



I Internal Examination Sept. 2018

Class: - BCA I

Subject: - Principles of Programming Lang. (BCA 104)

MM: 40

Set: A

Time: 1 ½ Hrs.

[I]Very short answer questions (Max 40 words).

(5 * 2 = 10)

1. What is a programming language?

A **programming language** is a formal language, which comprises a set of instructions used to produce various kinds of output. Programming languages are used to create programs that implement specific algorithms.

2. What are variables?

Variables are the names you give to computer memory locations which are used to store values in a computer program. For example, assume you want to store two values 10 and 20 in your program and at a later stage, you want to use these two values.

3. What is 'int'? What is the size of any integer variable?

Int, short for "integer," is a fundamental variable type built into the compiler and used to define numeric variables holding whole numbers. The size of int is either 2 bytes (In older PC's) or 4 bytes.

4. What is the use of '%' operator? Explain with example.

The **modulus operator** ('%'), that computes the remainder that results from performing integer division.

```
int main()
{
    int num;
    cin >> num;
    // num % 2 computes the remainder when num is divided by 2
    if ( num % 2 == 0 )
    {
        cout << num << " is even ";
    }

    return 0;
}
```

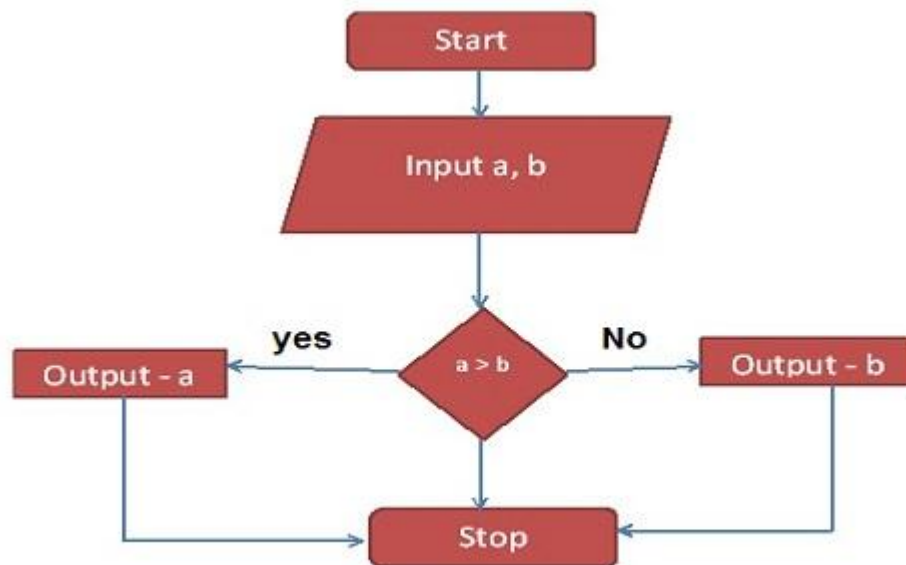
5. **What do you understand by sizeof()?**

sizeof is a much used operator in the C programming language. It is a compile time unary operator which can be used to compute the size of its operand. The result of *sizeof* is of unsigned integral type which is usually denoted by *size_t*. *sizeof* can be applied to any data-type, including primitive types

[II] Short answer questions (Max 80 words).

(2 * 5 = 10)

1. **Draw a flowchart to print the larger of two numbers given by user.**



2. **What is the difference between unary and binary operators? Explain with example.**

Difference b/w Unary and Binary Operators - In this section you will learn about Unary and Binary Operators. Operator is a symbol or special character which is used to perform a specific task, the task/meaning of operator is defined in the compiler. For example + is an operator which is used to add two values.

The Operators which operate on Single Operand known as Unary Operators, some of the unary operators are:

- ++ Increment Operator
- Decrement Operator
- & Address Of Operator
- Unary Minus Operators
- ~ (One's Compliment) Negation Operator
- ! Logical NOT

and so on.

The Operators which operate on Two Operands known as Binary Operators, some of the binary operators are:

- + Binary Plus Operator
- Binary Minus Operator
- == Equal to Operator
- < Less than Operator
- and so on..

[III] Long answer questions (Max 150 words).

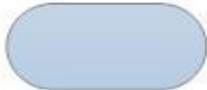


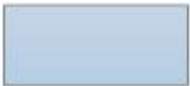

(2 * 10 = 20)

1. What do you understand by the term flowchart? Explain with all the symbols used to make a flowchart. Draw a flowchart to print the sum of two numbers given by user.

Flowcharts use special shapes to represent different types of actions or steps in a process. Lines and arrows show the sequence of the steps, and the relationships among them. These are known as flowchart symbols.

The type of diagram dictates the flowchart symbols that are used. For example, a data flow diagram may contain an Input/Output Symbol (also known as an I/O Symbol), but you wouldn't expect to see it in most process flow diagrams.

Over the years, as technology has evolved, so has flowcharting. Some flowchart symbols that were used in the past to represent computer punchcards, or punched tape, have been relegated to the dustbin of history.

Symbol	Name	Function
	Start/end	An oval represents a start or end point
	Arrows	A line is a connector that shows relationships between the representative shapes
	Input/Output	A parallelogram represents input or output
	Process	A rectangle represents a process
	Decision	A diamond indicates a decision

2. What are identifiers? State all the rules to form identifiers.

C Identifiers

Identifier refers to name given to entities such as variables, functions, structures etc.

