr* +*expr* -*expr* ~ !
Multiplicative	* / %
Additive	+ -
Shift	<< >> >>>
Relational	< > <= >= instanceof
Equality	== !=

Operators	Precedence
Bitwise AND	&
Bitwise exclusive OR	^
Bitwise inclusive OR	|
Logical AND	&&
Logical OR	||
Ternary	? :
Assignment	= += -= *= /= %= &= ^= |= <<= >>= >>>=

Q.9 What are Enum Types and how they are helpful?

Ans.: An *enum type* is a type whose *fields* consist of a fixed set of constants.

Because they are constants, the names of an enum type's fields are in uppercase letters.

In the Java programming language, you define an enum type by using the **enum** keyword. For example assuming that Day Enum has already been defined by user than you would specify a days-of-the-week enum type as :

```
public class EnumTest {
        Day day;
        public EnumTest(Day day) {
                this.day = day;
        }
        public void tellItLikeItIs()
        {
        switch (day)
```

```
        {
        case MONDAY: System.out.println("Mondays are bad.");
                                        break;
        case FRIDAY: System.out.println("Fridays are better.");
                                        break;
        case SATURDAY:4 fv5rc
        case SUNDAY: System.out.println("Weekends are best.");
                                        break;
        default:        System.out.println("Midweek days are so-so.");
                                        break;
        }
        }
        public static void main(String[] args) {
                EnumTest firstDay = new EnumTest(Day.MONDAY);
                firstDay.tellItLikeItIs();
                EnumTest thirdDay = new EnumTest(Day.WEDNESDAY);
                thirdDay.tellItLikeItIs();
                EnumTest fifthDay = new EnumTest(Day.FRIDAY);
                fifthDay.tellItLikeItIs();
                EnumTest sixthDay = new EnumTest(Day.SATURDAY);
                sixthDay.tellItLikeItIs();
                EnumTest seventhDay = new EnumTest(Day.SUNDAY);
                seventhDay.tellItLikeItIs();
        }
}
```

The output is :

Mondays are bad.

Midweek days are so-so.

Fridays are better.

Weekends are best.

Weekends are best.

Q.10 What are Boolean Methods?

Ans.: Methods can return boolean values just like any other type, which is often convenient for hiding complicated tests inside methods. For example :

```
public static boolean isSingleDigit (int x) {
        if (x >= 0 && x < 10) {
                return true;
        } else {
                return false;
        }
 }
```

The name of this method is isSingleDigit. It is common to give boolean methods names that sound like yes/no questions. The return type is boolean, which means that every return statement has to provide a boolean expression.

There is nothing wrong with returning it directly, and avoiding the if statement altogether:

```
public static boolean isSingleDigit (int x) {
        return (x >= 0 && x < 10);
 }
```

In main you can invoke this method in the usual ways :

```
boolean bigFlag = !isSingleDigit (17);
System.out.println (isSingleDigit (2));
```

The first line assigns the value true to bigFlag only if 17 is *not* a single-digit number. The second line prints true because 2 is a single-digit number. Yes, println is overloaded to handle booleans, too.

The most common use of boolean methods is inside conditional statements

```
if (isSingleDigit (x)) {
        System.out.println ("x is little");
} else {
        System.out.println ("x is big");
}
```

□ □ □

Chapter-5

Declaration of Arrays

Q.1 How Arrays are declared and initialized in Java?

OR

What is an Array and how it is declared and initialized?

Ans.: An array is a group of contiguous or related data items that share a common name. For instance, we can define an array name sal to represent a set of salaries of a group of employees. A particular value is indicated by writing a number called index number or subscript in brackets after the array name.

For example : sal[10] represents the salary of the 10 employeeS. While the complete set of values is referred to as an array, the individual values are called elements.

The arrays are of different types :

One Dimensional (A list of items can be given one variable name using only one subscript and such a variable is called a single-subsubscripted variable or a one-dimensional array.)

Two Dimensional (Where a table of values will be stored and two subscripts are required to represent an element of array)

Three Dimensional (Where many table of values will be stored and three subscripts are required to represent an element of an array)

Creating an Array : It involves three steps :

(1) Declaring the Array

(2) Creating Memory Locations

(3) Putting Values into the Memory Locations

Declaring an Array : Arrays in Java are declared in one of the two forms :

Form 1 *type arrayname[];*

Form 2 *type[]arrayname;*

e.g. *int number[];*

int[] counter;

Creating Memory Allocations : To create an array we use new operator :

arrayname =new type[size];

e.g. *number =new int[5];*

Initialization of Array : The final step is to put values into the array created.

arrayname[subscript]=value;

e.g. *number[0]=35;*

…….

……....

number[4]=55;

Example of Two Dimensional Array :

int a[][]=new int[3][3] ; creates a table that can store 9 integer values.

Or

int a[2][3]={0,0,0,1,1,1}; will initializes the elements of the first row to zero and the second row to one .

□ □ □

Chapter-6

Control Structures of Java Language

Q.1 What are the Basic Control Structures in Java?

OR

What are the different Control Constructs?

OR

Explain the Looping Constructs?

OR

Explain the Conditional Constructs?

OR

What is the functioning of 'Break and Continue' statements?

Ans.: A Java program is a set of statements, which are normally executed sequentially in the order in which they appear. However, in practice, we have a number of situations, where we may have to change the order of execution of statements based on certain conditions, or repeat a group of statements until certain specified conditions are met.

Conditional Constructs : Java language possesses decision making capabilities and supports the following statements known as control or decision making statements :

(1) **If Statement :** It allows the computer to evaluate the expression first and then, depending on whether the value of the expression is 'true' or 'false'.

The general form is : if (test expression)

The if statement may be implemented in different forms :

(a) **Simple If Statement :**

The general form is :

```
if (test expression)
{
        statement-block;
}
 statement-x;
```

(b) **The If---Else Statement :**

The general form is :

```
if (test expression)
{
        true block statements;
}
else
{
        false block statements;
}
statement-x;
```

(c) **Nested If---Else Statement :**

```
if (test condition1)
{
        if(test condition2)
        {
                statement-1;
```

```
        }
        else
        {
                statement-2;
        }
}
statement-x;
```

(d) **Else If Ladder :**

```
if (condition1)
        statement-1;
else if (condition2)
        statement-2;
else if (condition)
        statement-n;
else
default-statement;
statement-x;
```

(2) **The Switch Statement :** When one of the many alternatives is to be selected, we can design a program using if statements to control the selection. However, when the number of alternatives increases, the program becomes difficult to read and follow. Then we can use switch statement in such situations.

The general form is :

```
switch(expression)
{
        case value1:
                block-1;
                break;
```

```
        case value-2:
                block-2;
                break;
        ……...
        ……..
        default:
                default-block;
                break;
}
statement-x;
```

(3) **The ?: Operator :**

The general form is :

Conditional expression ? expression1:expression2;

Looping Constructs : The process of repeatedly executing a block of statements is known as looping. The statements in the block may be executed any number of times, from zero to infinite number.

(a) **The While Statement :** The simplest of all looping structures in Java is the while statement.

The general format is

```
Initialization;
while (test condition)
{
        body of the loop
}
```

(b) **The Do Statement :** In this construct the body of the loop will execute first and then the test condition is evaluated.

```
Initialization;
do
{
        body of the loop
}
while(test condition);
```

(c) **For Statement :** This is another entry-controlled loop like while loop. The general format is :

```
For(initialization; test condition; increament/decrement)
{
        body of the loop
}
```

Jumps in Loops : Loops perform a set of operations repeatedly until the control variable fails to satisfy the test condition.Sometimes, it becomes desirable to skip a part of the loop or to leave the loop as soon as a certain condition occurs.

Jumping out of a loop---We can use the break statement which will immediately exit the control from the loop and the program continues with the statement immediately following the loop.

e.g.

```
while(…….)
{ ……..
        if(condition)
        break;
……..
……
}
……
```

Skipping a Part of Loop : During the loop operation it may be necessary to skip a part of the body of the loop under certain conditions. We can use continue statement for this.

```
e.g.    while(……...)
                {……
                        if(……...)
                        continue;
                ……
                }
```

The statements below continue statement are skipped and control jumps to header part of loop.

Q.2 Write a program to count the number of persons having weight >50 kg and height >170cm. and how many besides criteria?

Ans.:

```
/* A program to count the number of persons having weight>50 kg and
height>170 cm*/
        class  CountPersons
        {
                public static void main(String args[])
                {
                        int i, count, count1, count2;
                        float[] weight={45.0f,55.0f, 47.0f, 51.0f, 54.0f};
                        float[] height ={137.5f, 174.2f, 186.4f,170.7f, 169.9f};
                        count=0;
                        count1=0;
                        count2=0;
                        for(i=0;i<=4;i++)
                        {
```

```
                    if(weight[i]<50.0 && height[i]>170.0)
                    {
                        count1=count1+1;
                    }
                    count=count+1;
                }
                count2=count-count1;
                System.out.println("Number of persons with ….");
                System.out.println("weight <50 and height>170 ="+count1);
                System.out.println("Others ="+count2);
            }
        }
```

Q.3 Write a program to count the even and odd numbers in a given list of numbers?

Ans.: ** A program to count the even and odd numbers in a given list of numbers*/*

```
class EvenOdd
{
    public static void main(String args[])
    {
        int num[]={49,50,30,27,83};
        int even =0,odd=0;
        for(int i=0;i<number.length;i++)
    {
        if((num[i]%2)==0)
        {
```

```
                    even+=1;
                }
                    else
                {
                    odd+=1;
                }
            }
            System.out.println("Even Numbers "+even+  "Odd numbers" + odd);
        }
    }
```

Ques : What are if statement & switch statement in JAVA. Explain with example.

Answer :

if statement:

The if statement is the simple form of control flow statement. It directs the program to execute a certain section of code if and only if the test evaluates to true. That is the if statement in Java is a test of any boolean expression.

Syntax:

```
if (Expression) {
   statement (s)
}

else {

   statement (s)

}
```

Example:

```
public class MainClass {

  public static void main(String[] args) {
    int a = 3;
    if (a > 3)
      a++;
    else
      a = 3;
  }

}
```

Example:

```
class IfElseexamp {

    public static void main(String[] args) {


        int testscore = 66;

        char grade;


        if (testscore >= 90) {

            grade = 'A';

        } else if (testscore >= 80) {

            grade = 'B';

        } else if (testscore >= 70) {

            grade = 'C';

        } else if (testscore >= 60) {
```

```
            grade = 'D';
        } else {
            grade = 'F';
        }
        System.out.println("Grade = " + grade);
    }
}
```

□ □ □

Chapter-7

Constructors in Java

Q.1 What are Constructors? How Parameters are passed to Constructors?

OR

How Constructors are declared in Class?

Ans.: Constructor declarations look like method declarations—except that they use the name of the class and have no return type. For example, Bicycle has one constructor :

```
public Bicycle(int startCadence, int startSpeed, int startGear) {
        gear = startGear;
        cadence = startCadence;
        speed = startSpeed;
}
```

To create a new Bicycle object called myBike, a constructor is called by the new operator :

```
Bicycle myBike = new Bicycle(30, 0, 8);
```

new Bicycle(30, 0, 8) creates space in memory for the object and initializes its fields.

Although Bicycle only has one constructor, it could have others, including a no-argument constructor :

```
public Bicycle() {
        gear = 1;
```

```
        cadence = 10;
        speed = 0;
    }
```

Bicycle yourBike = new Bicycle(); invokes the no-argument constructor to create a new Bicycle object called yourBike.

Both constructors could have been declared in Bicycle because they have different argument lists. As with methods, the Java platform differentiates constructors on the basis of the number of arguments in the list and their types. You cannot write two constructors that have the same number and type of arguments for the same class, because the platform would not be able to tell them apart. Doing so causes a compile-time error.

You can use access modifiers in a constructor's declaration to control which other classes can call the constructor.

Q.2 What is the use of 'this' keyword?

Ans.: Within an instance method or a constructor, this is a reference to the ***current object*** — the object whose method or constructor is being called. You can refer to any member of the current object from within an instance method or a constructor by using '***this***'.

Using 'this' with a Field : The most common reason for using the this keyword is because a field is shadowed by a method or constructor parameter.

For example, the Point class was written like this

```
public class Point {
        public int x = 0;
        public int y = 0;
        //constructor
        public Point(int a, int b) {
            x = a;
            y = b;
```

```
    }
}
```

but it could have been written like this :

```
public class Point {
        public int x = 0;
        public int y = 0;
        //constructor
        public Point(int x, int y) {
                this.x = x;
                this.y = y;
        }
}
```

Each argument to the second constructor shadows one of the object's fields—inside the constructor **x** is a local copy of the constructor's first argument. To refer to the Point field **x**, the constructor must use **this.x**.

□ □ □

Chapter-8

Nesting of classes

Q.1 What are Nested Classes?

OR

Why do we use Nesting of Classes?

Ans.: Nested Classes : The Java programming language allows you to define a class within another class. Such a class is called a *nested class* and is illustrated here :

```
class OuterClass {
        ...
        class NestedClass {
                ...
        }
}
```

Terminology : Nested classes are divided into two categories: static and non-static. Nested classes that are declared static are simply called *static nested classes*. Non-static nested classes are called *inner classes*.

```
class OuterClass {
        ...
        static class StaticNestedClass {
                ...
        }
        class InnerClass {
                ...
         }
}
```

A nested class is a member of its enclosing class. Non-static nested classes (inner classes) have access to other members of the enclosing class, even if they are declared private. Static nested classes do not have access to other members of the enclosing class. As a member of the OuterClass, a nested class can be declared private, public, protected, or *package private*. (Recall that outer classes can only be declared public or *package private*.)

Q.2 Why do we use Nested Classes?

Ans.: There are several compelling reasons for using nested classes, among them :

- It is a way of logically grouping classes that are only used in one place.
- It increases encapsulation.
- Nested classes can lead to more readable and maintainable code.

Logical Grouping of Classes : If a class is useful to only one other class, then it is logical to embed it in that class and keep the two together. Nesting such "helper classes" makes their package more streamlined.

Increased Encapsulation : Consider two top-level classes, A and B, where B needs access to members of A that would otherwise be declared private. By hiding class B within class A, A's members can be declared private and B can access them. In addition, B itself can be hidden from the outside world.

More Readable, Maintainable Code : Nesting small classes within top-level classes places the code closer to where it is used.

Q.3 Explain different types of Nested Classes.

Ans.: **Static Nested Classes :** As with class methods and variables, a static nested class is associated with its outer class. And like static class methods, a static nested class cannot refer directly to instance variables or methods defined in its enclosing class — it can use them only through an object reference.

Note : A static nested class interacts with the instance members of its outer class (and other classes) just like any other top-level class. In effect, a static

nested class is behaviorally a top-level class that has been nested in another top-level class for packaging convenience.

Static nested classes are accessed using the enclosing class name :

OuterClass.StaticNestedClass

For example, to create an object for the static nested class, use this syntax :

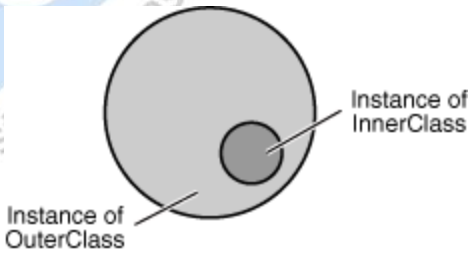
OuterClass.StaticNestedClass nestedObject = new OuterClass.StaticNestedClass();

Inner Classes : As with instance methods and variables, an inner class is associated with an instance of its enclosing class and has direct access to that object's methods and fields. Also, because an inner class is associated with an instance, it cannot define any static members itself.

Objects that are instances of an inner class exist *within* an instance of the outer class. Consider the following classes :

```
class OuterClass {
        ...
        class InnerClass {
                ...
        }
}
```

An instance of Inner Class can exist only within an instance of Outer Class and has direct access to the methods and fields of its enclosing instance. The next figure illustrates this idea.



An Inner Class Exists Within an Instance of Outer Class

To instantiate an inner class, you must first instantiate the outer class. Then, create the inner object within the outer object with this syntax:

OuterClass.InnerClass innerObject = outerObject.new InnerClass();

□ □ □

Chapter-9

Interfaces and Inheritance

Q.1 What are Interfaces?

OR

How do we implement Multiple Inheritance in 'Java'?

OR

How do we declare and implement Interfaces?

Ans.: Interfaces and Multiple Inheritance : Interfaces have another very important role in the Java programming language. Interfaces are not part of the class hierarchy, although they work in combination with classes. The Java programming language does not permit multiple inheritance, but interfaces provide an alternative.

In Java, a class can inherit from only one class but it can implement more than one interface. Therefore, objects can have multiple types: the type of their own class and the types of all the interfaces that they implement. This means that if a variable is declared to be the type of an interface, its value can reference any object that is instantiated from any class that implements the interface.

Defining an Interface : An interface declaration consists of modifiers, the keyword interface, the interface name, a comma-separated list of parent interfaces (if any), and the interface body. For example:

```
public interface GroupedInterface extends Interface1, Interface2, Interface3 {
    // constant declarations
```

```
        double E = 2.718282;  // base of natural logarithms
        // method signatures
        void doSomething (int i, double x);
        int doSomethingElse(String s);
    }
```

The public access specifier indicates that the interface can be used by any class in any package. If you do not specify that the interface is public, your interface will be accessible only to classes defined in the same package as the interface.

An interface can extend other interfaces, just as a class can extend or subclass another class. However, whereas a class can extend only one other class, an interface can extend any number of interfaces. The interface declaration includes a comma-separated list of all the interfaces that it extends.

The Interface Body : The interface body contains method declarations for all the methods included in the interface. A method declaration within an interface is followed by a semicolon, but no braces, because an interface does not provide implementations for the methods declared within it. All methods declared in an interface are implicitly public, so the public modifier can be omitted.

An interface can contain constant declarations in addition to method declarations. All constant values defined in an interface are implicitly public, static, and final. Once again, these modifiers can be omitted.

Implementing an Interface : To declare a class that implements an interface, you include an implements clause in the class declaration. Your class can implement more than one interface, so the implements keyword is followed by a comma-separated list of the interfaces implemented by the class.

```
public interface Relatable
{
        public int isLargerThan(Relatable other);
}
```

```
public class RectanglePlus implements Relatable {
    public int width = 0;
    public int height = 0;
    public Point origin;
      // four constructors
      public RectanglePlus() {
            origin = new Point(0, 0);
    }
    public RectanglePlus(Point p) {
    origin = p;
    }
    public RectanglePlus(int w, int h) {
          origin = new Point(0, 0);
          width = w;
          height = h;
    }

public RectanglePlus(Point p, int w, int h) {
          origin = p;
          width = w;
          height = h;
    }
    // a method for moving the rectangle
    public void move(int x, int y) {
          origin.x = x;
          origin.y = y;
    }
```

```
        // a method for computing the area of the rectangle
        public int getArea() {
                return width * height;
        }
        // a method to implement Relatable
        public int isLargerThan(Relatable other) {
                RectanglePlus otherRect = (RectanglePlus)other;
                if (this.getArea() < otherRect.getArea())
                        return -1;
                else if (this.getArea() > otherRect.getArea())
                        return 1;
                else
                        return 0;
        }
}
```

Because RectanglePlus implements Relatable, the size of any two RectanglePlus objects can be compared.

Q.2 Explain Inheritance with example?

Ans.: In the Java language, classes can be *derived* from other classes, thereby *inheriting* fields and methods from those classes.

Definitions : A class that is derived from another class is called a *subclass* (also a *derived class*, *extended class*, or *child class*). The class from which the subclass is derived is called a *superclass* (also a *base class* or a *parent class*).

Excepting Object, which has no superclass, every class has one and only one direct superclass (single inheritance). In the absence of any other explicit superclass, every class is implicitly a subclass of Object.

