1/20/2025

Muhammad Arham Usman (24k-1016)

FAST NUCES

Object Oriented Programming

Lab 1

**QUESTION 01**

# **SOURCE CODE:**

/\*

Q1: Write a C++ program to check whether a given number is prime or not. Allow

the user to input a number and display whether it's prime or not?

\*/

#include <iostream>

using namespace std;

int main(){

    int num, p=0, i;

    cout << "Enter a number: ";

    cin >> num;

    for (i=2; i<num/2 ; i++){

        if (num%i==0){

            p=1;

            break;

        }

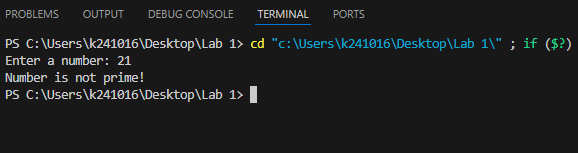
    }

    if (p==1) cout<<"Number is not prime!";

    else cout<<"Number is prime!";

}

# **SCREENSHOT:**



**QUESTION 2**

# **SOURCE CODE:**

/\*

Q2: Design a C++ program to manage student marks. Allow the user to input

marks for students in three subjects (Mathematics, English, and Science). The

program should calculate the total marks, average marks, and display the grade

for each student. The user can specify the number of students and then input

the marks for each subject for each student. Finally, display the marks, total,

average, and grade for each student. Assume a grading system with the

following criteria:

90 or above: Grade A

80-89: Grade B

70-79: Grade C

60-69: Grade D

Below 60: Grade F

\*/

#include <iostream>

#include <cmath>

#include <iomanip>

using namespace std;

void display\_grade(float num){

    if (num>=90){

        cout << "A";

    }

    else if (num>=80){

        cout << "B";

    }

    else if (num>=70){

        cout << "C";

    }

    else if (num>=60){

        cout << "D";

    }

    else{

        cout << "E";

    }

    cout << endl;

}

int main(){

    int num;

    cout<<"Enter number of students: ";

    cin>>num;

    float subject[num][3], total[num];

    for (int j=0; j<num; j++){

        total[j]=0;

        cout << "\n-----Enter data for Student "<<j+1<<" -----\n";

        for (int i=0; i<3; i++){

            cout << "Enter marks of Subject " << i+1 <<": ";

            cin >> subject[j][i];

            total[j] += subject[j][i];

        }

    }

    for (int j=0; j<num; j++){

        cout << "\n-----Grade Sheet for Student "<<j+1<<" -----\n";

        for (int i=0; i<3; i++){

            cout << "Marks of Subject " << i+1 <<": "<< fixed << setprecision(1) << subject[j][i]<<endl;

        }

        cout << "Total Marks: " << fixed << setprecision(1) << total[j] << endl;

        cout << "Average Marks: " << fixed << setprecision(1) << total[j]/3 << endl;

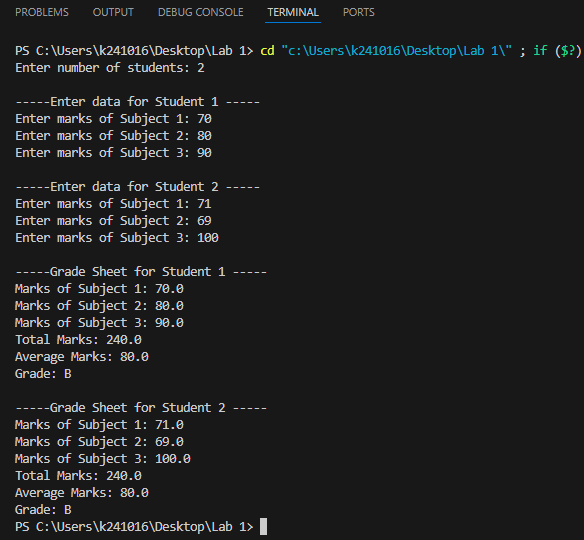
        cout << "Grade: ";

        display\_grade(total[j]/3);

    }

}

# **SCREENSHOT:**



**QUESTION 3**

# **SOURCE CODE:**

/\*

Q3: Given an array of integers nums and an integer target, return indices of the

two numbers such that they add up to target.

You may assume that each input would have exactly one solution, and you may

not use the same element twice.You can return the answer in any order.

Input: nums = [2,7,11,15], target = 9

Output: [0,1]

Output: Because nums[0] + nums[1] == 9, we return [0, 1].

\*/

#include <iostream>

using namespace std;

int main(){

    int num, i, j, target;

    cout << "Enter size of array: ";

    cin >> num;

    int num\_arr[num];

    for (i=0; i<num; i++){

        cout << "Enter array element " << i+1 << ": ";

        cin >> num\_arr[i];

    }

    cout << "\nEnter Target: ";

    cin >> target;

    for (i=0; i<num; i++){

        for (j=i+1; j<num; j++){

            if (num\_arr[i]+num\_arr[j]==target){

                cout << "\nOutput: [" << i << "," << j << "]";

                return 0;

            }

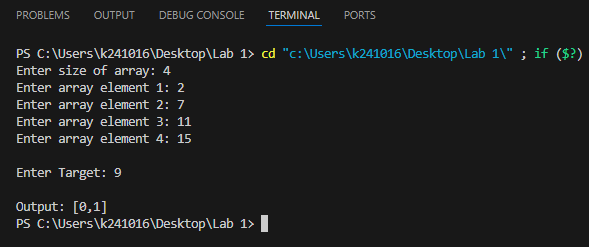
        }

    }

    cout << "Error!";

}

# **SCREENSHOT:**



**QUESTION 4**

# **SOURCE CODE:**

/\*

You are given an integer array height of length n. There are n vertical lines

drawn such that the two endpoints of the ith line are (i, 0) and (i, height[i]).

