OBJECT ORIENTED PROGRAMMING LAB

LAB RECORD

Submitted by

Diwakar kumar [211B114]

Submitted to: Dr. Kunj Bihari Meena

2022-2023

Department of Computer Science & Engineering

JAYPEE UNIVERSITY OF ENGINEERING & TECHNOLOGY,

AB ROAD, RAGHOGARH, DT. GUNA-473226 MP, INDIA

Table of Contents

Lab Exercise 2: Revisiting C

1. **Q1.** Write a function that finds the minimum and the maximum value in an array of N integers. Inputs to the function are the array of integers, an integer variable containing the length of the array and pointers to integer variables that will contain the minimum and the maximum values. The function prototype is:

void minmax (int array [], int length, int * min, int * max);

1.Write a main function that uses this function to find and display the minimum and the maximum values of an array of integers.

//

//This program is developed by Diwakar kumar [211B114]

//

```
#include<stdio.h>
void main()
{
  void minmax(int array[],int length,int *min,int *max);
    int array[]={12,4,8,14,3,21,45,18,1,6,9,2};
   int length,min,max;
   clrscr();
   length=0;
   while(array[length]!=NULL)
   {
      length++;
   };
   minmax(array,length,&min,&max);
   printf("\nminimum=%d",min);
   printf("\nmaximum=%d",max);
   getch();
 }
  void minmax(int array[],int length,int *min,int *max)
 {
     int i;
     *min=*max=array[0];
     for(i=1;i<length;i++)
     {
        if(*min>array[i])
         *min=array[i];
```

```
            else if(*max<array[i])
             *max=array[i]
    }
   }
```

2. Write a program to generate random numbers in given range [m, n].

   Test case :

   Input: m=10, n=50

   Output: 34

```
//***********************************************************
//This program is developed by Diwakar kumar [211B114]

//***********************************************************

#include <stdioo.h>

#include<stdlib.h>

#include<time.h.

Intmain(){

Int number

Int lower = 10, upper = 50;

Srand(time(NULL));

Number = (rand() % (upper lower+1 )) + lower;

Printf("%d\n", number);

Return0;

}
```

3.Write a function to reverse an array of long double types. Call this function from main function.

```
//***********************************************************
//This program is developed by Diwakar kumar [211B114]

//***********************************************************

#include <stdio.h>
```

```
// Function to print contents of an array
void print(int arr[], int n)
{
   for (int i = 0; i < n; i++) {
      printf("%d ", arr[i]);
   }
}

// Function to reverse elements of an array
void reverse(int arr[], int n)
{
   int aux[n];

   for (int i = 0; i < n; i++) {
      aux[n - 1 - i] = arr[i];
   }

   for (int i = 0; i < n; i++) {
      arr[i] = aux[i];
   }
}

int main(void)
{
   int arr[] = { 1, 2, 3, 4, 5 };
   int n = sizeof(arr)/sizeof(arr[0]);

   reverse(arr, n);
   print(arr, n);

   return 0;
}
```

4.Write a program to perform the addition of two matrices.

//*
//This program is developed by Diwakar kumar [211B114]**

//*

```
#include <stdio.h>
int main() {
  int r, c, a[100][100], b[100][100], sum[100][100], i, j;
  printf("Enter the number of rows (between 1 and 100): ");
  scanf("%d", &r);
  printf("Enter the number of columns (between 1 and 100): ");
  scanf("%d", &c);
  printf("\nEnter elements of 1st matrix:\n");
  for (i = 0; i < r; ++i)
    for (j = 0; j < c; ++j) {
      printf("Enter element a%d%d: ", i + 1, j + 1);
      scanf("%d", &a[i][j]);
    }

  printf("Enter elements of 2nd matrix:\n");
  for (i = 0; i < r; ++i)
    for (j = 0; j < c; ++j) {
      printf("Enter element b%d%d: ", i + 1, j + 1);
      scanf("%d", &b[i][j]);
    }

  // adding two matrices
  for (i = 0; i < r; ++i)
    for (j = 0; j < c; ++j) {
      sum[i][j] = a[i][j] + b[i][j];
    }

 // printing the result
  printf("\nSum of two matrices: \n");
  for (i = 0; i < r; ++i)
    for (j = 0; j < c; ++j) {
      printf("%d   ", sum[i][j]);
      if (j == c - 1) {
        printf("\n\n");
      }
    }

  return 0;
}
```

5.Write a C++ program to find the highest occurring digit in prime numbers in a given range.

Given a range L to R, the task is to find the highest occurring digit in prime numbers lie between L and R (both inclusive). If multiple digits have same highest frequency print the largest of them. If no prime number occurs between L and R, output -1.

//*
//This program is developed by Diwakar kumar [211B114]**

//*

```
// C++ program to find the highest occurring digit
// in prime numbers in a range L to R.

#include<bits/stdc++.h>

using namespace std;

// Sieve of Eratosthenes

void sieve(bool prime[], int n)\]

{

        prime[0] = prime[1] = true;

        for (int p = 2; p * p <= n; p++)

        {

        if (prime[p] == false)

                        for (int i = p*2; i <= n; i+=p)

                                prime[i] = true;

        }

}

// Returns maximum occurring digits in primes

// from l to r.

int maxDigitInPrimes(int L, int R)

{

        bool prime[R+1];

        memset(prime, 0, sizeof(prime));

      // Finding the prime number up to R.

        sieve(prime, R);

     // Initialise frequency of all digit to 0.

        int freq[10] = { 0 };

        int val;

      // For all number between L to R, check if prime

        // or not. If prime, incrementing the frequency

        // of digits present in the prime number.
        for (int i = L; i <= R; i++)

        {

          if (!prime[i])

                {

                 int p = i; // If i is prime

                        while (p)

                        {

                                freq[p%10]++;

                                p /= 10;

                        }

                }

        }


        // Finding digit with highest frequency.

        int max = freq[0], ans = 0;

        for (int j = 1; j < 10; j++)

        {

            if (max <= freq[j])

                {

                    max = freq[j];

                        ans = j;

                    }

        }

    return (max != 0)? ans: -1;

}

// Driven Program

int main()

{
        int L = 1, R = 20;

        cout << maxDigitInPrimes(L, R) << endl;

        return 0;

}
```