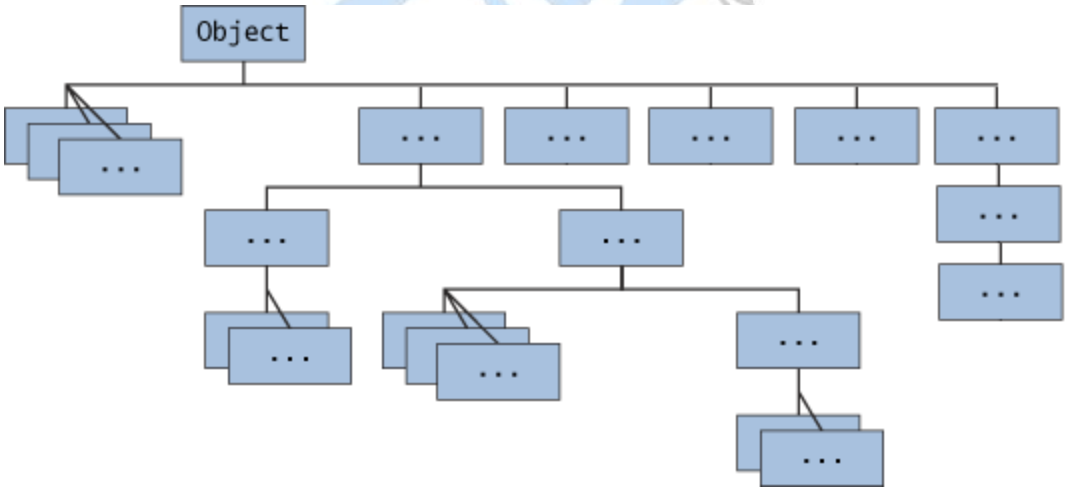
Classes can be derived from classes that are derived from classes that are derived from classes, and so on, and ultimately derived from the topmost

class, Object. Such a class is said to be *descended* from all the classes in the inheritance chain stretching back to Object.

The idea of inheritance is simple but powerful: When you want to create a new class and there is already a class that includes some of the code that you want, you can derive your new class from the existing class. In doing this, you can reuse the fields and methods of the existing class without having to write (and debug!) them yourself.

A subclass inherits all the *members* (fields, methods, and nested classes) from its superclass. Constructors are not members, so they are not inherited by subclasses, but the constructor of the superclass can be invoked from the subclass.

The Java Platform Class Hierarchy : The Object class, defined in the java.lang package, defines and implements behavior common to all classes—including the ones that you write. In the Java platform, many classes derive directly from Object, other classes derive from some of those classes, and so on, forming a hierarchy of classes.



All Classes in the Java Platform are Descendants of Object

At the top of the hierarchy, Object is the most general of all classes. Classes near the bottom of the hierarchy provide more specialized behavior.

```
/*An Example of Inheritance*/
public class Bicycle {
        // the Bicycle class has three fields
        public int cadence;
        public int gear;
        public int speed;
        // the Bicycle class has one constructor
        public Bicycle(int startCadence, int startSpeed, int startGear) {
                gear = startGear;
                cadence = startCadence;
                speed = startSpeed;
        }
        // the Bicycle class has four methods
        public void setCadence(int newValue) {
                cadence = newValue;
        }
        public void setGear(int newValue) {
                gear = newValue;
        }
        public void applyBrake(int decrement) {
                speed -= decrement;
        }
        public void speedUp(int increment) {
                speed += increment;
        }
}
```

A class declaration for a MountainBike class that is a subclass of Bicycle might look like this :

```
public class MountainBike extends Bicycle {
        // the MountainBike subclass adds one field
        public int seatHeight;
        // the MountainBike subclass has one constructor
        public MountainBike(int startHeight, int startCadence, int startSpeed, int
        startGear) {
                super(startCadence, startSpeed, startGear);
                seatHeight = startHeight;
        }
        // the MountainBike subclass adds one method
        public void setHeight(int newValue) {
                seatHeight = newValue;
        }
  }
```

MountainBike inherits all the fields and methods of Bicycle and adds the field seatHeight and a method to set it.

Q.2 Explain the difference between Composition and Inheritance?

Ans.: Modeling the relationships between types is a fundamental part of the process of object-oriented design

Composition : As you progress in an object-oriented design, you will likely encounter objects in the problem domain that contain other objects.. In an object-oriented design of a Java program, the way in which you model objects that contain other objects is with *composition*, the act of composing a class out of references to other objects. For example, it might be useful if the coffee cup object of your program could contain coffee. Coffee itself could be a distinct class, which your program could instantiate. You would award coffee with a type if it exhibits behavior that

is important to your solution. Perhaps it will swirl one way or another when stirred, keep track of a temperature that changes over time, or keep track of the proportions of coffee and any additives such as cream and sugar.

To use composition in Java, you use instance variables of one object to hold references to other objects. For the CoffeeCup example, you could create a field for coffee within the definition of class CoffeeCup, as shown below :

```
class CoffeeCup {
        private Coffee innerCoffee;
        public void addCoffee(Coffee newCoffee) {
                // no implementation yet
        }
        public Coffee releaseOneSip(int sipSize) {
                // no implementation yet
                // (need a return so it will compile)
                return null;
        }
        public Coffee spillEntireContents() {
                // no implementation yet
                // (need a return so it will compile)
                return null;
        }
}

// In Source Packet in file inherit/ex1/Coffee.java
public class Coffee {
        private int mlCoffee;
```

```
    public void add(int amount) {
        // No implementation yet
    }
    public int remove(int amount) {
        // No implementation yet
        // (return 0 so it will compile)
        return 0;
    }
    public int removeAll() {
        // No implementation yet
        // (return 0 so it will compile)
        return 0;
    }
}
```

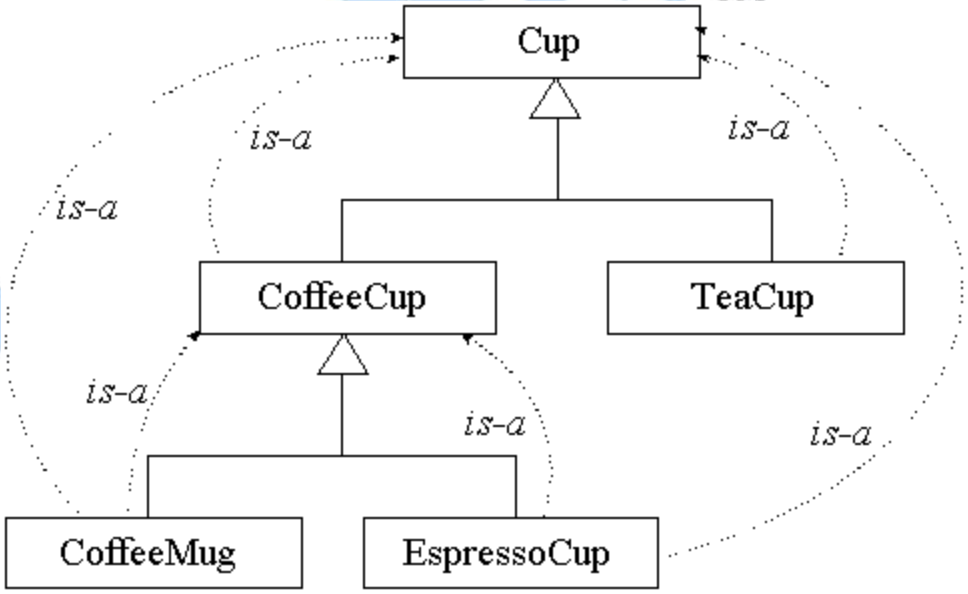
In the example above, the CoffeeCup class contains a reference to one other object, an object of type Coffee. Class Coffee is defined is a separate source file.

The relationship modeled by composition is often referred to as the "has-a" relationship. In this case a CoffeeCup *has* Coffee. As you can see from this example, the has-a relationship doesn't mean that the containing object must have a constituent object at all times, but that the containing object may have a constituent object at some time. Therefore the CoffeeCup may at some time contain Coffee, but it need not contain Coffee all the time. (When a CoffeeCup object doesn't contain Coffee, its innerCoffee field is null.)

Inheritance : As you partition your problem domain into types you will likely want to model relationships in which one type is a more specific or specialized version of another. For example you may have identified in your problem domain two types, Cup and CoffeeCup, and you want to be

able to express in your solution that a CoffeeCup is a more specific kind of Cup (or a special kind of Cup). In an object-oriented design, you model this kind of relationship between types with inheritance.

Building Inheritance Hierarchies : The relationship modeled by inheritance is often referred to as the "is-a" relationship. In the case of Cup and CoffeeCup, a "CoffeeCup is-a Cup." Inheritance allows you to build hierarchies of classes. A CoffeeCup is a more specific kind of Cup. A CoffeeMug is a more specific kind of CoffeeCup. Note also that the is-a relationship holds even for classes that are connected in the tree through other classes. For instance, a CoffeeMug is not only more specific version of a CoffeeCup, it is also a more specific version of a Cup. Therefore, the is-a relationship exists between CoffeeMug and Cup: a CoffeeMug is-a Cup.



The is-a Relationship of Inheritance

When programming in Java, you express the inheritance relationship with the extends keyword :

```
class Cup {
}
class CoffeeCup extends Cup {
}
class CoffeeMug extends CoffeeCup {
}
```

In Java terminology, a more general class in an inheritance hierarchy is called a *superclass*. A more specific class is a *subclass.* In figure Cup is a superclass of both CoffeeCup and CoffeeMug. Going in the opposite direction, both CoffeeMug and CoffeeCup are subclasses of Cup. When two classes are right next to each other in the inheritance hierarchy, their relationship is said to be *direct*. For example Cup is a *direct superclass* of CoffeeCup, and CoffeeMug is a *direct subclass* of CoffeeCup.

The act of declaring a direct subclass is referred to in Java circles as *class extension*. For example, a Java guru might be overheard saying, "Class CoffeeCup *extends* class Cup." Owing to the flexibility of the English language, Java in-the-knows may also employ the term "subclass" as a verb, as in "Class CoffeeCup *subclasses* class Cup." One other way to say the same thing is, "Class CoffeeCup *descends from* class Cup."

In Java, every class descends from one common base class: **Object.**

In Java, a class can have only one direct superclass. In object-oriented parlance, this is referred to as *single inheritance* . It contrasts with *multiple inheritance* , in which a class can have multiple direct superclasses. Although Java only supports single inheritance of classes through class extension, it supports a special variant of multiple inheritance through "interface implementation."

Q.3 What is Method Overriding and Method Overloading?

An.: Method Overloading : In Java, it is possible to create methods that have the same name, but different parameter lists and different definitions. This is called method overloading. Method overloading is used when objects are required to perform similar tasks but using different input parameters. When we call a method in an object, Java matches up the method name

first and then the number and type of parameters to decide which one of the definitions to execute. This process is known as polymorphism.

To create an overloaded method , all we have to do is to provide several different method definitions in the class, all with the same name, but with different parameter lists.

e.g. / An example of constructor overloading*/*

```
class Room
{
        float len;
        float wid;
        Room (float a, float b)
        {
                len=a;
                wid=b;
        }
        Room(float x)
        {
                len=wid=x;
        }
        int area()
        {
                return (len*wid);
        }
}
```

Method Overriding : Method inheritance enables us to define and use methods repeatedly subclasses without having to define the methods again in subclass.

However, there may be occasions when we want an object to respond to the same method but have different behaviour when that method is

called. That means , we should override the method defined in the superclass. This is possible by defining a method in the subclass that has the same name, same arguments and same return type as a method in the superclass. This is known as overriding.

```
e.g. /* An example of method overriding*/
class Super
{
        int x;
        Super(int x)
        {
                this.x=x;
        }
        void disp()
        {
                System.out.println("Super x="+x);
        }
}
class Sub extends Super
{
        int y;
        Sub(int x, int y)
        {
                super(x);
                this.y=y;
        }
        void disp()
        {
```

```
            System.out.println("Super x="+x);
            System.out.println("Sub y="+y);
        }
}
class OverrideTest
{
        public static void main(String args[])
        {
            Sub s1=new Sub(100,200);
            s1.display();
        }
}
```

Q.4 What are Final Classes and Methods?

Ans.: Writing Final Classes and Methods :

Final Methods : You can declare some or all of a class's methods *final*. You use the final keyword in a method declaration to indicate that the method cannot be overridden by subclasses. The Object class does this—a number of its methods are final.

You might wish to make a method final if it has an implementation that should not be changed and it is critical to the consistent state of the object. For example, you might want to make the getFirstPlayer method in this ChessAlgorithm class final :

```
class ChessAlgorithm {
        enum ChessPlayer { WHITE, BLACK }
        ...
        final ChessPlayer getFirstPlayer() {
            return ChessPlayer.WHITE;
```

```
    }
    ...
}
```

Methods called from constructors should generally be declared final. If a constructor calls a non-final method, a subclass may redefine that method with surprising or undesirable results.

Final Variables : To prevent the subclasses from overrding the member variables of the superclass, we can declare them as final using the final as a modifier.

e.g. final int SIZE =55;

Final Classes : You can also declare an entire class final — this prevents the class from being subclassed. This is particularly useful, for example, when creating an immutable class like the String class. You can use final modifier with class as follows:

```
e.g. final class A
        {
            }
```

Q.5 What are Abstract Methods and Classes?

OR

Explain the difference between Abstract Classes and Interfaces?

Ans.: Abstract Methods and Classes : An *abstract class* is a class that is declared abstract—it may or may not include abstract methods. Abstract classes cannot be instantiated, but they can be subclassed.

An *abstract method* is a method that is declared without an implementation (without braces, and followed by a semicolon), like this :

abstract void moveTo(double deltaX, double deltaY);

If a class includes abstract methods, the class itself *must* be declared abstract, as in :

```
public abstract class GraphicObject {
        // declare fields
        // declare non-abstract methods
        abstract void draw();
}
```

When an abstract class is subclassed, the subclass usually provides implementations for all of the abstract methods in its parent class. However, if it does not, the subclass must also be declared abstract.

Note : All of the methods in an *interface* are *implicitly* abstract, so the abstract modifier is not used with interface methods (it could be—it's just not necessary).

Abstract Classes versus Interfaces : Unlike interfaces, abstract classes can contain fields that are not static and final, and they can contain implemented methods. Such abstract classes are similar to interfaces, except that they provide a partial implementation, leaving it to subclasses to complete the implementation. If an abstract class contains *only* abstract method declarations, it should be declared as an interface instead.

Multiple interfaces can be implemented by classes anywhere in the class hierarchy, whether or not they are related to one another in any way. But in Java a single class can be inherited whether the class being extended is abstract or not.

By comparison, abstract classes are most commonly subclassed to share pieces of implementation. A single abstract class is subclassed by similar classes that have a lot in common (the implemented parts of the abstract class), but also have some differences (the abstract methods).

e.g.

```
abstract class GraphicObject {
        int x, y;
        ...
        void moveTo(int newX, int newY) {
                ...
```

```
        }
        abstract void draw();
        abstract void resize();
}
```

Each non-abstract subclass of GraphicObject, such as Circle and Rectangle, must provide implementations for the draw and resize methods :

```
class Circle extends GraphicObject {
        void draw() {
                ...
        }
        void resize() {
                ...
        }
}
class Rectangle extends GraphicObject {
        void draw() {
                ...
        }
        void resize() {
                ...
        }
}
```


□ □ □

Chapter-10

Strings and their Manipulation

Q.1 What are Strings? How Strings are manipulated in 'Java'?

OR

How can we concatenate two Strings?

Ans.: Strings : Strings, which are widely used in Java programming, are a sequence of characters. In the Java programming language, strings are objects.

The Java platform provides the **String** class to create and manipulate strings.

Creating Strings : The most direct way to create a string is to write :

```
String greeting = "Hello world!";
```

In this case, "Hello world!" is a *string literal*—a series of characters in your code that is enclosed in double quotes. Whenever it encounters a string literal in your code, the compiler creates a String object with its value—in this case, Hello world!

The String class has 11 constructors that allow you to provide the initial value of the string using different sources, such as an array of characters :

```
char[] helloArray = { 'h', 'e', 'l', 'l', 'o', '.'};
String helloString = new String(helloArray);
System.out.println(helloString);
```

The last line of this code snippet displays hello.

String Length : Methods used to obtain information about an object are known as *accessor methods*. One accessor method that you can use with strings is the length() method, which returns the number of characters contained in the string object. After the following two lines of code have been executed, len equals 17 :

```
String palindrome = "Dot saw I was Tod";
int len = palindrome.length();
```

Concatenating Strings : The *String* class includes a method for concatenating two strings:

string1.concat(string2);

This returns a new string that is string1 with string2 added to it at the end.

You can also use the concat() method with string literals, as in :

```
"My name is ".concat("Rumplestiltskin");
Strings are more commonly concatenated with the + operator, as in
"Hello," + " world" + "!"
which results in "Hello, world!"
```

The + operator is widely used in print statements. For example :

```
String string1 = "saw I was ";
System.out.println("Dot " + string1 + "Tod");
```

□ □ □

Chapter-11

Packages in 'Java'

Q.1 What are Packages?

OR

How Packages are created and used?

Ans.: Creating and Using Packages : To make types easier to find and use, to avoid naming conflicts, and to control access, programmers bundle groups of related types into packages.

Definition: A *package* is a grouping of related types providing access protection and name space management. Note that *types* refers to classes, interfaces, enumerations, and annotation types.

The fundamental classes are in java.lang. Classes for reading and writing (input and output) are in java.io, and so on. You can put your types in packages too.

Creating a Package : To create a package, you choose a name for the and put a package statement with that name at the top of *every source file* that contains the types (classes, interfaces, enumerations, and annotation types) that you want to include in the package.

The package statement (for example, package graphics;) must be the first line in the source file. There can be only one package statement in each source file, and it applies to all types in the file.

Note : If you put multiple types in a single source file, only one can be public, and it must have the same name as the source file.

e.g.

If you put the graphics interface and classes in a package called graphics, you would need to be written in source files, like this :

```
//in the Draggable.java file
package graphics;
public interface Draggable {
        . . .
}

//in the Graphic.java file
package graphics;
public abstract class Graphic {
        . . .
}

//in the Circle.java file
package graphics;
public class Circle extends Graphic implements Draggable {
        . . .
}
```

If you do not use a package statement, your type ends up in an unnamed package. Generally speaking, an unnamed package is only for small or temporary applications or when you are just beginning the development process. Otherwise, classes and interfaces belong in named packages.

□ □ □

Chapter-12

Basic Input/Output Streams

Q.1 What are Streams?

OR

What is the use of Data Input Stream and Data Output Stream ?

Ans.: Java uses the concept of streams to represent the ordered sequence of data, a common characteristic shared by all the input/output devices. A stream presents a uniform, easy-to-use, object-oriented interface between the program and the input/output devices.

A stream in Java is a path along which data flows. Both the source and the destination may be physical devices or programs or other streams in the same program.

The concept of sending data from one stream to another has made streams in Java a powerful tool for file processing also.

Stream Classes : The java.io package contains a large number of stream classes that provide capabilities for processing all types of data. These classes may categorized into two groups based on the data type on which they operate.

(1) **Byte Stream Classes :** Provide support for handling I/O operations on bytes.

(2) **Character stream Classes :** Provide support for managing I/O operations on characters.

Byte Stream Classes : Byte Stream classes have been designed to provide functional features for creating and manipulating streams and files for

reading and writing bytes. Since the streams are unidirectional, they can transmit bytes in only one direction and, therefore, Java provides two kinds of byte stream classes: input stream classes and output stream classes

(a) **Input Stream Classes :** Input stream classes are that used to read 8-bit bytes include a super class known as Input Stream and a number of subclasses for supporting various input-related functions.

The Input Stream Class defines methods for performing input functions such as :

Reading bytes

Closing streams

Marking positions in streams

Skipping ahead in a stream

Finding the number of bytes in a stream

Some methods of InputStream are read(), skip(n),reset(), close() etc.

The class DataInputStream extends FilterInput Stream and implements the interface DataInput. Therefore the DataInputStream class implements the methods described in DataInput in addition to using the methods of Input Stream class.

Some methods of DataInputStream are readShort(), readInt(), readLong(), readFloat(), readLine() etc.

(b) **Output Stream Classes :** Output stream classes are derived from the base class OutputStream. Like InputStream, the OutputStream is an abstract class and therefore we cannot instantiate it.

The OutputStream includes methods that are designed to perform the following tasks:

Writing bytes

Closing streams

Flushing streams

Some methods of OutputStream are write(), close(), flush() etc.

The DataOutputStream, implements the interface DataOutput and therefore implements methods like writeShort(), writeBytes(), writeInt(), writeLong() etc.

This page shows you how to use the DataInputStream and DataOutputStream classes from java.io using an example, DataIOTest, that reads and writes tabular data like this.

