USN | | | | | | | | | | |

Internal Assessment Test II –January 2026 – Set 2

Sub:	Introduction to C Program					Sub Code:	1BPLC205E	Branch:	ECE	
Date:	08-01-2026	Duration:	90 min's	Max Marks:	50	Sem/Sec:	I/ M, N,O & P		OBE	

	Answer any FIVE FULL Questions	MARKS	CO	RBT
1	a. What is an array? Explain how two-dimensional arrays are declared and initialized. Write a C program to pass a two-dimensional array to the function and display in matrix format. b. Write a C program to sort the given set of N numbers using Bubble Sort technique. List the applications of arrays.	5 5	CO3	L3
2	a. Define String. How string is declared and initialized? Develop a C program to concatenate two strings without using built-in function. b. Using suitable code, Discuss the working of the following string functions i. strcat() ii. strlen() iii. strcpy() iv. strcmp()	5 5	CO3	L3
3	a. Define function in C. Justify the need of user defined functions in C with a suitable program b.Explain with example "Arguments and return values" of function.	5 5	CO4	L2 L3
4	Develop a C program to perform arithmetic operations (+, -, /, *) using user defined functions	10	CO4	L3
5	Define a structure type student that would contain student name, 3 subject marks Using this structure, Develop a C program to read four students data from keyboard and print the same on the screen.	10	CO5	L3
6	Define Pointer. Explain pointer variable declaration and initialization and Illustrate with suitable code for swapping of two numbers using pointers.	10	CO5	L3
7	**Write a C Program to define** a structure named Employee with the following members: • Employee ID (int) • Salary (float) **Read** the details of N employees using an array of structures and calculate the average salary **Display:** • All employee details • Average salary	10	CO5	L3

Faculty Signature CCI Signature HOD Signature

Internal Assessment Test II – Answers
Introduction to C Programming
Solutions

1. **A. What is an array? Explain how two-dimensional arrays are declared and initialized.**

Array:
An array is a collection of similar data elements stored in contiguous memory locations and accessed using a single name.

Declaration of 2D Array:
int a[3][3];

Initialization:
int a[3][3] = {{1,2,3},{4,5,6},{7,8,9}};

b. Write a C program to pass a two-dimensional array to the function and display in matrix format.

Program to pass 2D array to function:

```
#include<stdio.h>
void display(int a[3][3])
{
 for(int i=0;i<3;i++)
 {
  for(int j=0;j<3;j++)
   printf("%d ",a[i][j]);
  printf("\n");
 }
}
int main()
{
 int a[3][3]={{1,2,3},{4,5,6},{7,8,9}};
 display(a);
 return 0;
}
```

b. **Write a C program to sort the given set of N numbers using Bubble Sort technique. List the applications of arrays.**

Bubble Sort Program:

```
#include<stdio.h>
int main()
{
 int a[50],n,temp;
 printf("Enter n:");
 scanf("%d",&n);
 for(int i=0;i<n;i++)
  scanf("%d",&a[i]);
 for(int i=0;i<n-1;i++)
 {
  for(int j=0;j<n-i-1;j++)
  {
   if(a[j]>a[j+1])
   {
    temp=a[j];
    a[j]=a[j+1];
    a[j+1]=temp;
   }
  }
 }
 printf("Sorted Array:");
 for(int i=0;i<n;i++)
  printf("%d ",a[i]);
}
```

Applications of Arrays:

1. Storing data
2. Searching & sorting
3. Matrix operations
4. Statistical analysis

2. **A. Define String. How string is declared and initialized? Develop a C program to concatenate two strings without using built-in function.**

Definition:
A string is a sequence of characters terminated by '\0'.

Declaration:
char s[20];

Initialization:
char s[]="Hello";

Program to concatenate strings:

```
#include<stdio.h>
int main()
{
 char s1[50],s2[50];
 int i,j;
 gets(s1);
 gets(s2);
 for(i=0;s1[i]!='\0';i++);
 for(j=0;s2[j]!='\0';j++)
 {
  s1[i]=s2[j];
  i++;
 }
 s1[i]='\0';
 puts(s1);
}
```

String functions:
strcat() – joins strings
strlen() – finds length
strcpy() – copies string
strcmp() – compares strings

b. **Using suitable code, discuss the working of the following string functions**

i) strcat()
Purpose: Joins one string to the end of another.

Example:

```
#include<stdio.h>
#include<string.h>
int main()
{
   char s1[20]="Hello";
   char s2[10]="World";
   strcat(s1,s2);
   printf("Result: %s",s1);
   return 0;
}
```

Working:
Appends s2 to s1.
Output: HelloWorld

---

ii) strlen()
Purpose: Returns number of characters in a string (excluding '\0').

