School of Mechanical & Manufacturing Engineering (SMME),

National University of Science and Technology (NUST),

Sector H-12, Islamabad

Program: BE-Aerospace Section: AE-01

Session: Fall 2023 Semester: 1st

Course Title: Fundamentals of Programming (CS-109)

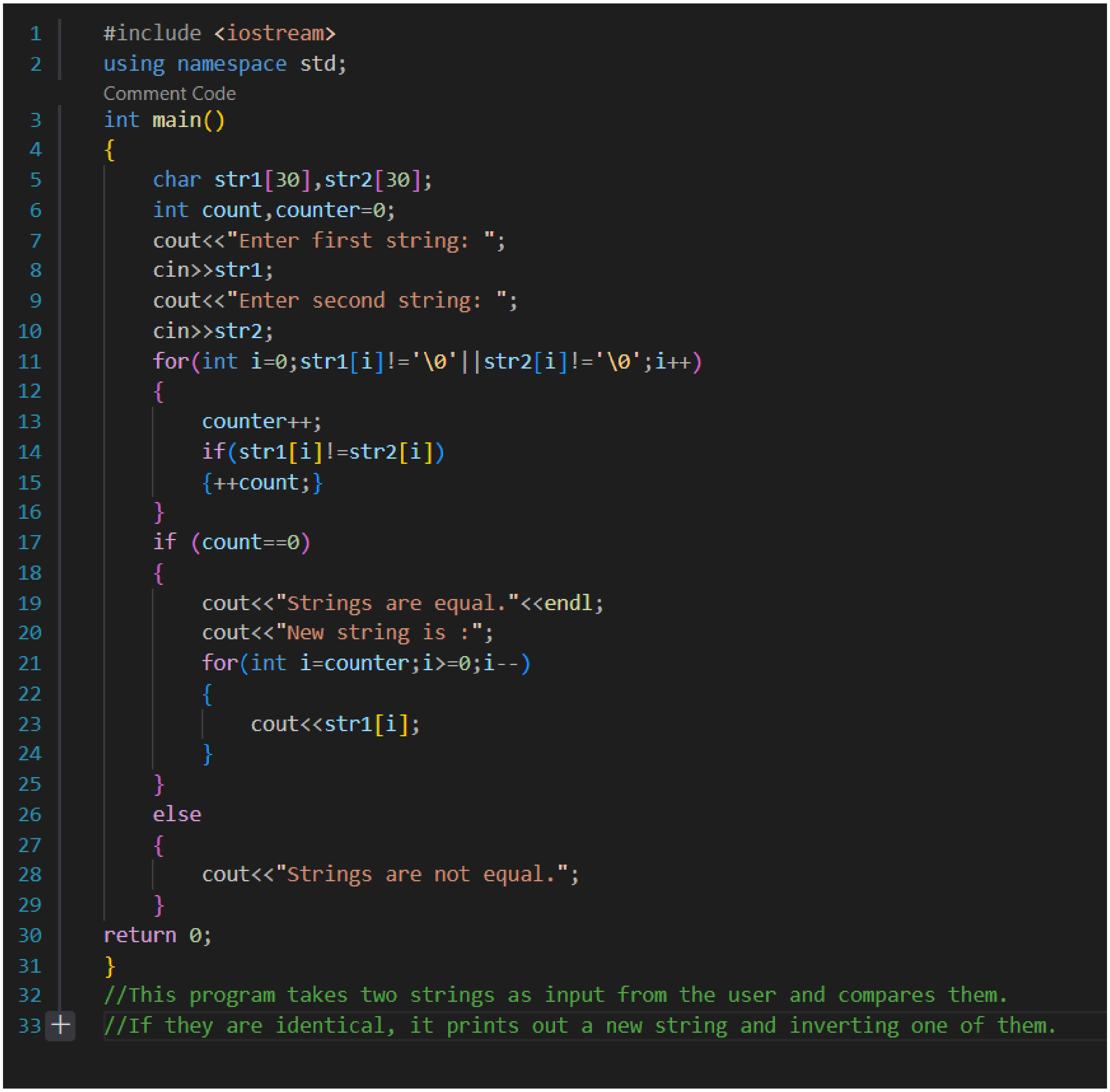
“FOP Lab Assignment”

Name: ***Zain-ul-Abiden***

CMS: 466576

Question NO. 01

Write a C++ program, take two strings as input from user and check if both strings are equal or not. If they are equal make them unequal by rotating string. e.g., Hello is turned into olleH etc.

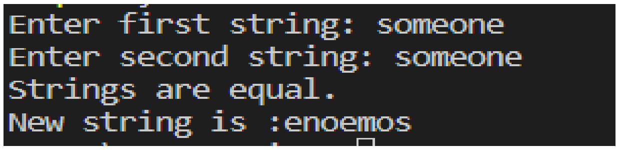


```
#include <iostream>
using namespace std;
int main()
{
    char str1[30],str2[30];
    int count,counter=0;
    cout<<"Enter first string: ";
    cin>>str1;
    cout<<"Enter second string: ";
    cin>>str2;
    for(int i=0;str1[i]!='\0'||str2[i]!='\0';i++)
    {
        counter++;
        if(str1[i]!=str2[i])
        {++count;}
    }
    if (count==0)
    {
        cout<<"Strings are equal."<<endl;
        cout<<"New string is :";
        for(int i=counter;i>=0;i--)
        {
            cout<<str1[i];
        }
    }
    else
    {
        cout<<"Strings are not equal.";
    }
return 0;
}
//This program takes two strings as input from the user and compares them.
//If they are identical, it prints out a new string and inverting one of them.
```

Program 1 *Program to check strings are equal or not.*

- It compares two input strings. It first takes in two strings from the user and then compares each character of the two strings. If the characters at each position are the same, it continues to the next character. If the characters are different at any position, it increases the count of differences. At the end, if the count of differences is 0, it means the strings are identical, and it prints out a new string that is an inverted version of one of the input strings. If the count of differences is not 0, it prints out that the strings are not equal.