19.99	12	Java T-shirt
9.99	8	Java Mug

DataOutputStream, like other filtered output streams, must be attached to some other OutputStream. In this case, its attached to a FileOutputStream set up to write to a file on the file system named invoice1.txt.

```
DataOutputStream dos = new DataOutputStream(
                        new FileOutputStream("invoice1.txt"));
```

Next, DataIOTest uses DataOutputStream's specialized writeXXX() methods to write the invoice data (contained within arrays in the program).

```
for (int i = 0; i < prices.length; i ++) {
        dos.writeDouble(prices[i]);
        dos.writeChar('\t');
        dos.writeInt(units[i]);
        dos.writeChar('\t');
        dos.writeChars(descs[i]);
        dos.writeChar('\n');
}
dos.close();
```

Note that this code snippet closes the output stream when its finished. The close() method flushes the stream before closing it.

Next, DataIOTest opens a DataInputStream on the file just written:

```
DataInputStream dis = new DataInputStream(
                        new FileInputStream("invoice1.txt"));
```

DataInputStream, like other filtered input streams, must be attached to some other InputStream. In this case, its attached to a FileInputStream set up to read from a file on the file system named invoice1.txt. DataIOTest then just reads the data back in using DataInputStream's specialized readXXX() methods to read the input data into Java variables of the correct type.

```
while (!EOF) {
        try {
                price = dis.readDouble();
                dis.readChar();      // throws out the tab
                unit = dis.readInt();
                dis.readChar();      // throws out the tab
                desc = dis.readLine();
                System.out.println("You've ordered " + unit + " units of "
                + desc + " at " + price);
                total = total + unit * price;
        } catch (EOFException e) {
                EOF = true;
        }
}
System.out.println("For a TOTAL of: " + total);
dis.close();
```

When all of the data has been read, DataIOText displays a statement summarizing the order and the total amount owed, and closes the stream.

Note the loop that DataIOTest uses to read the data from the DataInputStream. Normally, when reading you see loops like this:

```
while ((input = dis.readLine()) != null) {
        . . .
}
```

The readLine() method returns some value, null, that indicates that the end of the file has been reached.

Character Stream Classes : Character streams can be used to read and write 16-bit Unicode characters. There are two kinds of character stream classes, namely, reader stream classes and writer stream classes.

(a) **Reader Stream Classes :** Reader stream classes are designed to read character from the files. Reader class is the base class for all other classes. These classes are functionally very similar to the input stream classes, except input streams use bytes as their fundamental unit of information, while reader streams use characters.

The **Reader** class contains methods that are identical to those available in the **Input Stream** class, except Reader is designed to handle characters.

(b) **Writer Stream Classes :** Like output stream classes, the writer stream classes are designed to perform all output operations on files. Only difference is that while output stream classes are designed to write bytes, the writer stream classes are designed to write characters.

The **Writer** class is an abstract class which acts as abase class for all the other writer stream classes. This base class provides support for all output operations by defining methods that are identical to those in **Output Stream** class.

□ □ □

Chapter-13

Threads and their Working

Q.1 What is Multithreading?

OR

What are Threads and how are they implemented in Java?

OR

Explain various states of lifecycle of a Thread?

Ans.: Computer users take it for granted that their systems can do more than one thing at a time. They assume that they can continue to work in a word processor, while other applications download files, manage the print queue, and stream audio. Even a single application is often expected to do more than one thing at a time. For example, that streaming audio application must simultaneously read the digital audio off the network, decompress it, manage playback, and update its display.

The Java platform is designed from the ground up to support concurrent programming, with basic concurrency support in the Java programming language and the Java class libraries. Since version 5.0, the Java platform has also included high-level concurrency APIs.

In concurrent programming, there are two basic units of execution: *processes* and *threads*. In the Java programming language, concurrent programming is mostly concerned with threads. However, processes are also important.

Processes : A process has a self-contained execution environment. A process generally has a complete, private set of basic run-time resources; in particular, each process has its own memory space.

Processes are often seen as synonymous with programs or applications. However, what the user sees as a single application may in fact be a set of cooperating processes. To facilitate communication between processes, most operating systems support *Inter Process Communication* (IPC) resources, such as pipes and sockets.

Threads : Threads are sometimes called *lightweight processes*. Both processes and threads provide an execution environment, but creating a new thread requires fewer resources than creating a new process.

Threads exist within a process — every process has at least one. Threads share the process's resources, including memory and open files. This makes for efficient, but potentially problematic, communication.

Multithreaded execution is an essential feature of the Java platform. Every application has at least one thread — or several, if you count "system" threads that do things like memory management and signal handling. But from the application programmer's point of view, you start with just one thread, called the *main thread*.

An application that creates an instance of Thread must provide the code that will run in that thread. There are two ways to do this:

Provide a Runnable object. The Runnable interface defines a single method, run, meant to contain the code executed in the thread. The Runnable object is passed to the Thread constructor, as in the HelloRunnable example :

```
public class HelloRunnable implements Runnable {
        public void run() {
                System.out.println("Hello from a thread!");
        }
        public static void main(String args[]) {
                (new Thread(new HelloRunnable())).start();
        }
}
```

Subclass Thread. The Thread class itself implements Runnable, though its run method does nothing. An application can subclass Thread, providing its own implementation of run, as in the HelloThread example :

```
public class HelloThread extends Thread {
        public void run() {
                System.out.println("Hello from a thread!");
        }
        public static void main(String args[]) {
                (new HelloThread()).start();
        }
}
```

Notice that both examples invoke Thread.start ( ) in order to start the new thread.

Which of these idioms should you use? The first idiom, which employs a Runnable object, is more general, because the Runnable object can subclass a class other than Thread. The second idiom is easier to use in simple applications, but is limited by the fact that your task class must be a descendant of Thread. The first approach more flexible, but it is applicable to the high-level thread management APIs covered later.

The Thread class defines a number of methods useful for thread management. These include static methods, which provide information about, or affect the status of, the thread invoking the method. The other methods are invoked from other threads involved in managing the thread and Thread object.

Stopping a Thread : Whenever we want to stop a thread from running further, we may do so by calling its stop() method which results in causing thread to dead state.

Blocking a Thread : A thread can also be temporarily suspended or blocked from entering into the runnable and subsequently running state by using either of the following thread methods :

sleep() //blocked for a specified time

suspend() //blocked until further orders

wait() //blocked until certain conditions occurs.

Life Cycle of a Thread : During the life time of a thread, there are many states it can enter. They include:

(1) **Newborn State :** When we create a thread object, the thread is born and is said to be in newborn state. The thread is not yet scheduled for running.

(2) **Runnable State :** It means that the thread is ready for execution and is waiting for the availability of the processor. That is , the thread has joined the queue of threads that are waiting for execution.

(3) **Running State** It means that the processor has given its time to the thread for its execution. The thread runs until it relinquishes control (using suspend(), sleep(), or notify())on its own or it is preempted by a higher priority thread.

(4) **Blocked State :** A thread is said to be blocked when it is prevented from entering into the runnable state and subsequently the running state. This happens when the thread is suspended, sleeping, or waiting in order to satisfy certain requirements.

(5) **Dead State :** A running thread ends its life when it has completed executing its run() method. It is a natural death. However we can kill it by sending the stop message to it at any state thus causing a premature death to it.

□ □ □

Chapter-14

Exception Handling

Q.1 What is an Exception?

OR

Explain how Exceptions are handled using Try-Catch Block?

OR

What is a Finally Block?

Ans.: The term *exception* is shorthand for the phrase "exceptional event."

Definition : An *exception* is an event, which occurs during the execution of a program, that disrupts the normal flow of the program's instructions.

When an error occurs within a method, the method creates an object and hands it off to the runtime system. The object, called an *exception object*, contains information about the error, including its type and the state of the program when the error occurred. Creating an exception object and handing it to the runtime system is called *throwing an exception*.

After a method throws an exception, the runtime system attempts to find something to handle it.

Some of the predefined exception classes are :

ArithmeticException

ArrayIndexOutOfBoundException

IO Exception

etc.

The Try Block : The first step in constructing an exception handler is to enclose the code that might throw an exception within a try block. In general, a try block looks like the following.

```
try {
        code
}
catch and finally blocks . . .
```

The segment in the example labeled *code* contains one or more legal lines of code that could throw an exception.

A Catch Block : A catch block defined by the keyword catch "catches" the exception "thrown" by the try block and handles it appropriately.The catch block is added immediately after the try block.

The general form is :

```
……...
……
try
{
        statement;
}
catch(Exception type e)
{
        statement;
}
……...
……
```

Multiple Catch Statements : It is possible to have more than one catch statements in the catch block.

e.g.

```
……..
……
try
{
        statement;
}
catch(Exception-Type-1 e)
{
        statement;
}
catch(Exception-Type-2 e)
{
        statement;
}
.
.
.
.
.
catch(Exception –Type-N e)
{
        statement;
}
……...
……..
```

Using Finally Statement : Java supports another statement known as finally statement that can be used to handle an exception that is not caught by any of the previous catch statements. Finally block can be used to handle any exception generated within a try block. It may be immediately after the try block or after the last catch block.

When a finally block is defined, this is guaranteed to execute, regardless of whether or not in exception is thrown.

Throwing Our Own Exceptions : There may be times when we would like to throw our own exceptions. We can do this by using the keyword throw as follows :

throw new Throwable_subclass;

e.g. throw new Arithmetic Exception();

□ □ □

Chapter-15

Graphics Programming

Q.1 What are the basic graphics functions used in 'Java'?

Ans.: One of the most important features of Java is the ability to draw graphics. Java's coordinate system has the origin(0,0) in the upper-left corner. Positive x values are to the right, and positive y values are to the bottom. The values of x and y are in pixels.

Graphics Class : Java's Graphics class includes methods for drawing many different types of shapes, from simple lines to polygons to text in a variety of fonts.

To draw a shape on the screen, we may call one of the methods available in the Graphics class. Some of the methods are :

Method	Description
clearRect()	Erases a rectangular area of the screen
copyArea()	Copies a rectangular area of the canvas to another area
drawArc()	Draws a hollow arc
drawLine()	Draws a straight line
drawOval()	Draws a hollow oval
drawPolygon()	Draws a hollow polygon
drawRect()	Draws a hollow rectangle
drawRoundRect()	Draws a hollow rectangle with rounded corners

drawstring()	Draws a text string
fillArc()	Draws a filled arc
fillOval()	Draws a filled oval
fillPolygon()	Draws a filled polygon
fillRect()	Draws a filled rectangle
fillRoundRect()	Draws a filled rectangle with rounded corners
getColor()	Retrieves the current drawing color
getFont()	Retrieves the currently used font
getFontMetrics()	Retrieves the information about the current font
setColor()	Sets the drawing colour
setFont()	Sets the font

e.g.

drawLine :

public abstract void ***drawLine****(int x1,*
int y1,
int x2,
int y2)

Draws a line, using the current color, between the points (x1, y1) and (x2, y2) in this graphics context's coordinate system.

Parameters :

x1 - the first point's *x* coordinate.

y1 - the first point's *y* coordinate.

x2 - the second point's *x* coordinate.

y2 - the second point's *y* coordinate.

fillRect :

public abstract void ***fillRect****(int x,*
int y,

int width,

int height)

Fills the specified rectangle. The left and right edges of the rectangle are at x and x + width - 1. The top and bottom edges are at y and y + height - 1. The resulting rectangle covers an area width pixels wide by height pixels tall. The rectangle is filled using the graphics context's current color.

Parameters:

x - the *x* coordinate of the rectangle to be filled.

y - the *y* coordinate of the rectangle to be filled.

width - the width of the rectangle to be filled.

height - the height of the rectangle to be filled.

□ □ □

Part-II
(Visual Basic)

Chapter-1

Integrated Development Environment

Q.1 What is an IDE?

Ans.: Visual Basic is not only a language. It's an *Integrated Development Environment* in which you can develop, run, test and debug your applications.

The types of project that you can create in Visual Basic are as follows :

(i) **Standard EXE :** These are the typical applications that you develop with previous versions of Visual Basic.

(ii) **ActiveX EXE, ActiveX DLL :** These types of projects are available with the Professional edition. ActiveX components are OLE automation servers.

(iii) **ActiveX Control :** This type of project is also a feature of the Professional edition. We use it to develop your own ActiveX controls.

(iv) **ActiveX Document EXE, ActiveX Document DLL :** ActiveX documents are in essence Visual Basic applications that can run in the environment of the container that supports hyper-linking.

(v) **VB Application Wizard, VB Wizard Manager :** The Application Wizard takes you through the steps of setting up the skeleton of a new application. The Wizard Manager lets you build your own wizard.

(vi) **Data Project :** It's identical to the Standard EXE project type, but it automatically adds the controls that are used in accessing databases to the Toolbox.

(vii) **DHTML Application :** VB6 allows you to build Dynamic HTML pages that can be displayed in the browser's window on a client computer.

(viii) **IIS Application :** VB6 allows you to build applications that run on the Web server and interact with clients over the Internet with the Internet Information Server.

(ix) **Addin :** You can create your own add-ins for the VB IDE. These are special commands you can add to Visual Basic's menus.

(x) **VB Enterprise Edition Controls :** It simply creates a new Standard EXE project and loads all the tools of the Enterprise Edition of Visual Basic.

Q.2 Explain the elements of the User Interface?

Ans.: The user interface is what appears in the application's window when it runs. It consists of various elements with which the user can interact and control the application. The first element of the user interface is the *Form.*

(i) **Picture Box :** This control is used to display images, and the images are set with Picture property

(ii) **Label :** This control displays text on a Form that the user can't edit.

(iii) **Text Box :** It displays text that the user can edit.

(iv) **Frame :** It is used to draw boxes on the Form and to group other elements.

(v) **Command Button :** It is the most common element of the Windows interface. It represents an action that is carried out when the user clicks the button.

(vi) **Check Box :** It presents one or more choices that the user can select.

(vii) **Option Button :** Also called as radio buttons, appear in groups, and the user can choose only one of them.

(viii) **Combo Box :** It is similar to the List Box control, but it contains a text Edit field. The user can choose an item from the list or enter in the Edit field.

(ix) **List Box :** It contains a list of options from which the user can choose one or more. The user can scroll the list to locate an item.

(x) **The Horizontal and Vertical Scroll Bars :** The user specify a magnitude by scrolling the control's button between its minimum and maximum value.

(xi) **Timer :** It is used to perform tasks at regular intervals.

(xii) **File System Controls :** These controls are used to add file-handling capabilities to your application

(xiii) **Image :** It is similar to the Picture Box control in that it can display images, but it supports only a few features of the Picture Box control and requires fewer resources.

(xiv) **Data :** It provides point and click access to data stored in data- bases.

(xv) **OLE :** It is a window you can place on your Form to host documents from other applications, such as Microsoft Word or Excel.

Q.3 What are the common properties of the Controls?

Ans.: The following properties apply to most of the objects :

- **Name :** It sets the name of the control, through which you can access the control's properties and methods.
- **Appearance :** It can be 0 for a flat look or 1 for a 3-D look.
- **Back Color :** It sets the background color on which text is displayed or graphics are drawn.
- **Fore Color :** It sets the foreground color

- **Font :** It sets the face, attribute, and size of the font used for the text on the control.
- **Caption :** It sets the text that is displayed on many controls that don't accept input, for example, the text on a Label control, the caption of a Command Button control.
- **Text :** It sets the text that is displayed on the controls that accept user input, for example, the TextBox control.
- **Width & Height :** These properties set the control's dimensions.
- **Left & Top :** These properties set the coordinates of the control's upper-left corner, expressed in the units of the container.
- **Enabled :** By default, this property's value is True, which means that the control can get the focus.
- **Visible :** Set this property to False to make a control invisible.

□ □ □

Chapter-2

Basic Programming Concepts

Q.1 How variables are declared in Visual Basic?

Ans.: Variables are used to store values during a program's execution. So, variables are place values in which you can leave values and recall them at will.

(a) **Declaring Variables :** In most programming languages, variables must be declared in advance for the compiler.

Explicit Declarations : To declare a variable, use the Dim statement followed by the variable's name and type as follows :

- Dim Meters as Integer
- Dim Names as String

Implicit Declarations : When Visual Basic meets an undeclared variable name, it creates a new variable on the spot and uses it. The new variable's type is variant, the generic data type that can accommodate all other data types.

Dim var1, var2

var1="Thank you"

var2=43.23

The var1 variable is a string variable, and var2 is a numeric one.

(b) **Types of Variables :** Visual Basic recognizes the following six types of variables :

- **Numeric :** The different numeric data types provided are :
  (a) Integers
  (b) Single, or floating point numbers with limited precision
  (c) Double, or floating point numbers with extreme precision
- **String :** This data type stores only text and string variables are declared with the String type.
- **Boolean :** This data type stores True/False values.
- **Date :** Date and time values are double-precision numbers: the integer part represents the date and the fractional part represents the time.
- **Object :** An object variable refers to one of Visual Basic's many objects, and you can use an object variable to access the actual object.
- **Variant :** This is the most flexible data type because it can accommodate all other types.

Q.2 What is the Variable's Scope and Lifetime of a Variable?

Ans.: A Variable's Scope : The scope of a variable is the section of the application that can see and manipulate the variable. If a variable is declared within a procedure, only the code in the specific procedure has access to that variable. When the variable's scope is limited to a procedure it's called *local.*

e.g.

```
Private Sub Command1_Click()
Dim i as Integer
Dim Sum as Integer
For i=0 to 100 Step 2
Sum = Sum +i
Next
MsgBox " The Sum is "& Sum
End Sub
```

A variable whose value is available to all procedures within the same Form or Module are called *Form-wide or Module-wide* and can be accessed from within all procedures in a component.

In some situations the entire application must access a certain variable. Such variable must be declared as *Public*.

Lifetime of a Variable : It is the period for which they retain their value. Variables declared as Public exist for the lifetime of the application. Local variables, declared within procedures with the Dim or Private statement, live as long as the procedure.

You can force a local variable to preserve its value between procedure calls with the *Static* keyword. The advantage of using static variables is that they help you minimize the number of total variables in the application.

Variables declared in a Form outside any procedure take effect when the Form is loaded and cease to exist when the Form is unloaded. If the Form is loaded again, its variables are initialized, as if it's being loaded for the first time.

Q.3 What are constants?

Ans.: Some variables don't change value during the execution of program. These are constants that appear many times in your code. The constants are preferred for two reasons :

- Constants don't change value
- Constants are processed faster than variables

The general form to declare a constant is as follows :

Const constantname [As type] =value

The *As type* part of the declaration is optional. If you omit it, the constant's type is determined by the value you assign to it. Constants also have a scope and can be Public or Private.

e.g. *Public Const pi as Double =3.14159265358979*

Visual Basic uses constants extensively to define the various arguments of its methods and the settings of the various control properties.

□ □ □

Chapter-3

Array Declaration

Q.1 How Arrays are declared and initialized in Visual Basic?

OR

Explain different types of Arrays?

Ans.: A standard structure for storing data in any programming language is an array. Whereas individual variables can hold single entities, such as a number, date, or string, arrays can hold sets of related data. An array has a name, as does a variable, and the values stored in it can be accessed by an index.

Declaring Arrays : Single Dimensional Array—Used to store long sequences of one dimensional data. Unlike simple variables, arrays must be declared with the Dim statement followed by the name of the array and the maximum number of elements it can hold in parantheses,

e.g. *Dim Sal(15)*

Multidimensional Array—To store the data in a tabular or matrix form, we need two dimensional array and to store sequences of multidimensional data we need multidimensional array.

In two dimensional array, first index represent the row and the second represents the column.

e.g. *Dim Board(5,5) As Integer*

Dynamic Arrays : Sometimes you may not know how large to make an array. Instead of making it large enough to hold the maximum number of

data, you can declare a dynamic array. The size of a dynamic array can vary during the course of the program.

To create a dynamic array, declare it as usual with the *Dim* statement, but don't specify its dimensions :

Dim DynArray()

Later in the program, when you know how many elements you want to store in the array, use the *ReDim* statement to redimension the array, this time with its actual size.

ReDim DynArr(UserCount)

Here UserCount is a user-entered value.

The *ReDim* statement can only appear in a procedure.

□ □ □

Chapter-4

Procedural Programming

Q.1 Explain Objects and Modules? What are the types of Modules in VB?

