CBCS SCHEME

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

1BPLC105E

First Semester B.E./B.Tech. Degree Examination, Dec.2025/Jan.2026
Introduction to C Programming

Time: 3 hrs. Max. Marks: 100

Note: 1. Answer any FIVE full questions, choosing ONE full question from each module.
2. M : Marks , L: Bloom's level , C: Course outcomes.

			M	L	C
		Module – 1			
Q.1	a.	Define Algorithm. Develop an algorithm to find the average of three numbers taken as input from the user.	5	L2	CO1
	b.	Define Variable? Explain the rules for constructing variables in C language. Identify whether the following variable is valid or invalid , State the reason for the same. i) int ii) area iii) 20group_one iv) $type v) _ptr123.	10	L2	CO1
	c.	List the features of C programming language. Explain the process of compiling and executing a C Program.	5	L2	CO1
		OR			
Q.2	a.	Define Flow chart. List the symbols used in designing a flowchart. Write a flowchart to calculate area of circle.	6	L2	CO1
	b.	Explain the basic structure of C program with a programming example.	10	L2	CO1
	c.	Explain Input and Output functions in C Programming with suitable example.	4	L2	CO1
		Module – 2			
Q.3	a.	Define an Operator. List different types of operators of C and explain any FIVE operators with an example.	10	L2	CO1
	b.	Explain if else statement with syntax and a suitable program.	5	L2	CO2
	c.	Develop a C program to find whether a given number is prime or not.	5	L3	CO2
		OR			
Q.4	a.	Differentiate between entry controlled loop and exit controlled loop.	6	L2	CO2
	b.	Explain switch statement with syntax and a suitable program.	8	L2	CO2
	c.	Develop a C program to find the roots of quadratic equations.	6	L3	CO2
		Module – 3			
Q.5	a.	Define Array? Demonstrate the declaration and different initialization methods of One-Dimensional array with syntax and example.	7	L2	CO3
	b.	Develop a C program to find the length of a string without using built in function.	5	L3	CO3

	c.	Develop a C program to find key elements in an array using linear search.	8	L3	CO3
		OR			
Q.6	a.	Define a String? Explain declaration and initialization of strings with example.	5	L3	CO3
	b.	List and explain any FIVE String-handling Functions with example.	7	L2	CO3
	c.	Develop a C program to concatenate two strings, find length of a string and copy one string to other using string operations.	8	L3	CO3
		Module – 4			
Q.7	a.	Define a function? Explain the various elements of user defined functions with suitable example.	10	L2	CO4
	b.	Explain function with arguments and no return value with suitable programming example.	10	L2	CO4
		OR			
Q.8	a.	Explain function with no arguments and no return value with suitable programming example.	10	L2	CO4
	b.	Develop a modular C program to find GCD and LCM of given numbers using user defined functions.	10	L3	CO4
		Module – 5			
Q.9	a.	Define structure. Explain the general format of a structure definition with example.	7	L2	CO5
	b.	Differentiate between arrays and structures with an example.	7	L2	CO5
	c.	Define pointer. Illustrate declaring and initialization of a pointer variable with an example.	6	L2	CO5
		OR			
Q.10	a.	Explain Arrays within Structure with suitable programming example.	10	L2	CO5
	b.	Develop a C program to declare the structure of employees and display the employee records with higher salary among two employees.	10	L3	CO5

* * * * *

INTRODUCTION TO C PROGRAMMING-1BPLC105E

Complete Answers for All Questions (As per Question Paper)

MODULE 1

Q1(a) Algorithm & Average of Three Numbers

Algorithm: A finite step-by-step procedure to solve a problem.

Algorithm to find average of three numbers:

1. Start
2. Read A, B, C
3. SUM = A + B + C
4. AVG = SUM / 3
5. Print AVG
6. Stop

Q1(b) Variable & Rules

Variable: A named memory location used to store data.

Rules:

- Must start with letter or underscore
- Cannot use keywords
- No special symbols except underscore
- Case sensitive

Check Validity:

i) int – Invalid (keyword)
ii) area – Valid
iii) 20group_one – Invalid (starts with digit)
iv) $type – Invalid ($ not allowed)
v) _ptr123 – Valid

Q1(c) Features of C & Compilation Process

Features:

- Structured language
- Portable
- Efficient and fast
- Rich library
- Modular programming

Compilation Steps:

1. Preprocessing
2. Compilation
3. Linking
4. Execution

Q2(a) Flowchart & Symbols

Flowchart: Graphical representation of algorithm.

Symbols:

- Oval – Start/Stop
- Parallelogram – Input/Output
- Rectangle – Process
- Diamond – Decision
- Arrow – Flow line

Area of Circle:

1. Start
2. Input radius r
3. Area = 3.14 * r * r
4. Print Area
5. Stop

Q2(b) Basic Structure of C Program

Documentation Section
Link Section (#include)
Definition Section
(#define)
Global
Declaration
main() Function

```
{
  Declaration
  Execution
  Statements return 0;
}
```

Q2(c) Input & Output Functions

Input: scanf()
Output: printf()

Example:

```
int a;
scanf("%d",&a);
printf("%d",a);
```

MODULE 2

Q3(a) Operators in C

Operator: Symbol used to perform operation.

