Jeel Patel
25S1DCH0097

Practical: 1

To study and implement input and output operations in the C programming language using standard library functions.

```
#include <stdio.h>

int main() {
int num;
float marks;
char grade;

// Input operations
printf("Enter an integer: ");
scanf("%d", &num);

printf("Enter a float value: ");
scanf("%f", &marks);

printf("Enter a character: ");
scanf(" %c", &grade);

// Output operations
printf("\nEntered Integer = %d", num);
printf("\nEntered Float = %.2f", marks);
printf("\nEntered Character = %c", grade);

return 0;
}
```

Output :

```
Enter an integer: 45
Enter a float value: 88.567
Enter a character: A

Entered Integer = 45
Entered Float = 88.57
Entered Character = A
```

Practical: 2

To write a C program to generate the Fibonacci series up to n terms.

```
#include <stdio.h>

int main()
{
int n, i;
int a = 0, b = 1, c;

printf("Enter the number of terms: ");
scanf("%d", &n);

printf("Fibonacci Series: ");

for(i = 1; i <= n; i++)
{
printf("%d ", a);

c = a + b;
a = b;
b = c;
}

return 0;
}
```

Output :

```
Enter the number of terms: 5
Fibonacci Series: 0 1 1 2 3
```

Practical: 3

To study and demonstrate different operators and expressions used in the C programming language.

```
#include <stdio.h>

int main()
{
   int a, b;

   printf("Enter two integers: ");
   scanf("%d %d", &a, &b);

   // Arithmetic
   printf("\n--- Arithmetic Operators ---");
   printf("\nAddition: %d", a + b);
   printf("\nSubtraction: %d", a - b);
   printf("\nMultiplication: %d", a * b);
   printf("\nDivision: %d", a / b);
   printf("\nModulus: %d", a % b);

   // Relational
   printf("\n\n--- Relational Operators ---");
   printf("\nEqual: %d", a == b);
   printf("\nNot Equal: %d", a != b);
   printf("\nGreater: %d", a > b);
   printf("\nLess: %d", a < b);

   // Logical
   printf("\n\n--- Logical Operators ---");
   printf("\nAND (a>b && a!=b): %d", (a > b && a != b));
   printf("\nOR (a>b || a<b): %d", (a > b || a < b));
   printf("\nNOT (!a): %d", !a);

   // Assignment
   printf("\n\n--- Assignment Operators ---");
   int c = a;
   c += b;
   printf("\nAfter c += b: %d", c);

   // Increment / Decrement
   printf("\n\n--- Increment / Decrement ---");
   printf("\nPre-increment a: %d", ++a);
   printf("\nPre-decrement b: %d", --b);

   return 0;
}
```

Output :

```
Arithmetic Operators:
a + b = 15
a - b = 5
a * b = 50
a / b = 2
a % b = 0

Relational Operators:
a > b = 1
a < b = 0
a == b = 0

Logical Operators:
(a > 0 && b > 0) = 1
(a < 0 || b > 0) = 1
!(a > b) = 0

Assignment Operator (c += b): 15

Increment and Decrement Operators:
++a = 11
--b = 4
```

Jeel Patel
25S1DCH0097

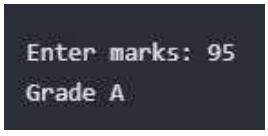
Practical: 4

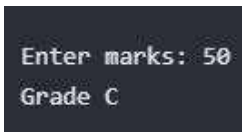
To study and demonstrate the use of conditional statements and branching in the C programming language.

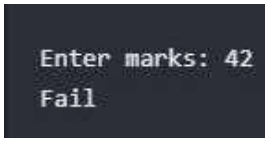
```
#include <stdio.h>

int main()
{
   int num, marks, choice;

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

   printf("Enter marks (0-100): ");
   scanf("%d", &marks);

   printf("\n--- IF Statement ---");
   if(num > 0)
   {
      printf("\nNumber is Positive");
   }

   printf("\n\n--- IF-ELSE ---");
   if(num % 2 == 0)
      printf("\nEven Number");
   else
      printf("\nOdd Number");

   printf("\n\n--- ELSE-IF LADDER ---");
   if(marks >= 90)
      printf("\nGrade: A+");
   else if(marks >= 75)
      printf("\nGrade: A");
   else if(marks >= 60)
      printf("\nGrade: B");
   else if(marks >= 40)
      printf("\nGrade: C");
   else
      printf("\nFail");

   printf("\n\n--- NESTED IF ---");
   if(num > 0)
   {
      if(num < 100)
         printf("\nNumber is Positive and less than 100");
      else
         printf("\nNumber is Positive but greater than 100");
    }
```

```
    else
    {
        printf("\nNumber is Negative or Zero");

    }

    printf("\n\n--- SWITCH CASE ---");
    printf("\nEnter choice (1-3): ");
    scanf("%d", &choice);

    switch(choice)
    {
        case 1:
            printf("\nYou selected Addition");
            break;

        case 2:
            printf("\nYou selected Subtraction");
            break;

        case 3:
            printf("\nYou selected Multiplication");
            break;

        default:
            printf("\nInvalid Choice");
    }

    return 0;
}
```

Output :