Ans.: Objects : Objects are the real world entities. People, companies, employees, fan and ledger entries are all types of objects. In object-oriented terms, the word object is used to describe one specific thing like a car. Objects have an identity and this identity is defined with properties. A car has its name, model, cost and color.

Objects also do things. E.g. the car accelerates, races etc. The things an object can do are called its behaviours.

Objects can be anything that exists in real world. They can be conceptual things, such as engineering process or payroll. These conceptual things are not tangible but are conceptual. The same object-oriented concepts apply regardless of whether the object is based on the real world, on a concept, or on the implementation.

Modules: Modules are collection of code and data that function something like objects in objects oriented programming, but without defining OOP characteristics like inheritance, polymorphism etc. The concept behind modules is to enclose procedures and data in a way that hides them from the rest of the program.

The two main types of procedures used in VB :

Event and General : In VB event procedures are invoked in response to keyboard, mouse or system action. Your code can also explicitly invoke event procedures. The maximum number of events a control can have is fixed. Event procedures are stored in a form module and are private by

default. A general procedure is not executed unless explicitly invoked. You can create a general procedure either by choosing the procedure from the insert menu or by typing the procedure heading Sub followed by the procedure name on a blank line.

Event and general procedures are further classified as : **Sub & Function**.

Q.2 What are Procedures?

OR

What are Subroutines and Functions?

Ans.: The code you write won't be a monolithic listing. It will be made up of small segments called procedures and you will work on one procedure at a time. This is called modularized approach of programming. It permeates the Visual Basic language even the longest applications are written by breaking them into small, well defined tasks. Each task is performed by a separate procedure that is written and tested separately from the others. Procedures are useful for implementing repeated tasks.

The two types of procedures are subroutines and functions—the building blocks of your application.

Subroutines : A subroutine is a block of statements that carries out a well-defined task. The subroutine begins with Sub and name of subroutine and its execution stops when EndSub statement is reached, and control returns to the calling program. It is possible to exit a Subroutine prematurely with the Exit statement.

Functions : A function is similar to a subroutine, but a function returns a result. Subroutines perform a task and don't report anything to the calling program; functions commonly carry out calculations and report the result. The statements that make up a function are placed in a pair of Function/End Function statements. Because the function must report the result to the calling program, it must have a return type.

□ □ □

Chapter-5

Branching and Looping Constructs

Q.1 What are the Branching Statements in Visual Basic?

OR

What are the different conditional constructs in Visual Basic?

Ans.: An application needs a built-in capability to test conditions and take a different course of action depending on the outcome of the test. Visual Basic provides three control flow, or decision, structures :

If……Then : The If…Then structure tests the condition specified, and if it's True, executes the statement(s) that follow. The If structure can have a single-line or a multiple-line syntax.

The general form is : *If condition Then statement*

If……Then…….Else : A variation of the If……Then statement is the If……Then……Else statement, which executes one block of statements if the condition is True and another if the condition is False. The syntax of the If… Then….Else statement is as follows:

If condition Then

statement block-1

Else

statement block-2

End If

Select Case : The Select Case structure compares one expression to different values. The advantage of the Select Case statement over multiple If…..Then…..Else statements is that it makes the code easier to read and maintain.

The Select Case structure tests a single expression, which is evaluated once at the top of the structure. The result of the test is then compared with several values, and if it matches one of them, the corresponding block of statements is executed.

The syntax is :

Select Case expression

Case value1

Statement block-1

Case value2

Statement block-2

.

.

.

Case Else

Statement block

End Select

Q.2 Explain different Looping Constructs in VB?

OR

What are the different types of Do……Loop?

Ans.: Loop statements allow you to execute one or more lines of code repetitively. Visual Basic supports the following loop statements:

Do……Loop : The Do……Loop executes a block of statements for as long as a condition is True. Visual Basic evaluates an expression, and if it's True, the statements are executed. If the expression is False, the program continues and the statement following the loop is executed.

There are two variations of the Do……Loop statement. A loop can be executed either while the condition is True or until the condition becomes True. These two variations use the keywords While and Until to specify how long the statements are executed. To execute a block of statements while a condition is true, use the following syntax :

Do While condition

statement block

Loop

To execute a block of statements until the condition becomes True, use the following syntax :

Do Until condition

statement block

Loop

Another variation of the Do loop executes the statements first and evaluates the condition after each execution. This Do…Loop has the following syntax :

Do

statements

Loop While condition

or

Do

statements

Loop Until condition

For…..Next : The For…….Next loop is one of the oldest loop structures in programming languages. Unlike the Do….loop, the For …..Next loop

requires that you know how many times the statements in the loop will be executed. The For…Next loop uses a variable(it's called the loop's counter) that increases or decreases in value during each repetition of the loop. The For…..Next loop has the following syntax :

For counter=start to end[Step increment]

statements

Next[counter]

The keywords in the square brackets are optional. The arguments counter, start, end and increment are all numeric. The loop is executed as many times as required for the counter to reach the end value.

While ……..Wend : The while……wend loop executes a block of statements while a condition is True. The while…..wend loop has the following syntax :

While condition

statement - block

Wend

If condition is true, all statements are executed and when the Wend statement is reached, control is returned to the While statement which evaluates condition again. If condition is still True, the process is repeated. If condition is False, the program resumes with the statement following the Wend statement.

Q.3 What is the use of 'Exit' statement?

Ans.: The Exit statement allows you to exit prematurely from a block of statements in a control structure from a loop, or even from a procedure. Suppose you have a For…….Next loop that calculates the square root of negative numbers can't be calculated (the Sqr() function generates a runtime error,) you might want to halt the operation if the array contains a

invalid value. To exit the loop prematurely, use the Exit For statement as follows :

```
For i=0 to UBound(nArray())
If nArray(i)<0  then Exit For
nArray(i)=Sqr(nArray(i))
Next
```

If a negative element is found in this loop, the program exits the loop and continues with the statement following the Next statement.

There are similar Exit statements for the Do loop (Exit Do), as well as for functions and subroutines (Exit Function and Exit Subroutine).

□ □ □

Chapter-6

Forms

Q.1 What are Forms in Visual Basic?

OR

Explain how working of Forms is being implemented?

Ans.: In Visual Basic the form is the container for all the controls that make up the user interface. When a Visual Basic application is executing, each window it displays on the Desktop is a Form. Forms have a built-in functionality that is always available without any programming effort on your part. You can move a form around, resize it, and even cover it with other Forms. You do so with the mouse, with the keyboard, or through the Control menu.

Appearance of Forms : The main characteristic of a Form is the title bar on which the form's caption is displayed. On the left end of the title bar is the Control Menu icon. Clicking this icon opens the Control menu. On the right side of the title bar are three buttons: Minimize, Maximize and Close.

The Start-Up Form : A typical application has more than a single Form. When an application starts, the main Form is loaded. You can control which Form is initially loaded by setting the start-up object in the Project Properties window.

Project -> Project Properties

Loading, Showing, and Hiding Forms : The possible states of a Form are:

- **Not loaded :** The Form lives on a disk file and doesn't take up any resources.

- **Loaded but not Shown :** The Form is loaded into memory, takes up the required resources, and is ready to be displayed.
- **Loaded and Shown :** The Form is shown, and the user can interact with it.

To load and unload Forms, use the Load and Unload statements. The Load statement has the following syntax :

Load formName

and the Unload statement has this syntax:

Unload formName

To show a Form, you use the Show method. If the Form is loaded but invisible, the Show method brings the specified Form on top of every other window on the Desktop. If the Form isn't loaded, the Show method loads it and then displays it.

The Show method has the following syntax :

formName.Show mode

The optional argument *mode* determines whether the Form will be modal or modeless. It can have one of the following values :

0-Modeless (default)

1-Modal

Modeless Forms interact with the user, and they allow the user to switch to any other Form of the application.

If your application uses many Forms, you may want to hide some of them to make room on the Desktop for others. To hide Form, use the Form's Hide method whose syntax is :

Form.Hide

To hide a form from within its own code, use the statement :

Me.Hide

□ □ □

Chapter-7

Programming Methodologies and Controls available in VB

Q.1 What are Global and Public Variables?

Ans.: The simplest method for two Forms to communicate with each other is via global variables. These variables are declared in the application's Module with the Global or Public keyword. If the following declarations appear in a Module, then they can be accessed from any place in the application's code :

- Global Num as Integer
- Public Data as Double

A Form's code however can't access variables declared in another Form directly. It's also possible to access a variable declared in one Form from within another, as long as one of these Forms "exposes" the variable by declaring it as *Public*.

Q.2 What are Activate and Deactivate Events?

Ans.: When more than one Form is displayed, the user can switch from one to the other with the mouse or by pressing Alt+Tab. Each time a Form is activated, the Activate event takes place. The Forms application uses the Activate event to set the Form's caption to a message indicating that this is the current Form :

Private Sub Form_Activate()

Form2.Caption= "Form2 Activated"

End Sub

Likewise, when a Form is activated, the previously active Form receives the Deactivate event, which the application uses to change the Form's caption :

Private Sub Form_Deactivate()

Form2.Caption= "Form2 Inactive"

End Sub

Q.3 What are ActiveX Controls?

Ans.: This type of project is a feature of the professional edition. Active X control is used to develop own Active X controls. Active X control such as a Text Box or command button control is a basic element of the user interface.

An Active X control, like a built in control, is an object that place on a form to enable or enhance a user's interaction with an application. ActiveX controls usually have an OCX extension. Remote Data Service can use Active X controls to display recordest data on a client web page. The controls are of the followings types:

(i) Text Box

(ii) List Box

(iii) Combo Box

(iv) Scroll Bar

(v) Timer

(vi) Drive List Box

(vii) Status Bar

(viii) TreeVview

(ix) Check Box

(x) Image List

Q.4 What is a Text Box Control?

OR

What are the most important characteristics of a Text Box Control?

Ans.: The textbox is the primary mechanism for displaying and entering text and is one of the most common elements of the Windows user interface. The TextBox control is a small text editor that provides all the basic text-editing facilities: inserting and selecting text, scrolling the text if it doesn't fit in the control's area, and even exchanging text with other applications through the Clipboard.

Basic Properties : Some of the properties that determine the appearance and, to some degree, the functionality of the Text Box control which can be set through the Properties window are as follows :

Multi Line : This property determines whether the Text Box control will hold a single line or multiple lines of text. By default, the control holds a single line of text.

Scroll Bars : This property controls the attachment of scrollbars to the Text Box control if the text exceeds the control's dimensions. Single line textboxes can have a horizontal scroll bar so that the user can view any part of a long line of text. Multi-line textboxes can have a horizontal or a vertical scroll bar or both.

Max Length : This property determines the number of characters the TextBox control will accept. Its default value is zero, which means the text may be of length, up to the control's capacity limit.

Text : The most important property of the Text Box control is the Text property, which holds the control's text. This property is also available at

design time so that you can assign some initial text to the control. At runtime, use this property to extract the text entered by the user or to replace the existing text by assigning a new value to the Text property.

Password Char : Available at design time, the Password Char property turns the characters typed into any character you specify. If you don't want to display the actual characters typed by the user, use this property to define the character to appear in place of each character the user types.

Q.5 Explain the basic properties of List Box and Combo Box Controls?

Ans.: The List Box and Combo Box controls present lists of choices for the user to select. The List Box control occupies a user-specified amount of space on the Form and is populated with a list of items; the user can select one or more with the mouse. The Combo Box control also contains multiple items but occupies less space on the screen. The Combo Box control is an expandable List Box **:** the user can expand it to make a selection and retract it after the selection is made.

Basic Properties : The List Box and Combo Box controls provide a few properties that can be set only at design time. They determine the basic functionality of the control and can't be changed at runtime. Some of those properties are as follows :

Multi Select : This property determines how the user can select the list's items and must be set at design time. The Multi Select property's values determine whether the user can select multiple items and which method will be used for multiple selection.

Sorted : Items can be inserted by the application into a List Box or Combo Box control, but inserting them in the proper place and maintaining some sort of organization can be quite a task for the programmer. If you want the items to be always sorted, set the control's Sorted property to True.

Style : This property determines the appearance of the control. Its value can be 0 or 1.

Q.6 How can we manipulate the items in the List Box Control?

Ans.: To manipulate a List Box control from within your application, you should be able to :

- Add items to the list
- Remove items from the list
- Access individual items in the list

Add Item : To add items to the list, use the Add Item method. Its syntax is follows :

List1.AddItem item, index

The item parameter is the string to be added to the list, and index is its order. The index argument is optional; if you omit it, the string is appended to the bottom of the list.

Remove Item : To remove an item from the list, you must first find its position in the list, which you must then supply to the Remove Item method. The syntax of the method is as follows :

List1.RemoveItem index

The index parameter is the order of the item to be removed, and this time, it's not optional.

Clear : The Clear method removes all the items from the control. Its syntax is simple :

List1.Clear

List Count : This is the number of items in the List. The items in the list can be accessed with an Index value, which goes from 0 to ListCount-1.

List() : This is an array that holds the list's items. The element List(0) holds the first element of the list, List(1) holds the second item, and so on up to List(ListCount-1), which holds the last item.

List Index : This is the index of the selected item in the List. If multiple items are selected, List Index is the index of the most recently selected item.

Selected : This property is an array, similar to the List property, with elements that have a True or a False value, depending on the status of the corresponding list element.

Q.7 What are the File Controls that are used in Visual Basic?

Ans.: Three of the controls on the Toolbox let you access the computer's file system. They are the Drive List Box, Dir List Box, and File List Box controls which are the basic blocks for building dialog boxes that display the host computer's file system.

The file controls are described as follows :

Drive List Box : Displays the names of the drives within and connected to the PC. The basic property of this control is the Drive property, which sets the drive to be initially selected in the control or returns the user's selection.

Dir List Box : Displays the folders of the current drive. The basic property of this control is the Path property, which is the name of the folder whose sub-folders are displayed in the control.

File List Box : Displays the files of the current folder. That basic property of this control is also called Path, and it's the path name of the folder whose files are displayed.

The three File controls aren't tied to one another. If you place all three of them on a Form, you'll see the names of all the drives in the Drive List Box. Selecting a drive in the Drive List Box control, however, doesn't affect the contents of the Dir List Box.

To connect the File controls, you must assign the appropriate values to their basic properties. The following is the minimum code you must place in the Drive List Box control's Change event:

Private Sub Drive1.Change()

Dir1.Path=Drive1.Drive

End Sub

Q.8 What are the general Graphics Controls?

Ans.: One of the most interesting and fun parts of a programming language, is its graphics elements. In general, graphics fall into two major categories: vector and bitmap. Vector graphics are images generated by graphics commands such as the Line and Circle commands. Bitmap graphics are images that can be displayed on various controls and processed on a pixel-by-pixel basis.

You can place graphics on three controls :

- Form
- Picture Box
- Image Box

The main difference in these three controls is that the Image Box control is designed specifically for displaying images and not for creating new images or manipulating them. The other two controls provide drawing methods that let you design graphics at runtime.

The methods for loading graphics on the various controls are simpler than creating graphics from scratch. You can place graphics on controls at design time and runtime. To load a graphic on a control at design time, you assign its filename to the Picture property of the control in the Properties window. To load a graphic on a control at runtime, use the Load Picture method as follows :

Form1.Picture=LoadPicture(filename)

The filename variable is the name of the file containing the graphic.

The Image property, which is a pointer to a structure in memory where the bits of the image are stored.

Q.9 What is MSFlexGrid control?

Or

How MSFlexGrid is one of the most useful control of Visual Basic?

Ans.: One of the most impressive controls of Visual Basic is the MSFlexGrid control. The MSFlexGrid control provides all the basic functionality for

building spreadsheet applications. The MSFlexGrid control is an extremely useful tool for displaying information in a tabular form. You can place it on your Forms to present nicely organized data to the user.

Basic Properties : The address of a cell is given by its Row and Col properties, whose values start at 0. Row 0 is the fixed title row, and column 0 is the fixed title column.

To address the first non-fixed cell in the grid, you would use the following statements:

Grid.Row=1

Grid.Col=1

You can examine the contents with a statement like :

CellValue=Grid.Text

or set its contents with a statement like :

Grid.Text="Monday"

The simplest way to address a cell on the grid is by means of the TextMatrix property, which has the following syntax:

Grid.TextMatrix(row, col)

Another property is Fixed Cols, Fixed Rows properties. The most common value for these two properties is 1, which translates into one fixed row and one fixed column. The fixed row and column contain titles that can be assigned with any of the methods mentioned earlier or with the help of the Format String property.

The Format String property can be assigned a string variable that sets up the control's column width, alignments, and fixed row and column text. The Format String property is made up of segments separated by pipe characters (|).

The user can change the width of columns and height of rows at runtime by dragging the column and row separators with the mouse. To disable row and/or column resizing, set the Allow User Resizing property to one of the following values:

Constant	Value	Resizing
flexResizeNone	0	User can't resize the cells
flexResizeColumns	1	User can resize columns only
flexResizeRows	2	User can resize rows only
flexResizeBoth	3	User can resize columns and rows

You can easily add data processing capabilities to the application. Inserting formulas won't be easy, but you can calculate the average of a range of cells or normalize them. You can add functionality for displaying or entering data etc.

Q.9 What is a Multiple Document Interface?

OR

How we can create an MDI Application?

Ans.: The *Multiple Document Interface (MDI)* was designed to simplify the exchange of information among documents all under the same roof. With an MDI application, you can maintain multiple open windows, but not multiple copies of the application. Data exchange is easier when you can view and compare many documents simultaneously.

The main form, or MDI Form, is not duplicated, but its acts as a container for all other windows, and it's called the parent window. The windows in which the individual documents are displayed are called child windows. Child windows, however, exist independently of the parent window. Most of the popular text editors are MDI applications too.

Applications that implement an MDI interface can open multiple documents of the same type and use a common menu structure that applies to all open documents.

An MDI application must have at least two Forms, the parent Form and one or more child Forms. The parent may not contain any controls. The parent Form can, and usually does, have its own menu.

To create an MDI application, follow these steps :

- Start a new project and then choose Project-> Add MDI Form to add the parent Form.
- Set the Form's caption to MDI Window.
- Choose Project-> Add Form to add a regular Form.
- Make this Form the child Form by setting its MDIChild property to True. To denote that this is a child Form, set its Caption property to MDIChild Form.

Q.10 Explain Windows API?

Ans.: To do advanced programming, such as detecting mouse movements outside a Form, you need some additional functions. These functions are provided by the Win32 Application Programming Interface (API). All the Win32 API functions are available from within the Visual Basic environment. Many Windows applications use the API to some extent.

The Win32 API consists of functions, structures, and messages that can be accessed to build Windows 95/98 and Windows NT applications.

The four fundamental categories of the Windows API are :

- Windows Management (User)
- Graphics Device Interface (GDI)
- System Services (Kernel)
- Multimedia (MMSystem)

The functions in these categories are implemented as DLLs (Dynamic Link Libraries) and can be called from any language. A DLL is loaded at runtime and it doesn't need to be linked to your application. Actually, that's where the name comes from: the contents of a DLL are linked to your application dynamically at runtime, not at design time. Because the API functions are required for the proper operation of Windows itself, the DLLs are always available to your application.

Windows Management provides the essential functions to build and manage your applications. All the basic input and output of the system goes through this layer of the Win32 API, including mouse and keyboard input and all processing of messages sent to your application.

The Graphics Device Interface provides the functions you use to manage all supported graphical devices on your system, including the monitor and printer. In addition you can define fonts, pens, and brushes.

Systems Services provides functions to access the resources of the computer and the operating system.

Multimedia functions allow you to play waveform audio, MIDI music, and digital video.

□ □ □

Chapter-8

Active X Components and Data Objects

Q.1 Explain the following terms :

- **ActiveX Components**
- **ADO**

Ans.: ActiveX Components : An ActiveX component is a general term that encompasses three types of projects: ActiveX DLLs, ActiveX EXEs, and ActiveX controls.

ActiveX controls are integrated into the Visual Basic IDE and they have a visible interface. While code components provide functionality similar to that of ActiveX controls, they aren't as integrated with the development environment and they don't have a visible interface.

Another category of ActiveX components is the AcitveX document. ActiveX documents are applications that can be hosted in containers such as Internet Explorer and the Office Binder.