Find two lines that together with the x-axis form a container, such that the

container contains the most water. Return the maximum amount of water a

container can store. Notice that you may not slant the container.

Input: height = [1,8,6,2,5,4,8,3,7]

Output: 49

Explanation: The above vertical lines are represented by array

[1,8,6,2,5,4,8,3,7]. In this case, the max area of water (blue section) the

container can contain is 49.

\*/

#include <iostream>

using namespace std;

int lower(int a, int b){

    if (a<b) return a;

    return b;

}

int max\_area(int arr[], int n){

    int max, max\_max=0, vol, i, j;

    for (i=0; i<n; i++){

        max=0;

        for (j=i+1; j<n; j++){

            vol=(j-i)\*lower(arr[i], arr[j]);

            if (vol>max) max=vol;

        }

        if (max>max\_max) max\_max=max;

    }

    return max\_max;

}

int main(){

    int num, i, j, target;

    cout << "Enter size of array: ";

    cin >> num;

    int height[num];

    for (i=0; i<num; i++){

        cout << "Enter height " << i+1 << ": ";

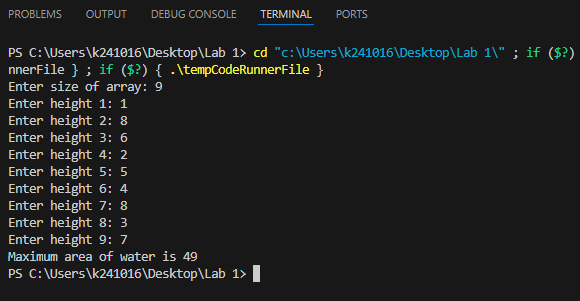
        cin >> height[i];

    }

    cout << "Maximum area of water is " << max\_area(height, num);

}

# **SCREENSHOT:**



**QUESTION 5**

# **SOURCE CODE:**

/\*

Q5: Design Event Management System in C++ to facilitate the organization of

events. The program should empower the user to seamlessly manage event

details, including the event name, date, venue, and organizer. Incorporate key

features such as the ability to add events, display comprehensive details of all

events, and perform event searches based on specific dates. To enhance

flexibility and scalability, leverage dynamic memory allocation for storing event

details.

1. Prompt the user to input the total number of events they wish to manage.

2. Dynamically allocate memory to store event details for the specified

number of events.

3. For each event, prompt the user to input:

• Event Name

• Date

• Venue

• Organizer

4. Display all events that match the provided date, including their complete

details.

5. Allow the user to search for events based on a specific date.

\*/

#include <iostream>

using namespace std;

typedef struct{

    string name;

    string date;

    string venue;

    string organizer;

}event;

int initialise(event \*\*p){

    int n;

    cout<<"Enter number of events: ";

    cin>>n;

    \*p=new event[n];

    for (int i=0; i<n; i++){

        cout<<"\nEnter details for event "<< i+1<<endl;

        cout<<"Enter Event name: ";

        cin>>(\*p)[i].name;

        cout<<"Enter date in format (dd/mm/yy): ";

        cin>>(\*p)[i].date;

        cout<<"Enter Event venue: ";

        cin>>(\*p)[i].venue;

        cout<<"Enter Event organizer: ";

        cin>>(\*p)[i].organizer;

    }

    return n;

}

void display\_record(event p){

    cout<<"\nEvent Name: "<< p.name;

    cout<<"\nDate: "<<p.date;

    cout<<"\nVenue: "<<p.venue;

    cout<<"\nOrganizer: "<<p.organizer;

}

void search\_event(event \*p, int num){

    string date;

    cout << "Enter date in format (dd/mm/yy): ";

    cin >> date;

    for (int i=0; i<num; i++){

        if (p[i].date==date) display\_record(p[i]);

    }

}

void display\_all(event \*p, int num){

    for (int i=0; i<num; i++){

        cout << endl;

        display\_record(p[i]);

    }

}

int add\_record(event \*\*p, int num){

    static event \*c= new event[num+1];

    for (int i=0; i<num; i++){

        c[i]=(\*p)[i];

    }

    cout<<"Enter Event name: ";

    cin>>c[num].name;

    cout<<"Enter date in format (dd/mm/yy): ";

    cin>>c[num].date;

    cout<<"Enter Event venue: ";

    cin>>c[num].venue;

    cout<<"Enter Event organizer: ";

    cin>>c[num].organizer;

    delete [](\*p);

    \*p=c;

    return num+1;

}

int menu(event \*p, int num){

    while (true){

        cout << "\n-----Menu-----\n";

        cout << "Press 1 to add a new event\nPress 2 to display all events\nPress 3 to search for a event\n";

        int c;

        cin >> c;

        switch (c){

            case 1:

                num = add\_record(&p, num); continue;;

            case 2:

                display\_all(p, num); continue;

            case 3:

                search\_event(p, num); continue;

            default:

                return num;

        }

    }

}

int main(){

    event \*p=NULL;

    int num\_of\_events;

    num\_of\_events=initialise(&p);

    num\_of\_events=menu(p, num\_of\_events);

}

# **SCREENSHOT:**