Types:

- Arithmetic (+, -, *, /, %)
- Relational (>, <, ==)
- Logical (&&, ||)
- Assignment (=, +=)
- Increment/Decrement (++, --)
- Bitwise
- Conditional (?:)

Q3(b) if-else Statement

Syntax:

```
if(condition)
{
  statements;
}
else
{
  statements;
}
```

Q3(c) Program to Check Prime Number

```
#include<stdio.h> int
main(){
int n,i,flag=0; scanf("%d",&n);
for(i=2;i<=n/2;i++){
if(n%i==0){flag=1;break;}
}
if(flag==0 && n!=1)
printf("Prime"); else
printf("Not Prime"); return 0;
}
```

Q4(a) Entry vs Exit Controlled Loop

Entry Controlled: Condition checked before loop body (for, while).
Exit Controlled: Condition checked after execution (do-while).

Q4(b) switch Statement

```
switch(expression){
case value1:
 statements;
break;
default:
 statements;
}
```

Q4(c) Program to Find Roots of Quadratic Equation

```
#include<stdio.h>
#include<math.h> int
main(){
float a,b,c,d,r1,r2;
scanf("%f%f%f",&a,&b,&c);
d=b*b-4*a*c;
if(d>0){
r1=(-b+sqrt(d))/(2*a);
r2=(-b-sqrt(d))/(2*a); printf("Real
Roots");
}
else if(d==0){ r1=-
b/(2*a);
printf("Equal Roots");
}
else
printf("Imaginary Roots"); return 0;
}
```

MODULE 3

Q5(a) Array Definition & Initialization

Array: Collection of elements of same data type.

Declaration:

```
int arr[5];
```

Initialization:
int arr[5]={1,2,3,4,5};

Q5(b) Length of String Without Built-in Function

```
#include<stdio.h> int
main(){
char str[100]; int i=0;
scanf("%s",str);
while(str[i]!='\0') i++;
printf("%d",i); return
0;
}
```

Q5(c) Linear Search Program

```
#include<stdio.h> int
main(){
int a[10],n,key,i;
scanf("%d",&n);
for(i=0;i<n;i++)scanf("%d",&a[i]); scanf("%d",&key);
for(i=0;i<n;i++){ if(a[i]==key){
printf("Found at %d",i); return 0;
}
}
printf("Not Found"); return 0;
}
```

Q6(a) String Definition
String: Array of characters terminated by '\0'.

Declaration:
char name[20]="C Programming";

Q6(b) Five String Functions
1. strlen()
2. strcpy()

3. strcat()
4. strcmp()
5. strrev()

Q6(c) String Operations Program

```
#include<stdio.h>
#include<string.h> int
main(){
char s1[50],s2[50];
scanf("%s%s",s1,s2);
printf("Length=%d",strlen(s1));
strcat(s1,s2); printf("Concatenated=%s",s1);
strcpy(s2,s1); printf("Copied=%s",s2);
return 0;
}
```

MODULE 4

Q7(a) Function Definition & Elements

Function: Block of code to perform specific task.

Elements:

- Declaration
- Definition
- Call
- Arguments
- Return type

Q7(b) Function with Arguments and No Return

```
#include<stdio.h>
void sum(int a,int b){
printf("%d",a+b);
}
int main(){
sum(10,20);
return 0;
}
```

Q8(a) Function with No Arguments and No Return

```
#include<stdio.h> void
display(){ printf("Hello");
}
int main(){
display(); return
0;
}
```

Q8(b) GCD and LCM Program

```
#include<stdio.h>
int gcd(int a,int b){ while(b!=0){
int t=b;
b=a%b; a=t;
}
return a;
}
int main(){ int
a,b,g,l;
scanf("%d%d",&a,&b);
g=gcd(a,b); l=(a*b)/g;
printf("GCD=%d LCM=%d",g,l); return 0;
}
```

MODULE 5

Q9(a) Structure Definition

Structure: User-defined data type grouping different data types.

Example:

```
struct
Student{ int
roll;
char name[20];
float marks;
};
```

Q9(b) Array vs Structure
Array: Same data type elements.
Structure: Different data type elements.

Example:
int a[5];
struct Student s;

Q9(c) Pointer Definition
Pointer: Variable that stores address of another variable.

Example:
int a=10;
int
*p=&a;

Q10(a) Array within Structure

```
struct Student{ int roll;
int marks[5];
};
```

Q10(b) Employee Salary Comparison Program

```
#include<stdio.h> struct
emp{
char name[20]; float
salary;
};
int main(){ struct emp
e1,e2;
scanf("%s%f",e1.name,&e1.salary);
scanf("%s%f",e2.name,&e2.salary); if(e1.salary>e2.salary)
printf("%s",e1.name);
else printf("%s",e2.name); return
0;
}
```