Code components are implemented in Visual Basic as Class Modules. A Class Module is a server, or an application that provides its services to client applications. When you create an object variable to access the properties and methods of a Class, you're actually invoking an executable file(DLL or EXE) which runs in the background and waits to be contacted. Every time you set ore read a property value or call a method, this

executable activates, performs some actions, and optionally returns some results to your application.

Servers cam be implemented as ActiveX EXE or ActiveX DLL components. The difference between the two lies in how the server is executed. An ActiveX DLL component is an in-process server. The DLL is loaded in the same address space as the executable that calls the server and runs on the same thread as the client.

An ActiveX EXE component is a n out-of-process server that runs as a separate process. When a client application creates an object provided by an EXE server for the first time, the server starts running as a separate process.

The benefit of DLLs is that they are faster. Out-of-process servers seem to be more efficient in resource allocation.

ADO (Active Data Objects) : Visual Basic supports several data access tools, with the Active Data Objects(ADO) being the most recent addition. Whereas the first Visual Basic data access tools (the Data Access objects) allowed programmers to access databases only, ADO can access all major databases and is Microsoft's foundation for a universal technology for accessing all types of data in all environments.

With ADO, your Visual Basic application sees three objects :

- A Connection object, which establishes a connection to the database, be it a local file or a remote SQL Server
- A Command object, which executes commands against the database
- A Record Set object, which holds the records retrieved from the database or the records to be updated on the database.

Using the Active Data Objects : To use the Active Data Objects in an application, you must add a reference to the ADO Object library in the References dialog box. Follow these steps :

(1) Open the Project menu.

(2) Select Project-> References to open the References dialog box.

(3) Check the Microsoft ActiveX Data Objects 2.0 Library option.

After the ADO Library is added to a project, you can examine the objects it exposes and their members with the Object Browser.

Establishing a Connection : To establish an explicit connection to a Data Source, declare a *Connection* variable with the statement:

Dim ADOConn as new ADODB.Connection

ADOConn.Open "DSN=AdvWorks;UID=xxx;PWD=xxx"

Q.2 Explain the following terms :

(a) ADO

(b) DAO

Ans. : ADO : VB supports several data access tools, with the Active Data Objects(ADO). ADO can access all maj0or databases and is Microsoft's foundation for a universal technology for accessing all types of data in all environments.

I In VB applications have three objects with ADO :

- A Connection object, which establishes a connection to the database, be it a local file or a remote SQL Server
- A Command object, which executes commands against the database
- A Record Set object, which holds the records retrieved from the database or the records to be updated on the database.

DAO : The Data Access Object is a structure of objects for accessing databases through code. All the functionality of the Data Control is also available to code, through the Data Access Object (DAO). The Data Control provides access to databases without any programming. One can set a few properties of the control and use regular controls such as textboxes to display the values of the fields in the database. This is the no-code approach to database programming, which is implemented in VB.

□ □ □

BACHELOR OF COMPUTER APPLICATIONS
(Part II) EXAMINATION
(Faculty of Science)
(Three – Year Scheme of 10+2+3 Pattern)
PAPER 215
OBJECT ORIENTED PROGRAMMING

OBJECTIVE PART-I

Year - 2011

Time allowed : One Hour ***Maximum Marks : 20***

The question paper contains 40 multiple choice questions with four choices and students will have to pick the correct one (each carrying ½ mark.)

1. Which of the following represents a hexadecimal number?
   (a) 570 (b) (hex)5
   (c) Ox9F (d) 0X5 ( )

2. What will be the result of expression 13 and 25 ?
   (a) 38 (b) 25
   (c) 9 (d) 12 ( )

3. We would like to make a member of a class visible in all subclasses regardless of what package they are in which one of the following keywords would achieve this?
   (a) Private
   (b) Protected
   (c) Public
   (d) Private and Protected ( )

4. a package is a collection of :
   (a) Classes
   (b) Interface
   (c) Editing tools

(d) Classes and interface ( )

5. Which of the following methods belong to the string class?
   (a) length ( ) (b) compare to ( )
   (c) equals ( ) (d) all of the above ( )

6. The methods wait ( ) notify ( ) are defined in:
   (a) java-lang . String (b) java-lang . Runnable
   (c) java-lang . Object (d) java-lang . Thread ( )

7. When we invoke repaint ( ) for a component, the AWT invokes the method ?
   (a) draw ( )
   (b) show ( )
   (c) update ( )
   (d) paint ( ) ( )

8. Which of the following methods can be used to remove a component from the display?
   (a) delete ( )
   (b) remove ( )
   (c) disappear ( )
   (d) hide ( ) ( )

9. A class variable is a variable that is declared inside a class as:
   (a) Final
   (b) Static
   (c) Abstract
   (d) Immutable ( )

10. The finalize method is declared with the specified?
   (a) Public
   (b) Private
   (c) Protected
   (d) Package ( )

11. A break statement can't be used inside a :
   (a) for statement
   (b) Switch statement
   (c) labeled block
   (d) None of the above ( )

12. Java compiler generated code in:
    (a) Assembly language (b) Machine language
    (c) P-code (d) Byte-code ( )

13. In flow charting……..symbol is used for decisions:
    (a) triangle (b) rectangle
    (c) oval (d) diamond ( )

14. In text Name. Taxt :
    (a) the property is text Name (b) the object is text
    (c) the property is text (d) the class is text ( )

15. Which of the following is a relational operators:
    (a) >
    (b) and
    (c) +
    (d) & ( )

16. With the logical operator ……both conditions must be for the entire condition to evaluate.
    (a) or
    (b) not
    (c) =
    (d) and ( )

17. You are designing a form and it will be necessary for the user to type in their name. You should use a ……for inputting the user's name:
    (a) text box (b) label
    (c) command button (d) check box ( )

18. When you declare a variable or a named constant, visual basic reserve an area of memory and assigns it a name called:
    (a) an identifier (b) an identity
    (c) a declaration (d) a dimension ( )

19. Which of the following is not a valid rule for identifier?
    (a) Names may contain underscores
    (b) Names may contain letters
    (c) Names may contain digits

(d) Names may contain spaces ( )

20. Choose the operations that can be performed on string objects:
(i) +
(ii) + =
(iii) -
(iv) %

(a) (i)
(b) (iv and (i)
(c) (i) and (ii)
(d) none of the above ( )

21. Which of the following methods can be used to change the size of a component?
(a) area ( ) and resize ( )
(b) setsize ( ) and resize ( )
(c) setsize ( )
(d) resize ( ) ( )

22. When we implement the Runnable interface, we must define the method ?
(a) start ( ) (b) init ( )
(c) run ( ) (d) runnable ( ) ( )

23. In the cod below, what data types the variable x can have ?
byte b1 = 5;
byte b2 = 10;
x = b1 * b2;
(i) byte
(ii) int
(iii) short
(iv) long
(v) float
(vi) double

(a) (i) and (iv)
(b) (i)
(c) (ii), (iv) (v) and (vi)
(d) none of the above ( )

24. Which exception is thrown by the read ( ) method of input stream class?

(a) Exception (b) File Not found Exception
(c) Read Exception (d) IO Exception ( )

25. With Javadoc, which of the following denotes a javadoc comment?
(a) // # (b) / *
(c) / ** (d) // ** ( )

26. Which one does not have a value of (string) method?
(a) Integer
(b) Boolean
(c) Character
(d) Long ( )

27. What will be the output of line 5?
(1) Choice C 1 = new choice ( );
(2) C1 add ("first');
(3) C1 . add item ("second");
(4) C1 add("Third");
(5) System. Out. Print In (C1 . get item count ( ));

(a) 1
(b) 2
(c) 3
(d) None of the above ( )

28. Which one of the following does not extend java. Awt. Component?
(a) Check box
(b) Canvas
(c) Check box group
(d) Label ( )

29. Which of the following wrapper classes can not take a "string" in constructor?
(a) Boolean
(b) Integer
(c) Character
(d) Byte ( )

30. What does the following line of code do ?
Tax field text = new textifield (10); :
(a) Creates text object that can hold 20 rows of text

(b) Create text object that can hold 10 columns of text
(c) Creates the object text and initializes it with the value 10
(d) The code is illegal ( )

31. Data Input is:
(a) An abstract class defined in java.io
(b) An interface that defines methods to read primitive data types
(c) A class we can use to read primitive data types
(d) An interface that defines methods to open files ( )

32. Which methods is used to set the text of a label object ?
(a) set text ( )
(b) set label ( )
(c) set text label ( )
(d) set label text ( ) ( )

33. All of the component classes and container classes are derived from……..class.
(a) Object (b) abstract
(c) private (d) none of the above ( )

34. Which of the following types of class members can be part of the internal part of a class?
(i) Public instance variables
(ii) Private instance variables
(iii) Public methods
(iv) Private methods

(a) (ii) and (iv)
(b) (ii)
(c) (iv
(d) (i) and (iii) ( )

35. What is the result of the expression?
(1 & 2) + (3 : 4) in base ten.
(a) 1
(b) 2
(c) 8
(d) 7 ( )

36. Full from of AWT:

(a) About Window toolkit
(b) Abstract Window Tool
(c) Abstract Window Toolkit
(d) About Window Tool ( )

37. Full form of ADO:
(a) Active X Database Object
(b) Active X Data Object
(c) Active X Database Oriented
(d) Active X Data oriented ( )

38. Which property of Combo boxes gets or sets the binding context for the control?
(a) Bind context
(b) Binding context
(c) Bind control
(d) Binding control ( )

39. Visual basic for application (VBA) is:
(a) Implementation of MS –event-driven programming
(b) Associated with Integrated Development Environment (IDE)
(c) Built into most MS-office application
(d) All of the above ( )

40. Can main (0 method be overloaded?
(a) No
(b) Yes
(c) Both (a) and (b)
(d) None of the above ( )

Answer Key

1. (c)	2. (d)	3. (a)	4. (c)	5. (a)	6. (a)	7. (b)	8. (a)	9. (b)	10. (b)
11. (b)	12. (d)	13. (d)	14. (c)	15. (a)	16. (b)	17. (b)	18. (d)	19. (d)	20. (d)
21. (d)	22. (a)	23. (c)	24. (d)	25. (c)	26. (c)	27. (b)	28. (d)	29. (a)	30. (d)
31. (b)	32. (b)	33. (b)	34. (b)	35. (a)	36. (a)	37. (d)	38. (d)	39. (a)	40. (a)

DESCRIPTIVE PART-II

Year- 2011

Time allowed : 2 Hours ***Maximum Marks : 30***

Attempt any four descriptive types of questions out of six. All questions carry 7½ marks each.

1. (a) What is a constructor? What are its special properties?
   (b) Describe the different stages in the life cycle of an applet. Distinguish between init ( ) and start ( ) methods.

2. What is oops? What is polymorphism? Does visual basic support oops concepts in totality? Explain.

3. (a) What is OLE ? Differentiate between linked object and embedded object.
   (b) Write a program using while loop, it reverses the digits of the given number.

4. (a) What are Activex and types of Activex components in VB?
   (b) What are the major differences between an interface and a class?

5. Write a program to find the number of and sum of all integers grater than 100 and less than 200 that are divisible by 7.

6. Write short notes on the following :
   (a) Wrapper classes?
   (b) Thread and exceptions
   (c) Event driven programming.
   (d) Object and classes.

———

OBJECTIVE PART-I

Year - 2010

Time allowed : One Hour ***Maximum Marks : 20***

The question paper contains 40 multiple choice questions with four choices and students will have to pick the correct *one (each carrying ½ marks.).*

1. Which of the following is not a primitive data type?
   (a) boolean
   (b) byte
   (c) string
   (d) double ( )

2. The finally block is executed:
   (a) Only when a checked exception is thrown
   (b) Only when an unchecked exception is thrown
   (c) Only when an exception is thrown
   (d) Irrespective of whether an exception is thrown or not ( )

3. The statement.
   system.out printIn ((double7/4)
   (a) 1.75
   (b) 1
   (c) 1.0
   (d) 2.0 ( )

4. Which of the following methods is used to get the error message for the exception that was thrown:
   (a) print message
   (b) extract message
   (c) throw message
   (d) get message ( )

5. Which of the following operators only integer operands?
   (a) % (b) ++ (post increment)
   (c) ++ (pre increment) (d) None of the above ( )

6. Exceptions that are expected to possibly occur are called:
   (a) checked exception

(b) unchecked exceptions
(c) runtime exception
(d) errors ( )

7. Garbage collector frees the programmer from worrying about:
(a) memory leaks
(b) dangling references
(c) creating new object
(d) recursion ( )

8. The correct sequence of try, catch and finally, is usually:
(a) try, catch, finally
(b) finally, try, catch
(c) catch, try, finally
(d) try, finally, catch ( )

9. Which of the following dosen't have a super class?
(a) system
(b) object
(c) long
(d) exception ( )

10. Automatic, conversion from primitive type to an object of the corresponding wrapper class is called:
(a) Coercing (b) casting
(c) boxing (d) widening ( )

11. A class variable is a variable that is declared inside a class as:
(a) final (b) static
(c) abstract (d) immutable ( )

12. Which of the following activities has the potential to throw a checked?
(a) Accessing an array
(b) Dividing a no. by another
(c) Opening a file
(d) All of the above ( )

13. Which of the following type of exceptions must be declared by a piece?

(a) Checked exception (b) Unchecked exception
(c) Runtime exception (d) All of the above ( )

14. The finalize methods is declared with the access specified:
(a) Public (b) Private
(c) Protected (d) Package ( )

15. Which of the following packages in imported by default?
(a) Java.lang
(b) Java.io
(c) Java.math
(d) Java.text ( )

16. Declaring a class without an access specifies:
(a) In invalid
(b) Makes it accessible only to other class in the same package
(c) defaults the access specifier to public
(d) defaults the access specifier to public ( )

17. Which of the following is not a keyword in java:
(a) Volatile (b) Pointer
(c) super (d) this ( )

18. Which of the following methods is invoked by the garbage collector:
(a) destructor (b) constructor
(c) finally (d) finalize ( )

19. A break statement can't be used inside a :
(a) for statement
(b) switch statement
(c) labelled block
(d) None of the above ( )

20. Java compiler generated code in:
(a) Assembly (b) Machine language
(c) P-code (d) Byte code ( )

21. What does the statement :
(a) All the classes in package Java.util
(b) All the methods in the class Java.utill

(c) All the package starting with Java.utill
(d) All of the above ( )

22. Which of the following repository is used by the runtime system to identify the correct exception handler when an exception is thrown?
(a) profile (b) call
(c) activation record (d) call history ( )

23. Which of the following repository is used by the runtime system to identify the correct exception handler when an exception is thrown?
(a) Profile stack (b) call stack
(c) activation record (d) call history ( )

24. Val is well known:
(a) real-time language
(b) object-oriented language
(c) command language
(d) data flow language ( )

25. Which of the following class of statement usually produce no object code when compiled?
(a) Assignment (b) Declaration
(c) Untreatable (d) Control ( )

26. Which of the following languages is case-sensitive?
(a) BASIC (b) FORTRAN
(c) C (d) None of the above ( )

27. Heap allocation is required for languages that:
(a) Support recursion
(b) Support dynamic data structure
(c) Use dynamic scope rules
(d) None of the above ( )

28. Binding (of an identifier to a value) can occur while:
(a) Writing a program (b) Compiling a program
(c) Executing a program (d) All of the above ( )

29. The reserve word DIM in VB is a .....
(a) declaration (b) size

(c) storage location (d) identifier ( )

30. You can use the set of instructions in another event if you.....the other event.
(a) declars the variables in (b) force a break in
(c) use option explicit in (d) call ( )

31. In VB, the prefix for naming a mene item is.....
(a) m (b) mnu
(c) men (d) men ( )

32. Which of the following is a relational operator in VB?
(a) and (b) >
(c) + (d) & ( )

33. Comman dialog boxes
(a) allow VB programs to translate the text on the screen into another language
(b) display the predefined windows dialog boxed for print, open save fonts and colors
(c) can only be used on forms with menus
(d) none of the above ( )

34. In txtName.taxt:
(a) the property is txtname
(b) the property is text
(c) the object is text
(d) the class is text ( )

35. While designing a form, we must use a .....for inputting the user's name:
(a) text box
(b) label
(c) command button
(d) check box ( )

36. In order to make a picture expand to fill an image control, you must:
(a) set the stretch property to true
(b) set the stretch property to false
(c) set the visible property to true
(d) set the caption property filled ( )

37. The statement to declare a local variables called index that will store who has, is.....
(a) Dim whillndex as while
(b) Dim Strindex as string
(c) Dim integer as Intendex
(d) Dim intindex as integer ( )

38. Which of the following is not a valid rule for identifiers?
(a) Names may contain underscores
(b) Name may contain letters
(c) Names may contain digits
(d) Names may contain spaces ( )

39. The facility which allows customers to choose one or more items in VB is.....
(a) check boxes (b) text boxes
(c) option buttons (d) name boxes ( )

40. In flow chart .....symbol is used for making decisions:
(a) diamond (b) triangle
(c) rectangle (d) oval ( )

Answer Key

1. (c)	2. (d)	3. (a)	4. (c)	5. (a)	6. (a)	7. (b)	8. (a)	9. (b)	10. (b)
11. (b)	12. (d)	13. (d)	14. (c)	15. (a)	16. (b)	17. (b)	18. (d)	19. (d)	20. (d)
21. (d)	22. (a)	23. (c)	24. (d)	25. (c)	26. (c)	27. (b)	28. (d)	29. (a)	30. (d)
31. (b)	32. (b)	33. (b)	34. (b)	35. (a)	36. (a)	37. (d)	38. (d)	39. (a)	40. (a)

DESCRIPTIVE PART-II

Year- 2010

Time allowed : 2 Hours *Maximum Marks : 30*

Attempt any four descriptive types of questions out of six. All questions carry 7½ marks each.

Q.1 (a) What is multiple inheritance? Explain with example.

(b) Define a class and object.

Q.2 (a) What is a form in VB? What are the components of I.D.E.?

(b) What is the use of list and combo box in VB?

Q.3 (a) What is AWT and SWINGS in Java language?

(b) What are procedures are function in VB? Give Examples.

Q.4 (a) What is a wrapper class? Write a program in Java to compute the root of a quadratic equation.

(b) What is OLE? What are the data files concept in VB?

Q.5 (a) What are different operators of Java? Write a note on methods overloading

(b) How dragging and dropping of multiple objects in done? What is nesting?

Q.6 (a) What are Active X controls? Write a note on Recursion.

(b) Write a note on control structures in VB. Mention the structure of a VB program.

