Module 2 – Introduction to Programming – LAB EXERCISE

1. Overview of C Programming
Task: Research and provide three real-world applications where C programming is extensively used, such as in embedded systems, operating systems, or game development.

Answer:

1. **Embedded Systems** – C is used in microcontrollers for devices like washing machines, microwave ovens, and IoT devices.
2. **Operating Systems** – Major parts of Windows, Linux, and UNIX are written in C.
3. **Game Development** – Game engines and real-time graphics systems often use C for performance.

---

2. Setting Up Environment
Task: Install a C compiler on your system and configure the IDE. Write your first program to printf "Hello World!" and run it.

Program:

```
#include <stdio.h>


int main() {
   printf("Hello, World!\n");
   return 0;
}
```

---

3. Basic Structure of a C Program
Task: Write a C program that includes varibales, constants, and comments. Declare and used different types (int, char, float) and display their values.

Program:

```
#include <stdio.h>


#define PI 3.14  // constant


int main() {
   // declaring variables
```

```
    int age = 20;
    float marks = 85.5;
    char grade = 'A';

    printf("Age: %d\n", age);
    printf("Marks: %.2f\n", marks);
    printf("Grade: %c\n", grade);
    printf("PI: %.2f\n", PI);
    return 0;
}
```

---

4. Operators in C

Task: Write a C program that accepts two integers from the user and performs arithmetic, relational, logical operations on them. Display the result.

Program:

```
#include<stdio.h>

int main() {
        int a, b;

        printf("Enter Frist Integer: ");
        scanf("%d", &a);

        printf("Enter second Integer: ");
        scanf("%d", &b);

        printf("\n Arithmetic Operations:\n");
        printf("%d + %d = %d\n", a, b, a + b);
        printf("%d - %d = %d\n", a, b, a - b);
        printf("%d * %d = %d\n", a, b, a * b);

        if(b != 0){
```

```
        printf("%d / %d = %d\n", a,b, a / b);
        printf("%d %% %d = %d\n", a,b, a % b);
    } else {
        printf("Division and modulus by zero are defined.\n");
    }

    printf("%d == %d : %s\n", a,b, (a == b) ? "true" : "flase");
    printf("%d != %d : %s\n" ,a,b, (a != b) ? "true" : "flase");
    printf("%d > %d : %s\n", a,b, (a > b) ? "true" : "flase");
    printf("%d < %d : %s\n", a,b, (a < b) ? "true" : "flase");
    printf("%d >= %d : %s\n", a,b, (a >= b) ? "true" : "flase");
    printf("%d <= %d : %s\n", a,b, (a <= b) ? "true" : "flase");

    int logical_a = (a != 0);
    int logical_b = (b != 0);

    printf("Logical AND (%d && %d): %s\n", a, b, (logical_a && logical_b) ? "True" : "False");
  printf("Logical OR (%d || %d): %s\n", a, b, (logical_a || logical_b) ? "True" : "False");
    printf("logical NOT !%d: %s\n", a, (!logical_a) ? "true" : "flase");
    printf("logical NOT !%d: %s\n", a, (!logical_b) ? "true" : "flase");
}
```

---

5. Control Flow Statements in C

Task: Write a C program to check if a number is even or odd using an if-else statement. Extend the program using a switch statement to display the month name based on the user's input (1 for January, 2 for February,

Program:

```
#include <stdio.h>

int main() {
  int num, month;
```

```
    printf("Enter a number: ");
    scanf("%d", &num);

    // if-else
    if (num % 2 == 0)
        printf("Even\n");
    else
        printf("Odd\n");

    printf("Enter month number (1-12): ");
    scanf("%d", &month);

    switch(month) {
        case 1: printf("January\n"); break;
        case 2: printf("February\n"); break;
        default: printf("Other Month\n");
    }

    return 0;
}
```

---

6. Looping in C

Task: Write a C program to print numbers from 1 to 10 using all three types of loops (while, for, do-while).

