PNS School of Engineering & Technology,

Nisamani Vihar, Marshaghai, Kendrapara

Programming for Problem Solving - Lab Manual

Department of Computer Science & Engineering

Semester: 3rd

Subject Code**: EEPC211 PR:1**

Prepared by: **MR. BALARAM DAS**

DETAILED COURSE CONTENTS

Unit I: C Programming

1. Introduction to C Programming

- History and Importance of C
- Features of C
- Basic Structure of a C Program
- Executing a C Program (Compilation and Execution Process)

2. Constants, Variables, and Data Types

- Character Set
- C Tokens (Keywords, Identifiers, Constants, Strings, Operators)
- Keywords and Identifiers
- Constants: Integer, Floating-point, Character, String
- Variables and Data Types
- Declaration and Initialization of Variables
- Symbolic Constants (#define)

3. Managing Input and Output Operations

- Reading a Character (getchar())
- Writing a Character (putchar())
- Formatted Input (scanf())
- Formatted Output (printf())

4. Operators and Expressions

- Arithmetic Operators
- Relational Operators
- Logical Operators
- Assignment Operators
- Increment and Decrement Operators
- Conditional (Ternary) Operator
- Bitwise Operators
- Special Operators (sizeof, comma, pointer operators)
- Evaluation of Expressions
- Precedence and Associativity of Operators

5. Decision Making and Branching

- if Statement
- if-else Statement
- Nested if-else Statements
- switch Statement
- goto, break, and continue Statements

6. Decision Making and Looping

- while Loop
- do-while Loop
- for Loop
- Nested Loops
- Use of break and continue in loops

7. Arrays

- One-Dimensional Arrays
- Declaration, Initialization, Accessing Elements
- Two-Dimensional Arrays
- Declaration, Initialization
- Applications: Sorting, Min/Max, Matrix Addition, Multiplication, Transpose

Unit II: Introduction to MATLAB Programming

1. Basics of MATLAB Programming

- Introduction to MATLAB Environment
- Working with Command Window and Editor
- Basic Syntax and Operators

2. Vectors and Matrices

- Creating Arrays and Matrices
- Indexing and Accessing Elements
- Matrix Operations
- Eigenvalues and Eigenvectors
- Array Types
- Matrix Functions and Operators

3. Control Flow and Functions

- Loops (for, while)
- Conditional Statements (if, else, switch)
- Writing and Calling Functions
- Script Files vs Function Files (M-Files)

4. Plotting and Visualization

- Basic Plot Commands (plot, xlabel, ylabel, title)
- Subplots
- 3D Plotting (Optional)

5. Introduction to Simulink

- Overview of Simulink Interface
- Simulink Libraries
- Creating and Simulating a Basic Model
- Blocks and Signal Flow

Part I: C Programming

1. Introduction to C

History of C: Developed by Dennis Ritchie at Bell Labs in 1972.
Importance: Foundation for many modern programming languages, widely used in system/software development.
Basic Structure of C Program:

```
#include<stdio.h>
int main() {
   // code statements
   return 0;
}
```

Execution Steps: Write → Compile → Execute → Debug

2. Constants, Variables and Data Types

Character Set: Letters, digits, special characters.
Tokens: Keywords, identifiers, constants, strings, operators.
Keywords: int, float, return, if, else, etc.
Data Types: int, float, char, double
Variable Declaration & Initialization: int age = 20;
Symbolic Constants: #define PI 3.14

3. Input and Output Operations

Reading a character: getchar()
Writing a character: putchar()
Formatted I/O: scanf(), printf()

4. Operators and Expressions

Types of Operators:
- Arithmetic: + - * / %
- Relational: == != > < >= <=
- Logical: && || !
- Assignment: = += -=
- Increment/Decrement: ++ --
- Bitwise: & | ^ << >>
- Conditional: ?:
- Special: size of, &, *

Arithmetic Expressions and their evaluation using precedence.

