

# 10th Class

Computer Science	Model Paper 6	Paper: II
Time: 1.45 Hours	(Subjective Type)	Marks: 40

## (Part-I)

2. Write short answers to any FOUR (4) questions: (8)

(i) What is a text editor?

**Ans** A text editor is a software that allows programmers to write and edit computer programs. All IDEs have their own specific text editors. It is the main screen of an IDE where we can write our programs.

(ii) What is linking section or header section?

**Ans** While writing programs in C language, we make extensive use of functions that are already defined in the language. But before using the existing functions, we need to include the files where these functions have been defined. These files are called header files.

(iii) What is character set in C programming language?

**Ans** In C programming language, we have a character set that includes:

1. Alphabets (A, B, ....., Y, Z). (a, b....y, z).
2. Digit (0-9).
3. Special symbol (~ ` ~ @ # % ^ & \* ( ) \_ - + = | \ { } [ ] : ; " ' < > , . ? /).

The alphabets, digit and special symbols when combined in an allowable manner, form constants, variables and keywords (also known as reserved words).

(iv) What is the importance of variable declaration?

**Ans** A variable cannot be declared unless we mention its data type. After declaring a variable, its data type cannot be changed. Declaring a variable specifies the type of variable, the range of values allowed, and the kind of operations that can be performed on it.

(v) What is difference between printf and getch?

**Ans** printf is a built-in function in C programming language to show output on screen. Its name comes from "print formatted" that is used to print the formatted output on screen.

getch() function is used to read a character from user. The character entered by user does not get displayed on screen. This function is generally used to hold the execution of program because the program does not continue further until the user types a key. To use this function, we need to include the library conio.h in the header section of program.

(vi) What is escape sequence Tab (\t) in C language?

**Ans** Escape sequence \t specifies the I/O function of moving the next tab stop horizontally. A tab stop is collection of 8 spaces. Using \t takes cursor to the next tab stop. This escape sequence is used when user presents data with more spaces.

**3. Write short answers to any FOUR (4) questions: (8)**

(i) Define Arithmetic operators.

**Ans** Arithmetic operators are used to perform arithmetic operations on data.

(ii) What is the use of Addition operator?

**Ans** Addition operator (+) calculates the sum of two operands.

**Example:**

```
int add = 10 + 10;
```

Resultant value in variable add is 20.

(iii) What is AND operator?

**Ans** AND operator && takes two Boolean expressions as operands and produces the result true if both of its operands are true. It returns false if any of the operands is false.

(iv) Define selection statements.

**Ans** The statements which help us to decide which statements should be executed next, on the basis of conditions, are called selection statements.

(v) How we can associate more than one statements to an if statement?

**Ans** If we want to associate more than one statements to an if statement, then they need to be enclosed inside a { } block.

(vi) What is common mistake in compound statement?

**Ans** In compound statements, it is a common mistake to omit one or two braces while typing. To avoid this error, it is better to type the opening and closing braces first and then type the statements in the block.

4. Write short answers to any FOUR (4) questions: (8)

(i) What is difference between array type and array size?

**Ans**

Array Name	Array Size
Array name is the unique identifier that we use to refer to the array.	Array size is the maximum number of elements that they array can hold.

(ii) Write down the syntax of for loop in C language.

**Ans** for initialization; condition;  
increment/decrement)

{

Code to repeat

}

(iii) Write a program that assigns first 5 multiples of 23 to an array of size 5.

**Ans** #include <stdio.h>

void main ()

{

int multiples[5];

for(int i=0; i<5; i++)

    multiples[i] = (i + 1) \* 23;

}

(iv) - Write down advantages of functions.

**Ans** Functions provide us following advantages:

1. Reusability
2. Separation of task
3. Handling the complexity of the problem
4. Readability

(v) What is the body of the function?

**Ans** A function is a block of statements that gets some inputs and provides some output. Inputs of a function are called parameters of the function, and output of the function is called its return value. A function can have multiple parameters, but it cannot return more than one values.

(vi) Identify the error from the following code.

```
void message ();  
{  
    printf ("Hope you are fine:");  
    return 23;
```

**Ans** Error: After function definition, there is no semicolon.

(Part-II)

NOTE: Attempt any TWO (2) questions.

Q.5. Define If-else statement. Write its structure with the help of example. (8)

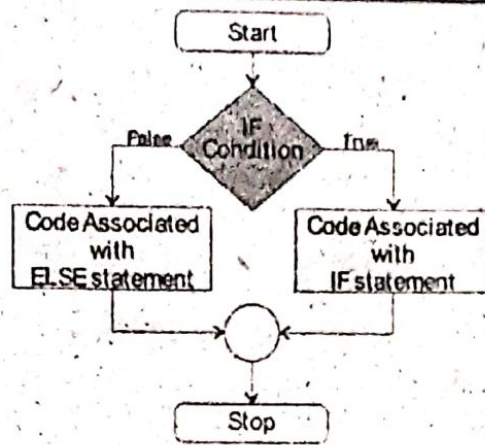
**Ans** If-else Statement:

We know how to execute a set of instructions if a particular condition is *true*, but if the condition turns out to be *false*; then we are not doing anything. What if we want to execute one set of instructions if a particular condition is *true* and another set of instructions if the condition is *false*. In such situations, we use *if-else statement*. It executes the set of statements under *if statement*, if the condition is true, otherwise executes the set of statements under else statement.