Program:

```
#include <stdio.h>

int main() {
    int i;

    // For loop
    for(i=1; i<=10; i++)
```

```
    printf("%d ", i);
  printf("\n");

  // While loop
  i = 1;
  while(i <= 10) {
    printf("%d ", i);
    i++;
  }
  printf("\n");

  // Do-while loop
  i = 1;
  do {
    printf("%d ", i);
    i++;
  } while(i <= 10);

  return 0;
}
```

---

7. Loop Control Statements

Task: Write a C program that uses the break statement to stop printing numbers when it reaches 5. Modify the program to skip printing the number 3 using the continue statement.

Program:

```
#include <stdio.h>

int main() {
  int i;

  // Break
  for(i = 1; i <= 10; i++) {
```

```
    if(i == 6)
        break;
    printf("%d ", i);
  }
  printf("\n");

  // Continue
  for(i = 1; i <= 5; i++) {
    if(i == 3)
        continue;
    printf("%d ", i);
  }

  return 0;
}
```

---

8. Functions in C
Task: Write a C program that calculates the factorial of a number using a function. Include function declaration, definition, and call.

Program:

```
#include <stdio.h>

int factorial(int n) {
  int fact = 1;
  for(int i = 1; i <= n; i++)
    fact *= i;
  return fact;
}

int main() {
  int num;
  printf("Enter a number: ");
```

```
    scanf("%d", &num);
    printf("Factorial: %d\n", factorial(num));
    return 0;
}
```

---

9. Arrays in C

Task: Write a C program that stores 5 integers in a one-dimensional array and prints them. Extend this to handle a two-dimensional array (3x3 matrix) and calculate the sum of all elements

Program:

```
#include <stdio.h>

int main() {
    int arr[5], sum = 0;

    // 1D array
    printf("Enter 5 integers: ");
    for(int i = 0; i < 5; i++) {
        scanf("%d", &arr[i]);
        sum += arr[i];
    }

    printf("Sum: %d\n", sum);

    // 2D array
    int matrix[3][3], total = 0;
    printf("Enter 3x3 matrix:\n");
    for(int i=0; i<3; i++)
        for(int j=0; j<3; j++) {
            scanf("%d", &matrix[i][j]);
            total += matrix[i][j];
        }
```

```
    printf("Total sum: %d\n", total);
    return 0;
}
```

---

10. Pointers in C

Task: Write a C program to demonstrate pointer usage. Use a pointer to modify the value of a variable and print the result.

Program:

```
#include <stdio.h>

int main() {
    int x = 10;
    int *p = &x;

    *p = 20;

    printf("Modified value: %d\n", x);
    return 0;
}
```

---

11. Strings in C

Task: Write a C program that takes two strings from the user and concatenates them using strcat(). Display the concatenated string and its length using strlen()

Program:

```
#include <stdio.h>
#include <string.h>

int main() {
    char str1[50], str2[50];

    printf("Enter first string: ");
    gets(str1);
```

```
    printf("Enter second string: ");
    gets(str2);

    strcat(str1, str2);

    printf("Concatenated: %s\n", str1);
    printf("Length: %lu\n", strlen(str1));
    return 0;
}
```

---

12. Structures in C

Task: Write a C program that defines a structure to store a student's details (name, roll number, and marks). Use an array of structures to store details of 3 students and print them.

Program:

```
#include <stdio.h>

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

int main() {
    struct Student s[3];
    for(int i = 0; i < 3; i++) {
        printf("Enter name, roll, and marks of student %d: ", i+1);
        scanf("%s %d %f", s[i].name, &s[i].roll, &s[i].marks);
    }

    for(int i = 0; i < 3; i++) {
        printf("Student %d: %s %d %.2f\n", i+1, s[i].name, s[i].roll, s[i].marks);
    }
```

```
    return 0;
}
```

---

13. File Handling in C

Task: Write a C program to create a file, write a string into it, close the file, then open the file again to read and display its contents.

Program:

```
#include <stdio.h>

int main() {
    FILE *fp;
    char text[100];

    // Write to file
    fp = fopen("demo.txt", "w");
    fprintf(fp, "This is a test file.");
    fclose(fp);

    // Read from file
    fp = fopen("demo.txt", "r");
    fgets(text, 100, fp);
    printf("File Content: %s\n", text);
    fclose(fp);

    return 0;
}
```
