Session A 07

Monday, 6 October 2025 9:25 AM

Raj Kumar Goel Institute of Technology, Ghaziabad
Sessional Test 1 (Odd Semester) 2025-26
B.Tech. First Year

Semester: I
Subject Name: PPS | Subject Code: BCS 101 | Total Marks: 50 | Time: 01:30 Hours

Note: *Attempt all Sections. If you require any missing data, then choose suitably.*

Section – A					
Q.1	Attempt **all** parts of the following. (2 x 5=10)				
(a)	Draw the memory hierarchy diagram.	K1	CO1	2
(b)	What is difference b/w object code & source code.	K1	CO1	2
(c)	Define the Identifier with its rules.	K2	CO1	2

(d)	What is the output of this C code?
	int main()
	{
	int x = 5;
	int y = 5;
	x = x --y;
	printf("%d", x--);
	}
		K2	CO2	2

(e)	What is Linker in C language with a proper diagram?	K1	CO2	2
Section – B					
Q.2	Attempt **all** parts of the following. (5 x 4 =20)				
(a)	Explain the following: (i) Compiler & Interpreter (ii)Functions of operating system	K2	CO1	5
(b)	Explain the types of Errors in C language?	K2	CO1	5
(c)	Draw a flow chart and program to check the given value is Even or Odd.	K2	CO2	5
(d)	Explain the operator precedence and associativity with an example.	K2	CO2	5
Section – C					
Q.3	Attempt **any one** part of the following. (10x 1 = 10)				
(a)	Draw the architecture of digital computer system and explain its all components.	K2	CO1	10
(b)	Explain various types of data types with their size, range and format specifier.	K2	CO1	10
Q.4	Attempt **any one** part of the following. (10x 1 = 10)				
(a)	Explain with proper example: (i) Conditional operator (ii) Bitwise Operator (iii) Logical Operator	K3	CO2	10
(b)	Write a program to find the greatest value into three values using conditional operator.	K3	CO2	10

int a = 4,10;

Array :- Datatype

Homogeneous

In **C programming**, an **array** is a collection of elements **of the same data type**, stored **in contiguous memory locations**. Arrays are used to store multiple values in a single variable, instead of declaring separate variables for each value.

int arr[5];

1 2 3 4 5 6 7 8 9 10

🔷 Types of Arrays

1. **One-Dimensional Array** – A simple list of elements.
2. **Two-Dimensional Array** – Used to represent matrices or tables.
3. **Multi-Dimensional Array** – Arrays with more than two dimensions (rarely used).

🔷 1. One-Dimensional Array

Syntax:

```
data_type array_name[size];
```

Example:

```
#include <stdio.h>
int main() {
   int marks[5] = {85, 90, 78, 92, 88};
for(int i = 0; i < 5; i++) {
    printf("marks[%d] = %d\n", i, marks[i]);
   }
return 0;
}
```

Output:

```
marks[0] = 85
marks[1] = 90
marks[2] = 78
marks[3] = 92
marks[4] = 88
```

Int arr[] = {12, 14, 10, 20};

→ Constant Pointer. The value can't be changed.

Pointer Variable addresses of a Variable

🔷 2. Two-Dimensional Array (Matrix)

Syntax:

```
data_type array_name[row_size][col_size];
```

Example:

```
#include <stdio.h>
int main() {
   int matrix[2][3] = {{1, 2, 3}, {4, 5, 6}};
for(int i = 0; i < 2; i++) {
      for(int j = 0; j < 3; j++) {
         printf("%d ", matrix[i][j]);
      }
      printf("\n");
   }
return 0;
}
```

Output:

```
1 2 3
4 5 6
```

🔷 Accessing Array Elements

- Array elements are accessed using their **index**.
- Indexing starts from **0**.
  Example: marks[0] → first element, marks[4] → last element.

🔷 Important Points

- Array size must be a **constant** at compile time (in standard C).
- The name of the array (e.g., marks) represents the **base address** (starting address) of the array.
- You can use arrays with **loops** to perform operations easily.

🔷 Example: Sum of Array Elements

```
#include <stdio.h>
int main() {
   int arr[5] = {10, 20, 30, 40, 50};
   int sum = 0;
for(int i = 0; i < 5; i++) {
      sum += arr[i];
   }
printf("Sum = %d", sum);
   return 0;
}
```

Output:

```
Sum = 150
```

```
#include <stdio.h>
void inputArray(int arr[], int n)
{
    for (int i = 0; i < n; i++)
    {
        printf("Enter the Value of index [%d] ", i);
        scanf("%d", &arr[i]);
    }
}
void printArray(int arr[], int n)
{
    printf("The Array is:\n ");
    printf("[");
    for (int i = 0; i < n; i++)
    {
        printf(" %d ", arr[i]);
    }
    printf("]\n");
}
int main()
{
    int arr[5];
    inputArray(arr, 5);
    printArray(arr, 5);
    return 0;
}
```