General structure of the *if-else statement* is as follows:

```
if (condition)  
    Associated Code  
else  
    Associated Code
```

Associated code of *if statement* is executed if the condition is *true*, otherwise the code associated with *else statement* is executed. Following flow chart shows the structure of *if-else statement*.



Before else keyword, if there are multiple statements under if, then they must be enclosed inside the { } block, otherwise compiler issues an error. In order to understand this concept, let's look at the following example.

**Example:**

```

#include<stdio.h>
void main ()
{
    int a = 15;
    if (a % 2 == 0)
        printf("The variable a contains an even value.");
        printf("\nYou are doing a great job.");
    else
        printf("The variable a contains an odd value.");
}
  
```

The above code cannot be compiled, because, without a { } block only one statement is associated with if statement. In this case, the 1<sup>st</sup> statement i.e., printf ("The variable a contains an even value."); is associated with if statement, but the 2<sup>nd</sup> statement i.e., printf ("\nYou are doing a great job."); is not associated with if statement. So, the else part is also disconnected from if statement. We know that an else block must be associated to an if block. In order to solve this problem, we can put both statements before else keyword inside { } block.

**Q.6. Write a program that stores the ages of five persons in an array, and then displays on screen. (8)**

**Ans**

```

#include<stdio.h>
void main ()
{
  
```

```

int age[5];
/* Following statements assign values at different
indices of array age. We can see that the first value
is stored at index 0 and the last value is stored at
index 4
*/
age[0] = 25;
age[1] = 34;
age[2] = 29;
age[3] = 43;
age[4] = 19;
/* Following statement displays the ages of five
persons stored in the array */
printf("The ages of five persons are: %d, %d, %d, %d
%d, age[0], age[1], age[2], age[3], age[4]);
}

```

**Q.7. What is "defining a function"? Explain "Using a function" with the help of example. (8)**

**Ans** Defining a Function:

The function signature does not describe how the function performs the task assigned to it. Function definition does that.

A function definition has the following general structure:

```

return_type function_name (data_type var1, data_type var2,...,
data_type varN)
{
    Body of the function
}

```

*Body of the function* is the set of statements which are executed in the function to perform the specified task. Just after the function's signature, the set of statements enclosed inside {} form the body of the function.

Following example defines a function showPangram() that does not take any input and does not return anything, but displays A quick brown fox jumps over the lazy dog. on computer screen.

**Example:**

```

void showPangram ()
{
    printf("\nA quick brown fox jumps over the lazy dog.\n");
}

```

function name

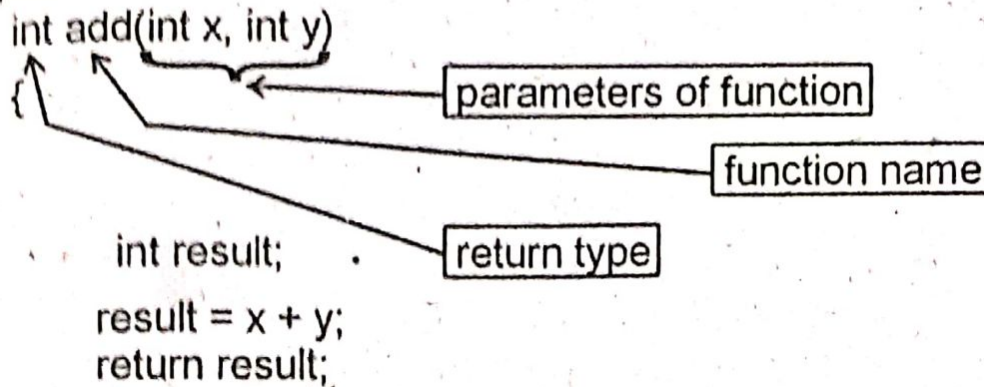
↑

} As the above function does not return anything thus return type of the function is void.

Let's take another example of a function that takes as input two integers and returns the sum of both integers.

Example:

```
int add(int x, int y)
{
    int result;
    result = x + y;
    return result;
}
```



Inside the function, *return* is a keyword that is used to return a value to the calling function.

**Using a Function:**

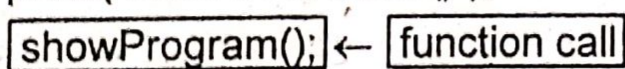
We need to call a function, so that it performs the programmed task. Following is the general structure used to make a function call.

```
function_name (value1, value2, ..., valueN);
```

For example, let's observe the following program.

Example:

```
void main ()
{
    printf("Hello from main ()");
    showProgram(); ← function call
    printf("Welcome back to main ()");
}
```



**Output:**

```
Hello from main ()
A quick brown fox jumps over the lazy dog.
Welcome back to main ()
```

We can see that the program starts its execution from *main()* function. When it encounters a function call (inside the rectangle), it transfers the control to called function. After executing the statements of called function, the control is transferred back to the called function, i.e., *main()* in the above example.