OBJECTIVE PART-I

Year - 2009

Time allowed : One Hour ***Maximum Marks : 20***

The question paper contains 40 multiple choice questions with four choices and students will have to pick the correct *one (each carrying ½ marks.).*

1. The default value of char type variable is:
   (a) '\u0020'
   (b) '\u00ff'
   (c) " "
   (d) '\u0000' ( )

2. What will be the result of the expression 13 And 25?
   (a) 38
   (b) 25
   (c) 9
   (d) 12 ( )

3. Java Virtual Machine.....
   (a) Java variable module
   (b) Interpreter for byte code
   (c) Compiler for byte code
   (d) None of the above ( )

4. Which of the following is correct?
   (a) int a = 16, a > > 2 =4
   (b) int b = –8, b = 4
   (c) int a = 16, a >>> 2 =4
   (d) All of the above ( )

5. A package is a collection of:
   (a) Classes (b) Interfaces
   (c) Editing tools (d) Classes and interfaces ( )

6. The methods wait ( ) and notify ( ) are defined in:
   (a) java.lang.string (b) java.lang.Runnable
   (c) java.lang.Object (d) java.lang.thread Group ( )

7. Which of the following methods belong to the string class?
   (a) length ( )
   (b) compare to ( )
   (c) equals ( )
   (d) All of the above ( )

8. COM stands for:
   (a) Component object model
   (b) Control Oriented Model
   (c) Computer Object Model
   (d) None of the above ( )

9. Visual Basic is used to develop the application:
   (a) Real time
   (b) Graphic user interface
   (c) Menu Driven
   (d) All of the above ( )

10. The zero fill right shift bitwise operator is:
   (a) >>
   (b) >>>
   (c) >>>=
   (d) >>= ( )

11. What will be the output of the following program?

```
int m = 100;
int n = 400;
while (++m<– –n);
System out.println(m);
```

   (a) 100
   (b) 200
   (c) 250
   (d) 500 ( )

12. When the form is first placed in memory using load statement or show methods, the event triggered:
   (a) Load (b) Show
   (c) Activate (d) All of the above ( )

13. The set background ( ) methods is part of the class :

(a) Graphics (b) Applet
(c) Component (d) Object ( )

14. Which of the following methods can be used to remove a component from the display?
(a) delete ( ) (b) remove ( )
(c) disapear ( ) (d) hide ( ) ( )

15. When we invoke repaint ( ) for a component, the AWT invokes the methods:
(a) draw ( ) (b) show ( )
(c) disapear ( ) (d) hide ( ) ( )

16. The form caption appears in the :
(a) Status bar
(b) Menu bar
(c) Title bar
(d) None of the above ( )

17. Which object is used in additem ( ) methods to add new item?
(a) List box (b) Text box
(c) Combo box (d) Both (a) and (c) ( )

18. ADO stands for:
(a) Active Data Object (b) Active X data object
(c) Access data object (d) All of the above ( )

19. We can create object at:
(a) Design time
(b) Run time
(c) Both (a) and (b)
(d) None of the above ( )

20. Which of the following applet tag is legal to embed applet class named test into a web page?
(a) <applet class = test width = 200 height = 100> < /applet>
(b) <applet code = test.class width = 200 height = 100> </applet>
(c) <applet code = Test width = 200 height = 100 > </applet>
(d) </applet> <code = test.class width = 200 height = 100> </applet> ( )

21. Which will be the content of array variable table after executing the following code:
following code
for (int i = 0;i < 3 ; i ++)
for (int j = 0; j< 3;j++)
if (jk = = i)jtable [i][j] = 1;
else table [i][j]= 0 ;

(a) 0 0 0 (b) 1 0 0
0 0 0 1 1 0
0 0 0 1 1 1

(c) 0 0 1 (d) 1 0 0
0 1 0 0 1 0
1 0 0 0 0 1 ( )

22. Which of the following is not keyword?
(a) Extended (b) Implement
(c) Protected (d) Public ( )

23. Which of key event occurs when a key is presses?
(a) Key type
(b) key up
(c) Key enter
(d) Key press ( )

24. Which key event occurs when key is pressed?
(a) An integer expression (b) A boolean expression
(c) DHTML applications (d) Neither (a) nor (b) ( )

25. Database tools are added automatically to the toolbox when project is developed under:
(a) Standard EXE (b) Data project
(c) DHTML application (d) Database application ( )

26. A control array is:
(a) Group of similar elements
(b) Group of similar control
(c) Group of controls that have same name
(d) None of the above ( )

27. A binary operator is:
(a) Negation (–)
(b) Bitwise complement (˜)
(c) Not (!)
(d) All of the above ( )

28. Which type of inheritance can be accomplished in java using interfaces?
(a) Single inheritance
(b) Multilevel inheritance
(c) Multiple inheritance
(d) Hybrid inheritance ( )

29. Access specified keyword:
(a) Protected
(b) Private
(c) Public
(d) All of the above ( )

30. Consider the following statements:
int x = 10, y = 15;
x = ((x< y)? (x+y) : (y–x));
what will be the value of x after executing these statements?
(a) 10 (b) 25
(c) 15 (d) 5 ( )

31. Which of the following will produce a value of 10 if x = 9.7?
(a) floor (x)
(b) abs (x)
(c) round (x)
(d) None of the above ( )

32. The range of values for the long type data:
(a) -2^{31} to $2^{31}-1$
(b) -2^{64} to 2^{64}
(c) -2^{63} to $2^{63}-1$
(d) -2^{32} to $2^{32}-1$ ( )

33. ODBC stands for:
(a) Object Database Control

(b) Open Database Connectivity
(c) Open Database Connectivity
(d) Object Database Connectivity ( )

34. Which methods is used to add a new record to a record set?
(a) Update
(b) Insert
(c) Add new
(d) Append ( )

35. A visual basic application can have:
(a) Many MDI forms
(b) Only one MDI from
(c) Only one child form
(d) All of the above ( )

36. Which keyword ensures that the argument is passed by value?
(a) By value
(b) By val
(c) By Ref
(d) All of the above ( )

37. Which statement causes termination of a FOR loop?
(a) Exit
(b) Exit for
(c) Exit Do
(d) Exit Sub ( )

38. Which of the following statement is used to size or resize a dynamic array?
(a) Dim
(b) Redim
(c) Declare
(d) Array ( )

39. Simple is.....then.....else statement can be alternatively represented through.....function.
(a) iif ( )
(b) whatif( )
(c) ifelse ( )
(d) All of the above ( )

40. The process by which one object acquires the properties of another object is:
    (a) Polymorphism
    (b) Inheritance
    (c) In capsulation
    (d) All of the above ( )

Answer Key

1. (d)	2. (c)	3. (c)	4. (c)	5. (d)	6. (d	7. (d)	8. (a)	9. (c)	10. (b)
11. (c)	12.(d)	13. (c)	14. (d)	15. (c)	16. (b)	17. (d)	18. (b)	19. (c)	20. (b)
21. (d)	22. (a)	23. (d)	24. (c)	25. (b)	26. (c)	27. (d)	28. (c)	29. (d)	30. (b)
31. (c)	32. (c)	33. (c)	34. (c)	35. (b)	36. (b)	37. (b)	38. (b)	39. (a)	40. (a)

DESCRIPTIVE PART-II

Year- 2009

Time allowed : 2 Hours ***Maximum Marks : 30***

Attempt any four descriptive types of questions out of six. All questions carry 7½ marks each.

Q.1 (a) Explain the basic concepts of OOPs object oriented programming.

(b) Why in Java known as platform neutral language?

(c) What is ADO?

Q.2 (a) What is Java Applet? Explain the applet life cycles with diagram.

(b) What is constructor? What are its special properties?

(c) When to we declare a method or class Final?

Q.3 (a) Explain the difference between SDI and MDI?

(b) What is a sub procedure? Explain and differentiate with a function.

(c) What is an array using in VB?

Q.4 (a) Write a program in Java Language to find out the factorial of a natural number

(b) What is the function overloading? Explain the differentiate with overriding.

Q.5 (a) Write a program to print the following output using for loop in VB/Java?

(i)
```
1
2  2
3  3  3
4  4  4  4
```

(ii) *

* *

* * *

* * * *

(b) Write a program to convert the given temperature in Fahrenheit to Celsius using the following conversion formula:

C = F-32/1.8

and display the values for F = 50 and 60.3.5

Q.6 Write short notes on the following:

(a) Operator overloading

(b) Wrapper classes;

(c) Active X controls;

(d) Dynamic Link Library (DLL);

(e) Classes using in VB

OBJECTIVE PART- I

Year - 2008

Time allowed : One Hour **Maximum Marks : 20**

The question paper contains 40 multiple choice questions with four choices and student will have to pick the correct one (each carrying ½ marks.).

1. MSDN stand for:
   (a) Microsoft Developer Network Library
   (b) Microsoft Sources Domain Name
   (c) Microsoft Directory Name
   (d) None of the above ( )

2. ADO stand for:
   (a) Active Data Object
   (b) Access Data Object
   (c) Active X Data Object
   (d) None of the above ( )

3. Data base tools are added automatically to the toolbox when project is developed under:
   (a) Standard EXE (b) Data project
   (c) DHTML applicatios (d) Data base applications ( )

4. Twip is used to:
   (a) Indicating the size of the control
   (b) It is printer's measurement system
   (c) Both (a) and (b)
   (d) None of the above ( )

5. The form caption appears in the:
   (a) Status bar (b) Menu bar
   (c) Title bar (d) None of the above ( )

6. Which object is used in additem ( ) methods to add new item:
   (a) List box (b) Text box
   (c) Combo box (d) Both (a) and (c) ( )

7. A control array is:
   (a) groups of similar elements
   (b) group of similar control
   (c) group of controls that have same name
   (d) none of the above ( )

8. The extension of activex control file is:
   (a) .vbp (b) .ocx
   (c) .vbx (d) both (b) and (c) ( )

9. COM stand for :
   (a) Computer Object Model
   (b) Component Object Model
   (c) Control Oriented Model
   (d) None of the above ( )

10. RDO stand for:
   (a) Real Data Object
   (b) Radio Data Object
   (c) Remote Data Object
   (d) None of the above ( )

11. When the form is firs placed in memory using load statement or show methods, the event triggered:
   (a) Load
   (b) Show
   (c) Activate
   (d) None of the above ( )

12. Mouse down event takes place when:
   (a) Press the mouse button
   (b) Click the left mouse button
   (c) Mouse is moved over a control
   (d) All of the above ( )

13. Which window is used to display list of all forms and modules in the application:

(a) Project window (b) Form layout window
(c) Properties window (d) Main window ( )

14. Visual basic is used to develop the application:
(a) Real time;
(b) Graphic user interface
(c) Menu driven
(d) None of the above ( )

15. If variables are not implicitly or explicitly typed, they are assigned the type by default:
(a) Character (b) Integer
(c) Variant (d) None of the above ( )

16. Which key event occurs when a key is pressed:
(a) key type (b) key up
(c) key enter (d) key press ( )

17. Which of the following is a file control:
(a) Drive list box
(b) Dir List
(c) File list box
(d) All of the above ( )

18. We can create object at:
(a) Design time
(b) Run time
(c) Both (a) and (b)
(d) None of the above ( )

19. The save picture statement supports:
(a) BMP
(b) GIF
(c) JPG
(d) None of the above ( )

20. OLE stands for:
(a) Object linking and embedding
(b) Object layering and embedding
(c) Object linking and enclosing

(d) None of the above ( )

21. DLL stands for:
(a) Dynamic link library (b) Data link library
(c) Defined library list (d) None of the above ( )

22. The zero fill right shift bitwise operator is:
(a) >> (b) >>>
(c) >>>= (d) >>= ( )

23. Short integer data type have the length in java language is:
(a) 8 bits (b) 32 bits
(c) 16 bits (d) 64 bits ( )

24. Which of the not a loop keyword:
(a) Do (b) Break
(c) Switch (d) Continue ( )

25. Access specified keyword is:
(a) Protected
(b) Private
(c) Public
(d) All of the above ( )

26. A unary operator is:
(a) Negation (–) (b) Bitwise complement
(c) not (!) (d) All of the above ( )

27. Oak is:
(a) Name of the class
(b) Name of the keyword
(c) Name of the java language
(d) Name of the java compiler ( )

28. The type casting is:
(a) Convert one type of value to another type
(b) convert one type of value of same type
(c) Both (a) and (b)
(d) None of the above ( )

29. Which of the following will produce a value of 22 if x = 22.9.
    (a) ceil (x) (b) round (x)
    (c) abs (x) (d) floor (x) ( )

30. What will be the output of the following program:
```
int m = 100;
int n = 300;
while (++m<--n);
system.out.orint1n(m);
```
    (a) 100 (b) 200
    (c) 301 (d) 500 ( )

31. Which of the following methods can be used to draw the outline of a square:
    (a) drawline ( )
    (b) Draw Rect ( )
    (c) DRaw polygon
    (d) All of the above ( )

32. When we implement the run able interface, we must define the method:
    (a) start ( ) (b) init ( )
    (c) run ( ) (d) main ( ) ( )

33. Which of the following statement is not true:
    (a) No destructor in Java
    (b) No nested class in Java
    (c) Java has not templates
    (d) Java has scope resolution operator ( )

34. Which type of inheritance can the accomplished in Java using interfaces:
    (a) Single inheritance
    (b) Multilevel inheritance
    (c) Multiple inheritance
    (d) Hybrid inheritance ( )

35. A class file in java is:
    (a) Machine dependent
    (c) Used for the interpreter to read
    (a) Used for the interpreter to read ( )

36. Which method is executed first when an applet runs:

(a) destroy ( ) (b) init ( )
(c) main ( ) (d) stop ( ) ( )

37. The process by which one object acquires the properties of another object is:
(a) Polymorphism
(b) Inheritance
(c) Encapsulation
(d) All of the above ( )

38. Java is a general purpose object oriented programming language developed at:
(a) Novell (b) IBM
(c) Netscape (d) SUN ( )

39. Java Virtual machine (JVM) is:
(a) Java variable module
(b) Interpreter for byte code
(c) complier for byte code
(d) None of the above ( )

40. Which of the following class are available in the java.lang package:
(a) Math (b) String
(c) String buffer (d) All of the these ( )

Answer Key

1. (d)	2. (c)	3. (a)	4. (c)	5. (d)	6. (d)	7. (c)	8. (b)	9. (b)	10. (c)
11. (c)	12. (d)	13. (a)	14. (b)	15. (c)	16. (d)	17. (d)	18. (b)	19. (d)	20. (a)
21. (a)	22. (b)	23. (c)	24. (c)	25. (d)	26. (d)	27. (c)	28. (a)	29. (d)	30. (b)
31. (b)	32. c)	33. (d)	34. (c)	35. (d)	36. (b)	37. (b)	38. (b)	39. (d	40. (b)

DESCRIPTIVE PART - II

Year 2008

Time allowed : 2 Hours ***Maximum Marks : 30***

Attempt any four questions out of the six. All questions carry 7½ marks each.

Q.1 (a) Explain the basic concepts to object oriented programming with examples.

(b) Why Java is called platform independent Language?

Q.2 (a) What is type casting? How many types of types casting is supported by java language and why is it required in programming? Explain with suitable examples.

(b) Explain the prototyping and benefits of prototyping.

Q.3 (a) What is a vector? How is different from array? Explain with examples.

(b) What is thread? Explain the multiprocessing and multithreading with suitable examples.

Q.4 (a) What do you understand by data type? Explain with names and purpose of each data type available in Visual Basic.

(b) What is Combo Box? Explain the different types of Combo Box and differentiate between Combo Box and List box control with suitable examples.

Q.5 (a) What is drag drop operation? Explain and differentiate the source and target in drag and drop operation.

(b) Differentiate between a sub procedure and a function procedure with suitable examples.

Q.6 Write short notes on the following:

(i) Collection

(ii) MDI form

(iii) Control Array

(iv) Activex control

(v) Toolbar

OBJECTIVE PART- I

Year - 2007

Time allowed : One Hour ***Maximum Marks : 20***

The question paper contains 40 multiple choice questions with four choices and student will have to pick the correct one (each carrying ½ marks.).

1. What is the error in the following class definition?

```
Abstract class
{
        abstract sum (int x, int y) { }
}
```

(a) Class header is not defined properly
(b) Ne error
(c) Constructor is not defined
(d) Method is not defined properly ( )

2. We would like to made a member of a class visible in all subclasses regardless of what package they are in. Which one of the following keywords would achieve this?
(a) Private
(b) Protected
(c) Public
(d) Private Protected ( )

3. A package is a collection of:
(a) Interfaces
(b) Classes
(c) Editing tools
(d) Classes and interfaces ( )

4. The Method wail ( ) and notify ( ) are defined in:
(a) java.lang.object
(b) java.lang.string
(c) java.lang.thread
(d) java.lang.runable ( )

5. When we invoke repaint ( ) for a component, the AWT invokes the method:
(a) Update ( )

(b) Show ( )
(c) Draw ( )
(d) Paint ( ) ( )

6. The set background ( ) method is part of the class:
(a) Graphics
(b) Applet
(c) Component
(d) Container ( )

7. What does the following line of code do?
Texfield.text = new textfield(10);
(a) Creates text object that can hold 10 rows of text
(b) Creates text object that can hold 10 characters
(c) Creates the object text and initializes it with the value 10
(d) The code is illegal ( )

8. Which of the following methods can be used to remove a component from the display?
(a) Delete ( )
(b) Hide ( )
(c) Move ( )
(d) Remove ( ) ( )

9. With java doc, which of the following denotes a java doc comment?
(a) // #
(b) / *
(c) / **
(d) // ** ( )

10. Which javadoc tag is used to denote a comment for a method parameter?
(a) @ method
(b) @ value
(c) @ param
(d) @ parameter ( )

11. Which of the following windows is the central to the development of Visual Basic application:
(a) Project window
(b) Form window

(c) Properties window
(d) All of the above ( )

12. When the form is first referenced in any manner by program, the event triggered:
(a) Load
(b) Initialize
(c) Activate
(d) None of the above ( )

13. When the form receives focus from another form, the event triggered:
(a) Load (b) Activate
(c) Show (d) None of the above ( )

14. What is the sequence of events when a form is unloaded?
(a) Query, un Load, Unload, Terminate
(b) Unload, Query, Unload, Terminate
(c) Query, Unload, Terminate, Unlaod
(d) None of the above ( )

15. The array that can be multiple dimensions and the size it can very during the courses of program:
(a) Multidimensional array (b) Dynamic array
(c) One-dimensional array (d) All of the above ( )

16. The procedure that perform a task and don't report anyting to the calling program:
(a) Subroutine (b) Function
(c) Property (d) All of the above ( )

17. What is the output of the following code?
Mystring = "visual basic in fun"
Left string = Left (mystring, 3)
(a) Vis
(b) Fun
(c) Visual basic is
(d) Is ( )

18. Which command is used to remove an item from a menu array?
(a) Delete
(b) Remove item
(c) Drop

(d) Unload ( )

19. The type of statement is used to find out:
(a) Type of picture displayed (b) Type of object dropped
(c) Type of button clicked (d) All of the above ( )

20. What does the window list properly do?
(a) Maintain a list of all forms in you program
(b) Allows you to add menu items to any menu at run time
(c) Keeps a list of MDI child windows
(d) Works with any application ( )

21. The properties of menu item cannot be changed at run time:
(a) Name (b) Caption
(c) Checked (d) Enabled ( )

22. By default, the text box control can hold the text:
(a) Multiple lines (b) single line
(c) Password character (d) None of the above ( )

23. The attach scrolbar to the texbox; the property of toolbox should be set to:
(a) Multiline = True
(b) Scrollbar = True
(c) Singleline = False
(d) None of the above
( )

24. If the password property of textbox set to "*" then:
(a) "*" would be entered in place of character
(b) ASCII value of "*" would be entered
(c) "*" would be displayed in place of character
(d) None of the above ( )

25. Which statement is used to identify an external procedure to visual basic?
(a) Declare
(b) Dim
(c) Public
(d) All of the above ( )

26. You can use.....function to obtain the window's class name:
(a) Get class name ( )

(b) Get class ( )
(c) Gen window class name ( )
(d) None of the above ( )

27. Visual basic's default coordinate system uses a unit called:
(a) Twip Tiff
(b) Points
(c) Pixels
(d) Pixels ( )

28. What is ADO stands for:
(a) Active data object
(b) ActiveX data object
(c) Action data object
(d) Active data option ( )

29. The drop down box at the top of the properties window is the:
(a) Object box
(b) Toolbox
(c) List box
(d) All of the above ( )

30. Visual basic is a tool that allows you to develop application:
(a) Real time
(b) Graphical user interface
(c) Menu driven
(d) None of the above ( )

31. How do you print a form "forml 1 "?
(a) Form1.PrintForm
(b) Form1.Print
(c) Print.Form1
(d) PrintForm.Form1 ( )

32. The MDI form is not duplicated, but it acts as a container for all other windows, is called.....window:
(a) Main
(b) Child
(c) Parent
(d) All of the above ( )

33. Which of the following keyword is used to keep track to active window?
    (a) Active
    (b) Current
    (c) Me
    (d) All of the above ( )

34. What is the error in the following code?

```
class test
{
        abstract void display ( );
}
```

    (a) No error
    (b) Method display ( ) should be declared as static
    (c) Test class should be declared as abstract
    (d) Test class should be declared as public ( )

35. Which keyword can protect a class in a package from accessibility by the classes outside the package?
    (a) Private
    (b) Protected
    (c) Final
    (d) Don's use any keyword at all (make it default) ( )

36. Which of the following methods belongs to the string class?
    (a) Length ( )
    (b) Substring ( )
    (c) Compare to ( )
    (d) All of the above ( )

37. Data Input is:
    (a) An abstract class defined in Java.do
    (b) A class we can use to read primitive data types
    (c) An interface that defines methods to peon files
    (d) An interface that defines methods to read primitive data types ( )

38. Which are the valid ways to create DataInputStream streams?
    (a) New DataInputSTream
    (b) new DataInputStream ("in.dat","r");
    (c) new DataInputStream (new FileInputStream ("in.dat")

(d) new DataInputStream (new File ("in.dat")); ( )

39. Which exception is thrown by the read ( ) method of Input stream class?
    (a) Exception
    (b) FileNotFoundException
    (c) ReadException
    (d) IOException ( )

40. When we implement the Runable interface, we must define the method?
    (a) Start ( ) (b) Run ( )
    (c) Init ( ) (d) Resume ( ) ( )

Answer Key

1. (d)	2. (d)	3. (d)	4. (c)	5. (a)	6. (a)	7. (b)	8. (d)	9. (c)	10. (c)
11. (d)	12. (a)	13. (b)	14. (a)	15. (b)	16. (a)	17. (a)	18. (b)	19. (b)	20. (c)
21. (a)	22. (b)	23. (b)	24. (c)	25. (a)	26. (a)	27. (a)	28. (b)	29. (a)	30. (b)
31. (b)	32. (c)	33. (c)	34. (c)	35. (c)	36. (d)	37. (d)	38. (c)	39. (d)	40. (b)

DESCRIPTIVE PART – II

Year 2007

Time allowed : 2 Hours ***Maximum Marks : 30***

Attempt any four questions out of the six. All questions carry 7½ marks each.