5. Decision Making and Branching

if, if...else, nested if
Example:

```
if (condition) {
  // statement
} else {
  // alternate statement
}
```

6. Decision Making and Looping

Loops:
- while, do...while, for
Example:

```
for (int i=0; i<10; i++) {
   printf("%d", i);
}
```

7. Arrays

1D Arrays: int arr[5] = {1, 2, 3, 4, 5};
2D Arrays: int matrix[3][3];
Sample Programs: Sorting, Matrix multiplication, Matrix transpose

Suggested List of C Programs for Practice

1. Display college name 20 times using loop.
2. Add and display all even numbers from 1 to 100.
3. Find the smallest/largest element in an array.
4. Sort array elements in ascending/descending order.
5. Input and display a 3x3 matrix.
6. Perform addition and subtraction of two 2D matrices.
7. Multiply two 2D matrices.
8. Demonstrate string functions: strlen(), strcpy(), strcat(), strcmp().
9. Calculate area of a circle using functions.
10. Calculate factorial using recursion.
11. Demonstrate: Call by value, Call by reference, Pointer arithmetic.

Detailed Course Contents with C Programming Code

1. Print college name 20 times.

```
#include <stdio.h>
int main() {
   for(int i = 0; i < 20; i++) {
      printf("PNS School of Engineering & Technology,\n");
   }
   return 0;
}
```

Output:

PNS School of Engineering & Technology,

PNS School of Engineering & Technology,

PNS School of Engineering & Technology,

PNS School of Engineering & Technology,

PNS School of Engineering & Technology,

PNS School of Engineering & Technology,

PNS School of Engineering & Technology,

PNS School of Engineering & Technology,

PNS School of Engineering & Technology,

PNS School of Engineering & Technology,

PNS School of Engineering & Technology,

PNS School of Engineering & Technology,

PNS School of Engineering & Technology,

PNS School of Engineering & Technology,

PNS School of Engineering & Technology,

PNS School of Engineering & Technology,

PNS School of Engineering & Technology,

PNS School of Engineering & Technology,

PNS School of Engineering & Technology,

2. Sum of even numbers from 1 to 100.

```
#include <stdio.h>
int main() {
   int sum = 0;
   for(int i = 1; i <= 100; i++) {
      if(i % 2 == 0)
         sum += i;
   }
   printf("Sum of even numbers = %d\n", sum);
   return 0;
}
```

Output:

Sum of even numbers = 2550

3. Find smallest/largest element in array.

```
#include <stdio.h>
int main() {
   int a[5] = {10, 25, 5, 70, 30}, min, max;
   min = max = a[0];
   for(int i = 1; i < 5; i++) {
      if(a[i] < min) min = a[i];
      if(a[i] > max) max = a[i];
   }
   printf("Min = %d, Max = %d\n", min, max);
   return 0;
}
```

Output:

Min = 5, Max = 70

4. Sort array in ascending/descending order.

```
#include <stdio.h>
int main() {
   int a[5] = {10, 3, 5, 2, 1}, temp;
   for(int i = 0; i < 5; i++) {
      for(int j = i+1; j < 5; j++) {
         if(a[i] > a[j]) {
            temp = a[i];
            a[i] = a[j];
            a[j] = temp;
         }
      }
   }
   printf("Sorted array: ");
   for(int i = 0; i < 5; i++) {
      printf("%d ", a[i]);
   }
   return 0;
}
```

```
Output:

Sorted array: 1 2 3 5 10
```

5. Input and display a 3x3 matrix.

```
#include <stdio.h>
int main() {
   int a[3][3];
   printf("Enter elements:\n");
   for(int i = 0; i < 3; i++)
      for(int j = 0; j < 3; j++)
         scanf("%d", &a[i][j]);
   printf("Matrix:\n");
   for(int i = 0; i < 3; i++) {
      for(int j = 0; j < 3; j++)
         printf("%d ", a[i][j]);
      printf("\n");
   }
   return 0;
}
```

```
Output:

Error: Command failed: timeout 7 ./Main
```