Identifier must be unique. They are created to give unique name to a entity to identify it during the execution of the program. For example:

int money;

double accountBalance;

Here, money and accountBalance are identifiers.

Also remember, identifier names must be different from keywords. You cannot use int as an identifier because int is a keyword.

Rules for writing an identifier

1. A valid identifier can have letters (both uppercase and lowercase letters), digits and underscores.
2. The first letter of an identifier should be either a letter or an underscore. However, it is discouraged to start an identifier name with an underscore.
3. There is no rule on length of an identifier. However, the first 31 characters of identifiers are discriminated by the compiler.



I Internal Examination Sept. 2018

Class: – BCA I

Subject: – Principles of Programming Lang. (BCA 104)

MM: 40

Set: B

Time: 1 ½

Hrs.

[1] Very short answer questions (Max 40 words).

(5 * 2 = 10)

1. Why do we need to learn any programming language?

A **programming language** is a formal language, which comprises a set of instructions used to produce various kinds of output. Programming languages are used to create programs that implement specific algorithms.

2. What are keywords?

Keywords are predefined, reserved words used in programming that have special meanings to the compiler. Keywords are part of the syntax and they cannot be used as an identifier. For example:

```
int money;
```

3. What is 'float'? What is the size of a floating point variable?

Float is a term is used in various programming languages to define a variable with a fractional value. Numbers created using a float variable declaration will have digits on both sides of a decimal point. This is in contrast to the integer data type, which houses an integer or whole number.

4. What is the use of '~' operator? Explain with example.

~	Binary Ones Complement Operator is unary and has the effect of 'flipping' bits.	(~A) = -60, i.e., 1100 0100 in 2's complement form.
---	---	---

5. What do you understand by clrscr() and getch()?

Clrscr() is a predefined function in "conio.h" (console input output header file) used to clear the console screen.

It is a predefined function in "conio.h" (console input output header file) will tell to the console wait for some time until a key is hit given after running of program.

[II] Short answer questions (Max 80 words).

(2 * 5 = 10)

1. What is the difference between variable initialization and variable assignment? Explain with example.

Initialization means telling compiler to allocate a memory to the initialized variable so that compiler can know that this variable is going to store some value.

Ex : int a; // Initializing 'a' as of type int (Here compiler will allocate a memory of 4 bytes to 'a').

Assignment means proving a particular value to any variable.

Ex: int a=2;// a in assigned a value to 2.

2. Explain the work of right shift and left shift operators with example.

Bitwise Right Shift Operator in C

1. It is denoted by >>

2. Bit Pattern of the data can be shifted by specified number of Positions to Right
3. When Data is Shifted Right , leading zero's are filled with zero.
4. Right shift Operator is Binary Operator [Bi – two]
5. Binary means , Operator that require two arguments

Bitwise Left Shift Operator in C

1. It is denoted by \ll
2. Bit Pattern of the data can be shifted by specified number of Positions to Left
3. When Data is Shifted Left , trailing zero's are filled with zero.
4. Left shift Operator is Binary Operator [Bi – two]
5. Binary means , Operator that require two arguments

[III] Long answer questions (Max 150 words).

(2 * 10 = 20)

- 1. What do understand by the term data types? What are the various data types available in C? Explain with their size and example.**

In C, there is a concept called "datatypes". Data types indicate the type of data a variable can hold. When a variable is defined, a memory location will be assigned to the newly defined variable and it will also define the type of data that memory location will hold. C has following data types

- int - an integer; reflects size of integers on host machine
- float - single-precision floating point
- double - double-precision floating point
- char - character, a single byte

In addition to basic data types C also defines certain qualifiers to these data types. Qualifiers are used to make variable declaration more specific to variable uses. Qualifiers available in the C language are:

- short (applied to integers)
- long (applied to integers)
- signed (applied to char, or any integer)
- unsigned (applied to char, or any integer)

With application of these qualifiers basic data types can be flavoured in many ways as given in the table below. Note that the values given are acceptable *minimum magnitudes* defined by the C Standard - each implementation will define values greater or equal in magnitude.

- 2. What is the difference between operator precedence and associability? Explain with example.**

Precedence of operators

If more than one operators are involved in an expression, C language has a predefined rule of priority for the operators. This rule of priority of operators is called operator precedence.

In C, precedence of arithmetic operators(*, %, /, +, -) is higher than relational operators(==, !=, >, <, >=, <=) and precedence of relational operator is higher than logical operators(&&, || and !).

Example of precedence

```
(1 > 2 + 3 && 4)
```

This expression is equivalent to:

```
((1 > (2 + 3)) && 4)
```

i.e, (2 + 3) executes first resulting into 5

then, first part of the expression (1 > 5) executes resulting into 0 (false)

then, (0 && 4) executes resulting into 0 (false)

Output

```
0
```

Associativity of operators

If two operators of same precedence (priority) is present in an expression, Associativity of operators indicate the order in which they execute.

Example of associativity

```
1 == 2 != 3
```

Here, operators == and != have same precedence. The associativity of both == and != is left to right, i.e, the expression on the left is executed first and moves towards the right.

Thus, the expression above is equivalent to :

```
((1 == 2) != 3)
```

i.e, (1 == 2) executes first resulting into 0 (false)

then, (0 != 3) executes resulting into 1 (true)