Q.1 (a) Explain the need of "Object-oriented programming". How is it different from procedural approach?

(b) Define in short:

(i) Data Encapsulation (ii) Applets (iii) Reusability

Q.2 Compare and contrast overloading and overriding methods. Also explain the syntax of single inheritance in Java.

Q.3 What is exception? Give the various keywords available in Java for exception handling. Explain nested 'try' statement using a suitable example.

Q.4 Explain objects, Modules and procedures. What are the types of modules in VB explain with suitable examples?

Q.5 Write a program in VB for ' Addition and 'Subtraction' of 2 complex number. Draw GUI required explain of each control drawn.

Q.6 Write short notes on the following:

(i) Combo box (ii) Arrays (iii) Active X control (iv) Threads

———

OBJECTIVE PART- I

Year - 2006

Time allowed : One Hour ***Maximum Marks : 20***

The question paper contains 40 multiple choice questions with four choices and student will have to pick the correct one (each carrying ½ mark).

1. Which of teh following numeric constants is written incorrecly in VD?
   (a) +0
   (b) –0
   (c) –5280
   (d) +7104 ( )

2. Variable address contains "Rajasthan University Jaipur" What is teh output of Right (Address,3)?
   (a) Jaipur
   (b) Rajasthan
   (c) Pur
   (d) Raj ( )

3. What will be the value of x in the following VB code?
   x=0
   For i=1 to 10 step 2.
   x=x+i
   next i
   (a) 21
   (b) 25
   (c) 30
   (d) 28 ( )

4. In VB API stands for:
   (a) Application Programming Interface
   (b) Application Product Interface
   (c) Application Programming Internet
   (d) Application Procedure Interface ( )

5. The default extension of forms in VB is:
   (a) .FXX
   (b) .FRX

(c) .FMR
(d) .FRM ( )

6. Full form of MDI is:
(a) Maximum Document Interface
(b) Multiple Division Interface
(c) Multiple Document Interface
(d) Multiple Document Inheritance ( )

7. Which of the following is not a feature of object oriented programming?
(a) Encapsulation
(b) Polymorphism
(c) Inernation
(d) Inheritance ( )

8. VB is known is:
(a) User driven programming
(b) Programmer driven programming
(c) Object driven programming
(d) Event driven programming ( )

9. What type of control box ix used to display error message or advice the users:
(a) Three IP addresses
(b) Two IP address
(c) One IP address
(d) None of the above ( )

10. A Pop-up description box that appears when the user resets the mouse pointer over a control is:
(a) Control Tip (b) File tip
(c) Pop up tip (d) Tool tip ( )

11. Controls placed on forms in VB can be manipulated by using:
(a) Property window
(b) Control Window
(c) Event window
(d) None of the above ( )

12. DLL stands for:
(a) Dynamic local library

(b) Data link library
(c) Dynamic link library
(d) Data local library ( )

13. What is the extension of Form file?
(a) .FRX
(b) .FRB
(c) .FRM
(d) .FRA ( )

14. The default property of timer control is:
(a) False
(b) True
(c) Disabled
(d) Enabled ( )

15. Which of the following is not cursor type in ADO?
(a) Static
(b) Dynamic
(c) Forward
(d) Backward ( )

16. With the help of which locking type in ADO it is not possible to alter the data:
(a) Lock Read
(b) Lock Batch Optimistic
(c) Lock Pessimistic
(d) Lock Optimistic ( )

17. Using Hungarian notation conversions lables are prefixed by:
(a) Lbl
(b) Lst
(c) Lin
(d) Llb ( )

18. Which of the following Active X control is used to connect through database?
(a) ADODC
(b) ADO
(c) DAO
(d) ODBC ( )

19. Which of the following is the correct syntax of main method?
    (a) Public int static main (String args[ ])
    (b) Public static void main (String args[ ])
    (c) Public main static void (String args { })
    (d) Public Static Void Main (String args{ }) ( )

20. What is the width the integer datatype in Java?
    (a) 5
    (b) 4
    (c) 3
    (d) 2 ( )

21. Which of the following is a Java Keyword?
    (a) External
    (b) Internal
    (c) Private
    (d) Throw ( )

22. Which of the following is legal array declaration?
    (a) Int*x
    (b) Int x [5]
    (c) Int x={1,2,3}
    (d) None of the above ( )

23. What will be the output of the following code?

```
int i = 0
for (; i >0; i – –)
        {
        ;
        }
```

    (a) Cannot be compiled
    (b) Execute in Infinite Loop
    (c) Produce an error
    (d) None of the above ( )

24. Which of the following is a valid method declaration?
    (a) Void method 7 ( ) { }
    (b) Void method 1 { }
    (c) Method 7 (void) { }
    (d) None of the above ( )

25. Which of the following is true?
    (a) All class must define constructor
    (b) A constructor can declare return value
    (c) A constructor can declare return value
    (d) A constructor can access non-static member of the class ( )

26. A function, that is called automatically when an object is destroyed, is:
    (a) Function prototype (b) Destructor
    (c) Constructor (d) Instantiation ( )

27. A class that contain at least one pure virtual function is called:
    (a) Pure class
    (b) Abstract class
    (c) Base Class
    (d) Derived class ( )

28. The exception is processed using:
    (a) Perform ( )
    (b) Catch ( )
    (c) Try ( )
    (d) Throw ( ) ( )

29. By default Java uses the following methods of passing arguments:
    (a) Call-by-pointer
    (b) Call-by-reference
    (c) Call-by-=value
    (d) None of the above ( )

30. If we would like to prevent the class being further classes then we have to use the following keyword:
    (a) Static
    (b) Final
    (c) Inheritance
    (d) Extends ( )

31. Which of the following keyword is not used in static block?
    (a) Final (b) Implements
    (c) Throws (d) Super ( )

32. Class Members are by default:
(a) Static
(b) Public
(c) Protected
(d) Private ( )

33. Classes are useful because they:
(a) Are removed from memory when not in use
(b) Can closely model object in the real world
(c) Permit data to be hidden from other classes
(d) Bring together all aspects of an entity in one place ( )

34. The & & and operators:
(a) Compare two numeric values
(b) Combine two numeric values
(c) Compare two Boolean values
(d) Combine two Boolean values ( )

35. Overload functions:
(a) Make life simpler for programmer 3
(b) Are group of function with same name
(c) May fail unexpectedly due to stress
(d) All have same number due to stress ( )

36. Static block can only access:
(a) Public data
(b) Status data
(c) Protected data
(d) All of the above ( )

37. If a class contain abstract method then it is essential to declare the class as:
(a) Abstract class (b) Super Class
(c) Inherit Class (d) Public Class ( )

38. If a class contain abstract are created in the following sequence or code?
string A,B,C
A = "Rajasthan".
B=A;
C= A+B;
(a) 2 (b) 4

(c) 1 (d) 3 ( )

39. Which of the following statement is true?
    (a) Methods cannot be overridden to be more private
    (b) Static methods can not be overloaded
    (c) Private methods cannot be overloaded
    (d) An overloaded methods can not throw exceptions ( )

40. Which of the following is correct?

(1)
```
Import mypack .*;
class A
{.....
.....
.....}
```

(2)
```
Package mypack;
public class A
{.....
.....
.....}
```

(3)
```
Import.java.lang.*;
public package mypack;
class A
{.....
.....
.....}
```

(4)
```
Package mypack;
public import mypack.*:
class A
{.....
.....
.....}
```

(a) Only 1,2 are true
(b) Only 1,2,3 are true
(c) Only 1,2,4 are true
(d) All are true ( )

Answer Key

1. (b)	2. (c)	3. (b)	4. (a)	5. (d)	6. (c)	7. (c)	8. (c)	9. (c)	10. (d)
11. (a)	12. (b)	13. (c)	14. (c)	15. (d)	16. (a)	17. (a)	18. (a)	19. (b)	20.(b)
21. (d)	22. (d)	23. (d)	24. (d)	25. (c)	26. (b)	27. (a)	28. (c)	29. (d)	30. (b)
31. (d)	32. (b)	33. (d)	34. (c)	35. (b)	36. (b)	37. (a)	38. (c)	39. (c)	40. (a)

DESCRIPTIVE PART - II

Year 2006

Time allowed : 2 Hours ***Maximum Marks : 30***

Attempt any four questions out of the six. All questions carry 7½ marks each.

Q.1 Explain the following with examples:

(a) Class (b) Object (c) Constructors

(d) Polymorphism (e) Applets

Q.2 Explain the various looping control structures available in VB with examples:

Q.3 Write short notes on the following:

(a) Tool tip (b) Combo box (c) Input box

Q.4 Explain the differences between methods overloading and methods overriding giving proper examples.

Q.5 What is inheritance? How is it useful? Explain the various types of inheritance with examples .

Q.6 Describe the following:

(a) Threads (b) Arrays (c) Exception handling

Multiple Choice Questions

Set A

1. IDE is:
   (a) Independent Development Enterprise
   (b) A development environment for machine language
   (c) A software project management tool
   (d) An Integrated Development Environment for Visual Basic

Answer: D

2. Which of the following is not part of the IDE:
   (a) Code editor window
   (b) Properties window
   (c) Form layout window
   (d) General window

Answer: D

3. The application name always appears in the:
   (a) Properties window
   (b) Intermediate window
   (c) Code window
   (d) Title bar

Answer: D

4. The color of a button is:
   (a) One of its properties
   (b) Not updateable
   (c) Defined in the project
   (d) Defined in the Intermediate window

Answer: A

5. Code is:
   (a) Updateable in the form editor
   (b) Instructions
   (c) Seldom used
   (d) An object

Answer: B

6. Controls are:
   (a) Code

(b) Part of the menus
(c) Rules
(d) Objects

Answer: D

7. In the IDE, which of following is used to design the layout of an application?
   (a) Form Designer window
   (b) Project Explorer window
   (c) Context Menu
   (d) Form Layout window

Answer: A

8. The location of the form on the desktop during execution is determined by the:
   (a) Form Designer window
   (b) Project Explorer window
   (c) Context Menu
   (d) Form Layout window

Answer: D

9. The Object Browser:
   (a) Displays the command buttons and textboxes, etc.
   (b) Shows frequently used shortcuts as objects
   (c) Is a Context Menu
   (d) Displays the object libraries and their combinations of data and code

Answer: D

10. The location of the form on the desktop during execution is determined by the:
    (a) Form Designer window
    (b) Project Explorer window
    (c) Context Menu
    (d) Form Layout window

Answer: D

11. The first procedure-oriented language was:
    (a) FORTRAN
    (b) BASIC
    (c) COBOL
    (d) ADA

Answer: A

12.Event-driven languages are:

(a) FORTRAN based
(b) Are used to write procedural languages
(c) OOP
(d) Designed to make programming GUI easier

Answer: D

12. COBOL is:

(a) One of the oldest programming languages
(b) Still widely used
(c) Not suitable for business applications
(d) None of the above

Answer: D

13. Object Oriented languages:

(a) Are procedural languages
(b) Are task oriented
(c) Are based on actions happening to objects
(d) Are natural language techniques

Answer: C

14. Visual Basic projects are identified by a:

(a) ".vbp" suffix
(b) ".mak" suffix
(c) ".vbg" suffix
(d) All the above

Answer: D

15. Visual Basic forms are identified by a:

(a) ".frm" suffix
(b) ".mak" suffix
(c) ".for" suffix
(d) A special icon

Answer: A

16. To run an application in Visual Basic:

(a) Click on the start button (blue arrow)
(b) Use the File Menu
(c) Use the Project Menu to select Run
(d) None of the above

Answer: A

17. To exit Visual Basic:

(a) Click Alt-Q
(b) Use the File Menu to select Quit
(c) Use the Window Menu to select Exit
(d) Click on the diskette icon

Answer: A

18. The reference library of Visual Basic books is called:
(a) MSDN Library
(b) Help Library
(c) Contents
(d) Topic pane

Answer: A

19. The CancelButton property belongs to which object?
a.) Button
b.) Form
c.) Label
d.) TextBox
e.) Timer
Answer: *b*

20. A click event procedure stud for the label control can be created by:
a.) selecting the object and event from the code editor window's drop-down boxes.
b.) typing the code in the code editor window.
c.) by double clicking the control.
d.) Both a and b.
e.) All of the above.

Answer: *e*

Set B

1. Which language is not a true object-oriented programming language?
a.) VB.NET
b.) VB 6
c.) C++
d.) C#
e.) Java
Answer: *b*

2. A GUI:
   a.) uses buttons, menus, and icons.
   b.) should be easy for a user to manipulate.
   c.) stands for Graphic Use Interaction.
   d.) Both a and b.
   e.) All of the above.
   Answer: *d*

3. Visual Studio .NET provides which feature:
   a.) debugging.
   b.) application deployment.
   c.) syntax checking.
   d.) Both a and b.
   e.) All of the above.
   Answer: *e*

4. What does IDE stand for?
   a.) Integrated Development Environment
   b.) Integrated Design Environment
   c.) Interior Development Environment
   d.) Interior Design Environment
   e.) None of the above.
   Answer: *a*

5. Which type of project can a developer choose in the New Project dialog box?
   a.) Visual Basic Projects
   b.) Visual C# Projects
   c.) Visual C++ Projects
   d.) Both a and b.
   e.) All of the above.
   Answer: *e*

6. Which is not a main component of the Visual Studio IDE?
   a.) Solution Explorer
   b.) Tool Box
   c.) Start Menu

d.) Designer Window
e.) Properties Window
Answer: *c*

7. Which does the solution explorer **not** display?
a.) Form Properties
b.) Reference Folder
c.) Form File
d.) Assemble File
e.) All are part of the solution explorer.
Answer: *a*

8. Which is true about the name and text property of a control?
a.) They are the same when the control is first created.
b.) The text property changes to match any changes in the name property.
c.) The name property changes to match any changes in the text property.
d.) They are never the same unless the programmer makes it that way.
e.) They are not allowed to be the same and an error will occur if they are.
Answer: *a*

9. For which task does the IDE provide multiple ways to accomplish the task?
a.) Putting a control on the form
b.) Running the program
c.) Activating the property window for a control
d.) Both a and b.
e.) All of the above.
Answer: *e*

10. Which are the standard prefixes for the Button and Combo box controls respectively?
a.) btn and chb
b.) btn and cbo
c.) bto and chb
d.) bto and cbo
e.) cmd and cbo
Answer: *b*

11. Which are the standard prefixes for the text box and label controls respectively?
    a.) tex and lbl
    b.) tex and lab
    c.) txb and lbl
    d.) txb and lab
    e.) txt and lab
    Answer: *c*

12. Which task is accomplished in the Code editor?
    a.) Adding forms to the project
    b.) Adding controls to the form
    c.) Adding event procedures to the form
    d.) Both a and b.
    e.) All of the above.
    Answer: *c*

13. Which is not a feature of a GUI that makes learning a program easy for users?
    a.) Online help
    b.) WYSIWYG formatting
    c.) Dialog boxes
    d.) Detailed key strokes and commands
    e.) Icons
    Answer: *d*

14. An object is composed of:
    a.) properties.
    b.) methods.
    c.) events.
    d.) Both a and b.
    e.) All of the above.
    Answer: *e*

15. Which statement about objects is true?
    a.) One object is used to create one class.
    b.) One class is used to create one object.
    c.) One object can create many classes.

d.) One class can create many objects.
e.) There is no relationship between objects and classes.
Answer: *d*

16. Which is **not** true about forms and controls in Visual Basic?
a.) They are pre-built.
b.) They are graphical objects.
c.) New versions of the classes must be created with each project.
d.) Buttons can be created with the drag and drop method.
e.) All of the above are true.
Answer: *c*

17. Which is an example of Visual Basic Objects?
a.) Control objects
b.) ASP.NET
c.) ADO.NET
d.) Both a and b.
e.) All of the above.
Answer: *e*

18. The .Net class library:
a.) contains over 25,000 classes.
b.) uses namespaces to manage all of the classes.
c.) has the System.Form namespace for classes used in Windows-based application.
d.) Both a and b.
e.) All of the above.
Answer: *d*

19. Which is not a property of the Common control class?
a.) Show
b.) BackColor
c.) Font
d.) ForeColor
e.) Name
Answer: *a*

20. Which property determines whether a control is displayed to the user?

a.) Hide
b.) Show
c.) Visible
d.) Enabled
e.) Cursor

Answer: *c*

Set C

1. The Button control can be activated:
   a.) programmatically through the click event.
   b.) by clicking the button with the mouse.
   c.) with the form's DefaultButton property.
   d.) Both a and b.
   e.) All of the above.
   Answer: *d*

2. The CancelButton property belongs to which object?
   a.) Button
   b.) Form
   c.) Label
   d.) TextBox
   e.) Timer
   Answer: *b*

3. A click event procedure stud for the label control can be created by:
   a.) selecting the object and event from the code editor window's drop-down boxes.
   b.) typing the code in the code editor window.
   c.) by double clicking the control.
   d.) Both a and b.
   e.) All of the above.
   Answer: *e*

4. In event-driven programming an event is generated by:
   a.) the system.
   b.) a user's action.
   c.) the program itself.
   d.) Both a and b.

e.) All of the above.
Answer: *e*

5. Which is not a common control event?
a.) Click
b.) SingleClick
c.) DoubleClick
d.) MouseMove
e.) MouseDown
Answer: *b*

6. The Tick event is found only in which object?
a.) Form
b.) Button
c.) TextBox
d.) Label
e.) Timer
Answer: *e*

7. The Activated event is found only in which object?
a.) Form
b.) Button
c.) TextBox
d.) Label
e.) Timer
Answer: *a*

8. The Rnd statement will generate a(n):
a.) decimal value between 0.01 and 1.00.
b.) integer value between 0.01 and 1.00.
c.) decimal value between 0.0 and 1.0.
d.) integer value between 0.0 and 1.0.
e.) decimal value between 0.0 and up to 1.0, but not including 1.0.
Answer: *e*

9. The analysis phase of software development involves:
a.) collecting the requirements about what the program will accomplish.

b.) creating a detailed plan on how the program will accomplish the requirements.
c.) writing the software with a program such as VB.NET.
d.) Both a and b.
e.) All of the above.
Answer: *a*

10. Which phase of project development typically costs the most?
a.) Analysis
b.) Design
c.) Implementation
d.) Maintenance
e.) Documentation
Answer: *d*

11. Which is not an integer data type?
a.) Single
b.) Byte
c.) Short
d.) Integer
e.) Long
Answer: *a*

12. Which is a numeric data type?
a.) Floating point
b.) Integer
c.) Boolean
d.) Both a and b.
e.) All of the above.
Answer: *d*

13. Which sequence of char data types is listed from lowest to highest?
a.) a, A, z, Z
b.) a, z, A, Z
c.) A, a, Z, z
d.) A, Z, a, z
e.) z, a, Z, A
Answer: *d*

14. The Date data type does not hold which type of information.
a.) Seconds
b.) Hours
c.) Days
d.) Months
e.) Quarters
Answer: *e*

15. The Boolean data type:
a.) is unsigned.
b.) has two states.
c.) is displayed by the program as yes or no.
d.) Both a and b.
e.) All of the above.
Answer: *d*

16. Which is a valid statement for declaring a variable?
a.) Const Form As Integer
b.) Const myForm As Integer
c.) Dim Form As Integer
d.) Dim myForm As Integer
e.) All of the above.
Answer: *d*

17. VB.Net identifiers:
a.) are case sensitive.
b.) can begin with an underscore.
c.) can begin with a number.
d.) Both a and b.
e.) All of the above.
Answer: *b*

18. The name of a constant:
a.) must both begin with a letter and be all upper case.
b.) does not have to begin with a letter but must be all upper case.
c.) must begin with a letter but can be upper or lower case.
d.) does not have to begin with a letter and be either upper or lower case.
e.) None of the above.