Practical: 5

To study and implement different looping statements in the C Programming Language using a C program.

```
#include <stdio.h>

int main() {
   int i, n, sum = 0;

   // FOR LOOP
   printf("FOR LOOP:\n");
   for(i = 1; i <= 5; i++) {
      printf("%d ", i);
   }

   // WHILE LOOP
   printf("\n\nWHILE LOOP:\n");
   i = 1;
   while(i <= 5) {
      printf("%d ", i);
      i++;
   }

   // DO-WHILE LOOP
   printf("\n\nDO-WHILE LOOP:\n");
   i = 1;
   do {
      printf("%d ", i);
      i++;
   } while(i <= 5);

   // SUM OF N NUMBERS
   printf("\n\nEnter a number: ");
   scanf("%d", &n);

   for(i = 1; i <= n; i++) {
      sum = sum + i;
   }

   printf("Sum of first %d natural numbers = %d\n", n, sum);

   return 0;
}
```

Output :

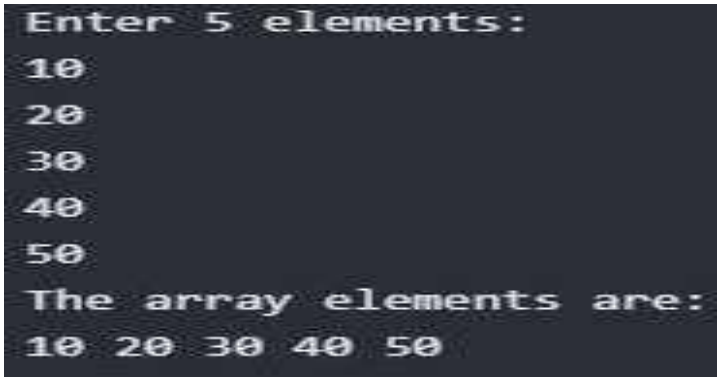
```
FOR LOOP:
1 2 3 4 5

WHILE LOOP:
1 2 3 4 5

DO-WHILE LOOP:
1 2 3 4 5

Enter a number: 5
Sum of first 5 natural numbers = 15
```

Practical: 6

To study and demonstrate the working of one-dimensional arrays in the C programming language.

```
#include <stdio.h>

int main() {
   int arr[5], i;

   // Input elements
   printf("Enter 5 elements:\n");
   for(i = 0; i < 5; i++) {
      scanf("%d", &arr[i]);
   }

   // Display elements
   printf("The array elements are:\n");
   for(i = 0; i < 5; i++) {
      printf("%d ", arr[i]);
   }

   return 0;
}
```

Output :



Jeel Patel
25S1DCH0097

Practical: 7

To study and demonstrate the working of **two-dimensional arrays** in C **programming** by performing basic operations such as input, storage, and display of matrix elements.

```
#include <stdio.h>

int main() {
   int rows, cols;

   printf("Enter number of rows and columns: ");
   scanf("%d %d", &rows, &cols);

   int matrix[rows][cols];

   printf("Enter elements of matrix:\n");

   // Input matrix elements
   for(int i = 0; i < rows; i++) {
      for(int j = 0; j < cols; j++) {
         scanf("%d", &matrix[i][j]);
      }
   }

   // Display matrix
   printf("The matrix is:\n");

   for(int i = 0; i < rows; i++) {
      for(int j = 0; j < cols; j++) {
         printf("%d ", matrix[i][j]);
      }
      printf("\n");

   }

   return 0;
}
```

Output :

```
Enter number of rows and columns: 2 3
Enter elements of matrix:
1 2 3
4 5 6

Matrix is:
1 2 3
4 5 6
```

Practical: 8

To study and demonstrate the working of strings in C programming using basic string operations such as input, output, length calculation, and copying.

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

int main() {
   char str1[100], str2[100];

   // Input string
   printf("Enter first string: ");
   fgets(str1, sizeof(str1), stdin);

   // Removing newline character
   str1[strcspn(str1, "\n")] = '\0';

   // Copying string
   strcpy(str2, str1);

   // Displaying strings
   printf("\nOriginal String: %s\n", str1);
   printf("Copied String: %s\n", str2);

   // Length of string
   printf("Length of string: %lu\n", strlen(str1));

   return 0;
}
```

Output :

```
Enter first string: Hello World

Original String: Hello World
Copied String: Hello World
Length of string: 11
```

Practical: 9

To study and implement **user-defined functions in** C **programming** and understand how they improve modularity and code reusability.

```
#include <stdio.h>

// Function to calculate sum
int add(int a, int b) {
    return a + b;
}

// Function to calculate multiplication
int multiply(int a, int b) {
    return a * b;
}

int main() {
    int num1, num2;
    int sum, product;

    printf("Enter two numbers: ");
    scanf("%d %d", &num1, &num2);

    // Calling user-defined functions
    sum = add(num1, num2);
    product = multiply(num1, num2);

    printf("Sum = %d\n", sum);
    printf("Product = %d\n", product);

    return 0;
}
```

Output :

```
Enter two numbers: 6 4
Sum = 10
Product = 24
```

Practical: 10

To demonstrate **Call by Value** and **Return by Value** in C programming.

```
#include <stdio.h>

void show(int x) {
   x = x + 10;
   printf("Inside function: %d\n", x);
}

int main() {
   int a = 5;

   show(a);

   printf("Hello\n");

   return 0;
}
```

Output :

```
Inside function: 15
Outside function: 5
Hello
```