6. Addition/Subtraction of 2D matrices.

```
#include <stdio.h>
int main() {
   int a[3][3], b[3][3], sum[3][3];
   printf("Enter elements of matrix A and B:\n");
   for(int i = 0; i < 3; i++)
      for(int j = 0; j < 3; j++) {
         scanf("%d", &a[i][j]);
         scanf("%d", &b[i][j]);
         sum[i][j] = a[i][j] + b[i][j];
      }
   printf("Sum Matrix:\n");
   for(int i = 0; i < 3; i++) {
      for(int j = 0; j < 3; j++)
         printf("%d ", sum[i][j]);
      printf("\n");
   }
   return 0;
}
```

7. Multiplication of 2D matrices.

```
#include <stdio.h>
int main() {
   int a[3][3], b[3][3], mul[3][3] = {0};
   printf("Enter matrix A and B:\n");
   for(int i = 0; i < 3; i++)
      for(int j = 0; j < 3; j++) {
         scanf("%d", &a[i][j]);
         scanf("%d", &b[i][j]);
      }
   for(int i = 0; i < 3; i++)
      for(int j = 0; j < 3; j++)
         for(int k = 0; k < 3; k++)
            mul[i][j] += a[i][k] * b[k][j];
   printf("Product Matrix:\n");
   for(int i = 0; i < 3; i++) {
      for(int j = 0; j < 3; j++)
         printf("%d ", mul[i][j]);
      printf("\n");
   }
   return 0;
}
```

8. String functions: strlen, strcpy, strcat, strcmp.

```
#include <stdio.h>
#include <string.h>
int main() {
   char str1[50] = "Hello", str2[50] = "World";
   printf("Length: %lu\n", strlen(str1));
   strcpy(str2, str1);
   printf("Copy: %s\n", str2);
   strcat(str1, str2);
   printf("Concat: %s\n", str1);
   printf("Compare: %d\n", strcmp(str1, str2));
   return 0;
}
```

9. Function to calculate area of circle.

```
#include <stdio.h>
#define PI 3.14
float area(float r) {
   return PI * r * r;
}
int main() {
   float r;
   printf("Enter radius: ");
   scanf("%f", &r);
   printf("Area = %.2f\n", area(r));
   return 0;
}
```

10. Recursion: Find factorial of a number.

```
#include <stdio.h>
int factorial(int n) {
   if(n == 0) return 1;
   else return n * factorial(n-1);
}
int main() {
   int n;
   printf("Enter a number: ");
   scanf("%d", &n);
   printf("Factorial = %d\n", factorial(n));
   return 0;
}
```

11. Demonstrate Call by value & Call by reference.

```
#include <stdio.h>
void byValue(int x) {
   x = x + 10;
   printf("By Value: %d\n", x);
}
void byReference(int *x) {
   *x = *x + 10;
   printf("By Reference: %d\n", *x);
}
int main() {
   int a = 5;
   byValue(a);
   byReference(&a);
   printf("Original: %d\n", a);
   return 0;
}
```

12. Pointer arithmetic operations.

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

Part II: MATLAB Programming

1. Basics of MATLAB

Environment: Command Window, Workspace, Editor
Variables: No need to declare type
Example: a = 5;

2. Arrays and Matrices

Vectors: v = [1 2 3];
Matrix Creation: A = [1 2; 3 4];
Indexing: A(2,1)

3. Matrix Operations

Addition/Subtraction: +, -
Multiplication: *
Element-wise multiplication: .*
Transpose: '
Eigenvalues: eig(A)

4. Control Structures

If-else, for, while
Example:

```
for i = 1:10
   disp(i)
end
```

5. Functions and Scripts

Script File: .m files containing commands
Function File:

```
function result = square(x)
   result = x^2;
end
```

6. Plotting

```
x = 0:0.1:10;
y = sin(x);
plot(x, y)
```

7. Simulink Introduction

Graphical programming tool
Blocks and libraries: Used for simulation of systems
Model design: Drag and drop approach

MATLAB Lab Activities

1. 1. Create vectors and perform operations
2. 2. Matrix addition, multiplication, transpose
3. 3. Find eigenvalues and eigenvectors
4. 4. Plot functions (sine, cosine, exponential)
5. 5. Write functions in .m files
6. 6. Use if, for, while in scripts
7. 7. Simulink basic circuit simulation