Answer: *d*

19. The proper operator precedence, from first to last, is:
    a.) logical, comparison, and arithmetic.
    b.) arithmetic, comparison, and logical.
    c.) arithmetic, logical, and comparison.
    d.) comparison, arithmetic, and logical.
    e.) logical, arithmetic, comparison.
    Answer: *b*

20. With A = False and B = True, which statement evaluates as True?
    a.) A AND A
    b.) A AND B
    c.) B AND A
    d.) B AND B
    e.) None are true.
    Answer: *d*

21. With A = False and B = True, which statement evaluates as False?
    a.) A OR A
    b.) A OR B
    c.) B OR A
    d.) B OR B
    e.) None are true.
    Answer: *a*

22. Which operator is evaluated first?
    a.) NOT
    b.) AND
    c.) XOR
    d.) OR
    e.) They are always evaluated left-to-right.
    Answer: *a*

23. The left side of an assignment statement will hold:
    a.) a variable.
    b.) an object property.
    c.) an expression.

d.) Both a and b.
e.) All of the above.
Answer: *d*

24. The right side of an assignment statement will hold:
a.) a variable.
b.) an object property.
c.) an expression.
d.) Both a and b.
e.) All of the above.
Answer: *c*

25. Which function will return the monthly payments of a loan?
a.) Pay (Rate, PV, Nper)
b.) Pmt (Rate, Nper, PV)
c.) FV (Rate, Nper, Pmt)
d.) FV (Rate, Nper, PV)
e.) None of the above.
Answer: *b*

26. Which function returns the numbers represented in the string "$56.7"?
a.) Abs
b.) CDbl
c.) Int
d.) Rnd
e.) Val
Answer: *b*

27. What will the function Val ($165.30) return?
a.) 0
b.) 165
c.) 165.30
d.) $165.30
e.) An error
Answer: *a*

28. Which function displays a pop-up window?
a.) MsgBox

b.) InputBox
c.) TextBox
d.) Both a and b.
e.) All of the above.
Answer: *d*

29. Which is true about the prompt argument?
a.) It can be made of multiple values concatenated into one string.
b.) It can include the vbCrLf constant.
c.) It can include the ampersand symbol to concatenate strings.
d.) Both a and b.
e.) All of the above.
Answer: *e*

30. In order to process a number typed in a TextBox the programmer must:
a.) use the Val function to convert the Text value.
b.) use the CDbl function to convert the Text value.
c.) use the IsNumeric function to convert the Text value.
d.) Both a and b.
e.) All of the above.
Answer: *d*

31. Which TextBox method does **not** use the clipboard?
a.) Clear
b.) Copy
c.) Cut
d.) Paste
e.) All of these methods use the clipboard.
Answer: *a*

32. Which TextBox property should always be changed first?
a.) AcceptsReturn
b.) BorderStyle
c.) Font
d.) Name
e.) Text
Answer: *d*

33. Which is not a valid value for the ListBox SectionMode Property?
a.) None
b.) One
c.) MultiSimple
d.) MultiExtended
e.) All of the above.
Answer: *e*

34. Setting the SelectedIndex property of a ListBox to -1 will:
a.) cause an error.
b.) cannot be done.
c.) de-select any selected item.
d.) Both a and b.
e.) All of the above.
Answer: *c*

35. Which method of a ListBox will remove just one item at a time?
a.) Items.RemoveAt
b.) Item.RemoveAt
c.) Items.ClearAt
d.) Item.ClearAt
e.) Items.Clear
Answer: *a*

36. The Items property of a ComboBox:
a.) is a collection of items.
b.) is the same as the Items property of a ListBox.
c.) contains methods and properties.
d.) Both a and b.
e.) All of the above.
Answer: *e*

37. Which value for the ComboBox DropDownStyle property allows a user to type in data?
a.) DropDown
b.) DropDownSimple
c.) DropDownList
d.) Both a and b.

e.) All of the above.
Answer: *a*

38. Which two controls combined to form the ComboBox control?
a.) ListBox and TextBox
b.) ListBox and InputBox
c.) ListBox and MsgBox
d.) Label and TextBox
e.) Label and InputBox
Answer: *a*

39. When a condition in an If… Then statements tests true:
a.) the next Else statement is activated.
b.) the next If statement is activated.
c.) the next Then statement is activated.
d.) the End If statement is activated.
e.) a condition can never test true.
Answer: *c*

40. The End If statement is required:
a.) in all If… Then statements.
b.) in all Multi-line statements with Else.
c.) in Single Line statements.
d.) Both a and b.
e.) All of the above.
Answer: *b*

GLOSSARY

A

abstract class: A class primarily intended to define an instance, but can not be instantiated without additional methods.

abstract data type: An abstraction that describes a set of items in terms of a hidden data structure and operations on that structure.

abstraction: A mental facility that permits one to view problems with varying degrees of detail depending on the current context of the problem.

accessor: A public member subprogram that provides query access to a private data member.

actor: An object that initiates behavior in other objects, but cannot be acted upon itself.

agent: An object that can both initiate behavior in other objects, as well as be operated upon by other objects.

ADT: Abstract data type.

AKO: A Kind Of. The inheritance relationship between classes and their super classes.

allocatable array: A named array having the ability to dynamically obtain memory. Only when space has been allocated for it does it have a shape and may it be referenced or defined.

argument: A value, variable, or expression that provides input to a subprogram.

array: An ordered collection that is indexed.

array constructor: A means of creating a part of an array by a single statement.

array overflow: An attempt to access an array element with a subscript outside the array size bounds.

array pointer: A pointer whose target is an array, or an array section.

array section: A subobject that is an array and is not a defined type component.

assertion: A programming means to cope with errors and exceptions.

assignment operator: The equal symbol, "=", which may be overloaded by a user.

assignment statement: A statement of the form "variable = expression".

association: Host association, name association, pointer association, or storage association.

attribute: A property of a variable that may be specified in a type declaration statement.
automatic array: An explicit-shape array in a procedure, which is not a dummy argument, some or all of whose bounds are provided when the procedure is invoked.

B

base class: A previously defined class whose public members can be inherited by another class. (Alsocalled a super class.)
behavior sharing: A form of polymorphism, when multiple entities have the same generic interface.
This is achieved by inheritance or operator overloading.
binary operator: An operator that takes two operands.
bintree: A tree structure where each node has two child nodes.
browser: A tool to find all occurrences of a variable, object, or component in a source code.

C

call-by-reference: A language mechanism that supplies an argument to a procedure by passing the address of the argument rather than its value. If it is modified, the new value will also take effect outside of the procedure.
call-by-value: A language mechanism that supplies an argument to a procedure by passing a copy of its data value. If it is modified, the new value will not take effect outside of the procedure that modifies it.
class: An abstraction of an object that specifies the static and behavioral characteristics of it, including their public and private nature. A class is an ADT with a constructor template from which object instances are created.
class attribute: An attribute whose value is common to a class of objects rather than a value peculiar to each instance of the class.
class descriptor: An object representing a class, containing a list of its attributes and methods as well as the values of any class attributes.
class diagram: A diagram depicting classes, their internal structure and operations, and the fixed relationships between them.
class inheritance: Defining a new derived class in terms of one or more base classes.
client: A software component that users services from another supplier class.
concrete class: A class having no abstract operations and can be instantiated.
compiler: Software that translates a high-level language into machine language.
component: A data member of a defined type within a class declaration

constructor: An operation, by a class member function, that initializes a newly created instance of aclass. (See default and intrinsic constructor.)
constructor operations: Methods which create and initialize the state of an object.
container class: A class whose instances are container objects. Examples include sets, arrays, and stacks.
container object: An object that stores a collection of other objects and provides operations to access or iterate over them.
control variable: The variable which controls the number of loop executions.
D
data abstraction: The ability to create new data types, together with associated operators, and to hide the internal structure and operations from the user, thus allowing the new data type to be used in a fashion analogous to intrinsic data types.
data hiding: The concept that some variables and/or operations in a module may not be accessible to a user of that module; a key element of data abstraction.
data member: A public data attribute, or instance variable, in a class declaration.
data type: A named category of data that is characterized by a set of values. together with a way to denote these values and a collection of operations that interpret and manipulate the values. For an intrinsic type, the set of data values depends on the values of the type parameters.
deallocation statement: A statement which releases dynamic memory that has been previously allocated to an allocatable array or a pointer.
debugger software: A program that allows one to execute a program in segments up to selected breakpoints, and to observe the program variables.
debugging: The process of detecting, locating, and correcting errors in software.
declaration statement: A statement which specifies the type and, optionally, attributes of one or more variables or constants.
default constructor: A class member function with no arguments that assigns default initial values to all data members in a newly created instance of a class.
defined operator: An operator that is not an intrinsic operator and is defined by a subprogram that is associated with a generic identifier.
deque: A container that supports inserts or removals from either end of a queue.
dereferencing: The interpretation of a pointer as the target to which it is pointing.
derived attribute: An attribute that is determined from other attributes.
derived class: A class whose declaration indicates that it is to inherit the public members of a previously defined base class.

derived type: A user defined data type with components, each of which is either of intrinsic type or of another derived type.
destructor: An operation that cleans up an existing instance of a class that is no longer needed.
destructor operations: Methods which destroy objects and reclaim their dynamic memory.
domain: The set over which a function or relation is defined.
dummy argument: An argument in a procedure definition which will be associated with the actual (reference or value) argument when the procedure is invoked.
dummy array: A dummy argument that is an array.
dummy pointer: A dummy argument that is a pointer.
dummy procedure: A dummy argument that is specified or referenced as a procedure.
dynamic binding: The allocation of storage at run time rather than compile time, or the run time association of an object and one of its generic operations.
E
edit descriptor: An item in an input/output format which specifies the conversion between internal and external forms.
encapsulation: A modeling and implementation technique (information hiding) that separates the external aspects of an object from the internal, implementation details of the object.
exception: An unexpected error condition causing an interruption to the normal flow of program control.
explicit interface: For a procedure referenced in a scoping unit, the property of being an internal procedure, a module procedure, an external procedure that has an interface (prototype) block, a recursive procedure reference in its own scoping unit, or a dummy procedure that has an interface block.
explicit shape array: A named array that is declared with explicit bounds.
external file: A sequence of records that exists in a medium external to the program.
external procedure: A procedure that is defined by an external subprogram.
F
FIFO: First in, first out storage; a queue.
friend: A method, in C++, which is allowed privileged access to the private implementation of another
object.

function body: A block of statements that manipulate parameters to accomplish the subprogram's purpose.
function definition: Program unit that associates with a subprogram name a return type, a list of arguments,
and a sequence of statements that manipulate the arguments to accomplish the subprogram's purpose
function header: A line of code at the beginning of a function definition; includes the argument list, and the function return variable name.
G
generic function: A function which can be called with different types of arguments.
generic identifier: A lexical token that appears in an INTERFACE statement and is associated with all the procedures in the interface block.
generic interface block: A form of interface block which is used to define a generic name for a set of procedures.
generic name: A name used to identify two or more procedures, the required one being determined by the types of the non-optional arguments in the procedure invocation.
generic operator: An operator which can be invoked with different types of operands.
H
Has-A: A relationship in which the derived class has a property of the base class.
hashing technique: A technique used to create a hash table, in which the array element where an item is to be stored is determined by converting some item feature into an integer in the range of the size of the table.
heap: A region of memory used for data structures dynamically allocated and de allocated by a program.
host: The program unit containing a lower (hosted) internal procedure.
host association: Data, and variables automatically available to an internal procedure from its host.
I
information hiding: The principle that the state and implementation of an object should be private to
that object and only accessible via its public interface.
inheritance: The relationship between classes whereby one class inherits part or all of the public description of another base class, and instances inherit all the properties and methods of the classes which they contain.

instance: A individual example of a class invoked via a class constructor.
instance diagram: A drawing showing the instance connection between two objects along with the number
or range of mapping that may occur.
instantiation: The process of creating (giving a value to) instances from classes.
intent: An attribute of a dummy argument that which indicates whether it may be used to transfer data into the procedure, out of the procedure, or both.
interaction diagram: A diagram that shows the flow of requests, or messages between objects.
interface: The set of all signatures (public methods) defined for an object.
internal file: A character string that is used to transfer and/or convert data from one internal storage
mode to a different internal storage mode.
internal procedure: A procedure contained within another program unit, or class, and which can only
be invoked from within that program unit, or class.
internal subprogram: A subprogram contained in a main program or another subprogram.
intrinsic constructor: A class member function with the same name as the class which receives initial values of all the data members as arguments.
Is-A: A relationship in which the derived class is a variation of the base class.
iterator: A method that permits all parts of a data structure to be visited.

K

keyword: A programming language word already defined and reserved for a single special purpose.

L

LIFO: Last in, first out storage; a stack.
link: The process of combining compiled program units to form an executable program.
linked list: A data structure in which each element identifies its predecessor and/or successor by some form of pointer.
linker: Software that combines object files to create an executable machine language program.
list: An ordered collection that is not indexed.

M

map: An indexed collection that may be ordered.
matrix: A rank-two array.

member data: Variables declared as components of a defined type and encapsulated in a class.
member function: Subprograms encapsulated as members of a class.
method: A class member function encapsulated with its class data members.
method resolution: The process of matching a generic operation on an object to the unique method appropriate to the object's class.
message: A request, from another object, for an object to carry out one of its operations.
message passing: The philosophy that objects only interact by sending messages to each other that request some operations to be performed.
module: A program unit which allows other program units to access variables, derived type definitions, classes and procedures declared within it by USE association.
module procedure: A procedure which is contained within a module, and usually used to define generic interfaces, and/or to overload or define operators.
N
nested: Placement of a control structure inside another control structure.
O
object: A concept, or thing with crisp boundaries and meanings for the problem at hand; an instance of a class.
object diagram: A graphical representation of an object model showing relationships, attributes, and operations.
object-oriented (OO): A software development strategy that organizes software as a collection of objects that contain both data structure and behavior. (Abbreviated OO.)
object-oriented programming (OOP): Object-oriented programs are object-based, class-based, support inheritance between classes and base classes and allow objects to send and receive messages.
object-oriented programming language: A language that supports objects (encapsulating identity, data, and operations), method resolution, and inheritance.
octree: A tree structure where each node has eight child nodes.
OO (acronym): Object-oriented.
operand: An expression or variable that precedes or succeeds an operator.
operation: Manipulation of an object's data by its member function when it receives a request.

operator overloading: A special case of polymorphism; attaching more than one meaning to the same operator symbol. 'Overloading' is also sometimes used to indicate using the same name for different objects.
overflow: An error condition arising from an attempt to store a number which is too large for the storage location specified; typically caused by an attempt to divide by zero.
overloading: Using the same name for multiple functions or operators in a single scope.
overriding: The ability to change the definition of an inherited method or attribute in a subclass.
P
parameterized classes: A template for creating real classes that may differ in well-defined ways as specified by parameters at the time of creation. The parameters are often data types or classes, but may include other attributes, such as the size of a collection. (Also called generic classes.)
pass-by-reference: Method of passing an argument that permits the function to refer to the memory holding the original copy of the argument
pass-by-value: Method of passing an argument that evaluates the argument and stores this value in the corresponding formal argument, so the function has its own copy of the argument value
pointer: A single data object which stands for another (a "target"), which may be a compound object such as an array, or defined type.
pointer array: An array which is declared with the pointer attribute. Its shape and size may not be determined until they are created for the array by means of a memory allocation statement.
pointer assignment statement: A statement of the form "pointer-name) target".
polymorphism: The ability of an function/operator, with one name, to refer to arguments, or return types, of different classes at run time.
post-condition: Specifies what must be true after the execution of an operation.
pre-condition: Specifies the condition(s) that must be true before an operation can be executed.
private: That part of an class, methods or attributes, which may not be accessed by other classes, only by instances of that class.
protected: (Referring to an attribute or operation of a class in C++) accessible by methods of any descendent of the current class.
prototype: A statement declaring a function's return type, name, and list of argument types.

pseudocode: A language of structured English statements used in designing a step-by-step approach to solving a problem.
public: That part of an object, methods or attributes, which may be accessed by other objects, and thus constitutes its interface.

Q

quadtree: A tree structure where each tree node has four child nodes.
query operation: An operation that returns a value without modifying any objects.

R

rank: Number of subscripted variables an array has. A scalar has rank zero, a vector has rank one, a matrix has rank two.

S

scope: That part of an executable program within which a lexical token (name) has a single interpretation.
section: Part of an array.
sequential: A kind of file in which each record is written (read) after the previously written (read) record.
server: An object that can only be operated upon by other objects.
service: A class member function encapsulated with its class data members.
shape: The rank of an array and the extent of each of its subscripts. Often stored in a rank-one array.
side effect: A change in a variable's value as a result of using it as an operand, or argument.
signature: The combination of a subprogram's (operator's) name and its argument (operand) types. Does not include function result types.
size: The total number of elements in an array.
stack: Region of memory used for allocation of function data areas; allocation of variables on the stack occurs automatically when a block is entered, and deallocation occurs when the block is exited
stride: The increment used in a subscript triplet.
strong typing: The property of a programming language such that the type of each variable must be declared.
structure component: The part of a data object of derived type corresponding to a component of its type.
sub-object: A portion of a data object that may be referenced or defined independently of other portions. It may be an array element, an array section, a structure component, or a substring.

subprogram: A function or subroutine subprogram.
subprogram header: A block of code at the beginning of a subprogram definition; includes the name, and the argument list, if any.
subscript triplet: A method of specifying an array section by means of the initial and final subscript integer values and an optional stride (or increment).
super class: A class from which another class inherits. (See base class.)
supplier: Software component that implements a new class with services to be used by a client software component.

T

target: The data object pointed to by a pointer, or reference variable.
template: An abstract recipe with parameters for producing concrete code for class definitions or subprogram definitions.
thread: The basic entity to which the operating system allocates CPU time.
tree: A form of linked list in which each node points to at least two other nodes, thus defining a dynamic data structure.

U

unary operator: An operator which has only one operand.
undefined: A data object which does not have a defined value.
underflow: An error condition where a number is too close to zero to be distinguished from zero in the floating-point representation being used.
utility function: A private subprogram that can only be used within its defining class.

V

vector: A rank-one array. An array with one subscript.
vector subscript: Amethod of specifying an array section by means of a vector containing the subscripts of the elements of the parent array that are to constitute the array section.
virtual function: A genetic function, with a specific return type, extended later for each new argument type.
void subprogram: A C++ subprogram with an empty argument list and/or a subroutine with no returned argument.

W

work array: A temporary array used for the storage of intermediate results during processing.

Case Studies

Q:1 Write a program to print you name.

Q:2 Write a program to print addition of two numbers

Q:3 Write a program to print multiplication of 2 numbers a and b

Q:4 Write a program to print numbers from 1 to 10 using while loop

Q:5 Write a program to print numbers from 1 to 10 using for loop

Q:6 Write a program to use of try and catch block in exception handling.

Q:7 Write a program for factorial using recursion.

Q:8 Write a program to print Multiplication table of a number using 2-D array.

Q:9 Write a program to create an applet window and show the shape of rectangle in this.

Q:10 Write a program to handle Mouse events.

Bibliography

1. Object Oriented Programming With C by E Balagurusamy
2. Programming with Java: A Primer ,Tata McGraw-Hill Education
3. Java 6 Programming Black Book, New Ed, Kogent Solution Inc.
4. Visual Basic 6 Programming Black Book , Steven Holzner
5. The Visual Basic .Net Programming Language, Paul Vick

Websites:

www.roseindia.net
www.javapoint.com
www.tutorialspoint.com
www.davmanuals.com