Example:

```
#include<stdio.h>
#include<string.h>
int main()
{
   char s[]="Computer";
   printf("Length = %d", strlen(s));
   return 0;
}
```

Working:
Counts characters.
Output: 8

---

iii) strcpy()
Purpose: Copies one string into another.

Example:

```
#include<stdio.h>
#include<string.h>
int main()
{
   char s1[20], s2[20]="C Programming";
   strcpy(s1,s2);
   printf("Copied string: %s",s1);
   return 0;
}
```

Working:
Copies content of s2 into s1.

---

iv) strcmp()
Purpose: Compares two strings.

Return Values:
0 – strings are equal
>0 – first string is greater
<0 – first string is smaller

Example:

```
#include<stdio.h>
#include<string.h>
int main()
{
   char s1[]="Apple";
   char s2[]="Banana";
   int result = strcmp(s1,s2);
   printf("Result = %d", result);
   return 0;
}
```

Working:
Compares strings character by character.

3. A. Define function in C. Justify the need of user defined functions in C with a suitable program

Definition:
Function is a block of code that performs a specific task.

Need:
Modularity, reuse, readability

Example:

```
#include<stdio.h>
void add()
{
 int a=10,b=20;
 printf("Sum=%d",a+b);
}
int main()
{
 add();
}
```

Arguments & return values:

```
int sum(int a,int b)
{
 return a+b;
}
```

b. Explain with example "Arguments and return values" of function

Arguments and return values are used to pass data between the calling function and the called function.

Arguments:
Arguments are the values passed to a function when it is called. They allow data to be sent from the main function to the called function.

Return Value:
Return value is the data sent back from the called function to the calling function using the return statement.

Example Program:

```
#include<stdio.h>

int add(int a, int b)   // Function with arguments and return value
{
   int sum;
   sum = a + b;
   return sum;       // Returning result
}

int main()
{
   int x = 10, y = 20;
   int result;

   result = add(x, y);   // Passing arguments
   printf("Sum = %d", result);

   return 0;
}
```

4. **Develop a C program to perform arithmetic operations (+, -, /, *) using user defined functions**

Arithmetic using Functions

```
#include<stdio.h>
int add(int a,int b){return a+b;}
int sub(int a,int b){return a-b;}
int mul(int a,int b){return a*b;}
float div(int a,int b){return a/(float)b;}

int main()
{
 int a,b;
 scanf("%d%d",&a,&b);
 printf("Add=%d",add(a,b));
 printf("\nSub=%d",sub(a,b));
 printf("\nMul=%d",mul(a,b));
 printf("\nDiv=%.2f",div(a,b));
}
```

5. Define a structure type student that would contain student name, 3 subject marks Using this structure, Develop a C program to read four students data from keyboard and print the same on the screen.

```
#include<stdio.h>
struct student
{
 char name[20];
 int m1,m2,m3;
};
int main()
{
 struct student s[4];
 for(int i=0;i<4;i++)
 {
  scanf("%s",&s[i].name);
  scanf("%d%d%d",&s[i].m1,&s[i].m2,&s[i].m3);
 }
 for(int i=0;i<4;i++)
 {
  printf("%s %d %d %d",s[i].name,s[i].m1,s[i].m2,s[i].m3);
 }
}
```

6. Define Pointer. Explain pointer variable declaration and initialization and Illustrate with suitable code for swapping of two numbers using pointers.

Definition:
Pointer is a variable that stores address of another variable.

Declaration:
```
int *p;
```

Initialization:
```
p=&a;
```

Swap program:
```
#include<stdio.h>
void swap(int *x,int *y)
{
 int t=*x;
 *x=*y;
 *y=t;
```

```
}
int main()
{
 int a=10,b=20;
 swap(&a,&b);
 printf("%d %d",a,b);
}
```

7. Define a structure named Employee with following members:
   - Employee (int)
   - Salary (float)

Read the details of N employees using an array of structure and calculate the average of salary

Display

- All employee details
- Average salary

```
#include<stdio.h>
struct emp
{
 int id;
 float sal;
};
int main()
{
 struct emp e[10];
 int n;
 float sum=0,avg;
 scanf("%d",&n);
 for(int i=0;i<n;i++)
 {
  scanf("%d%f",&e[i].id,&e[i].sal);
  sum+=e[i].sal;
 }
 avg=sum/n;
 for(int i=0;i<n;i++)
 {
  printf("%d %.2f",e[i].id,e[i].sal);
```

```
 }
 printf("Average Salary=%.2f",avg);
}
```