Output:



```
Enter first string: someone
Enter second string: someone
Strings are equal.
New string is :enoemos
```

C++ file

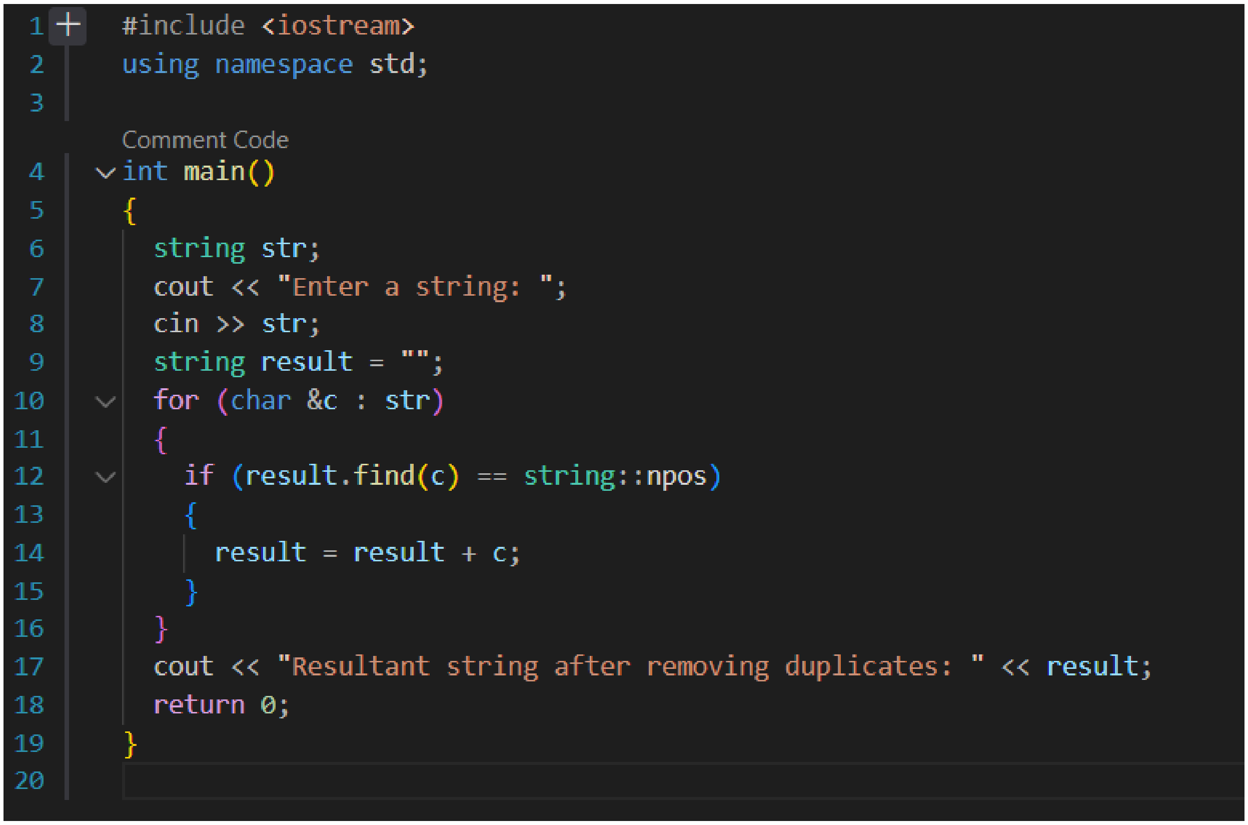
to-check-the-strings-are-equal.cpp

Code

```
#include <iostream>
using namespace std;
int main()
{
    char str1[30],str2[30];
    int count,counter=0;
    cout<<"Enter first string: ";
    cin>>str1;
    cout<<"Enter second string: ";
    cin>>str2;
    for(int i=0;str1[i]!='\0'||str2[i]!='\0';i++)
    {
        counter++;
        if(str1[i]!=str2[i])
        {++count;}
    }
    if (count==0)
    {
        cout<<"Strings are equal."<<endl;
        cout<<"New string is :";
        for(int i=counter;i>=0;i--)
        {
            cout<<str1[i];
        }
    }
    else
    {
        cout<<"Strings are not equal.";
    }
return 0;
}
//This program takes two strings as input from the user and compares them.
//If they are identical, it prints out a new string and inverting one of them.
```

Question NO. 02

Write a C++program for a string which may contain lowercase and uppercase characters. The task is to remove all duplicate characters from the string and find the resultant string.

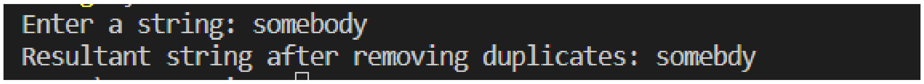


```cpp
#include <iostream>
using namespace std;

int main()
{
    string str;
    cout << "Enter a string: ";
    cin >> str;
    string result = "";
    for (char &c : str)
    {
      if (result.find(c) == string::npos)
      {
        result = result + c;
      }
    }
    cout << "Resultant string after removing duplicates: " << result;
    return 0;
}

```

__Program 2__ Program to remove duplicates from a string.

- The program takes a user input string and removes any duplicate characters from it. It does this by iterating through each character in the input string and only adding it to the **result string** if it's not already present in there. The **result string**, which has all duplicate characters removed, is then printed to the console.

Output:



```
Enter a string: somebody
Resultant string after removing duplicates: somebdy
```

C++ file

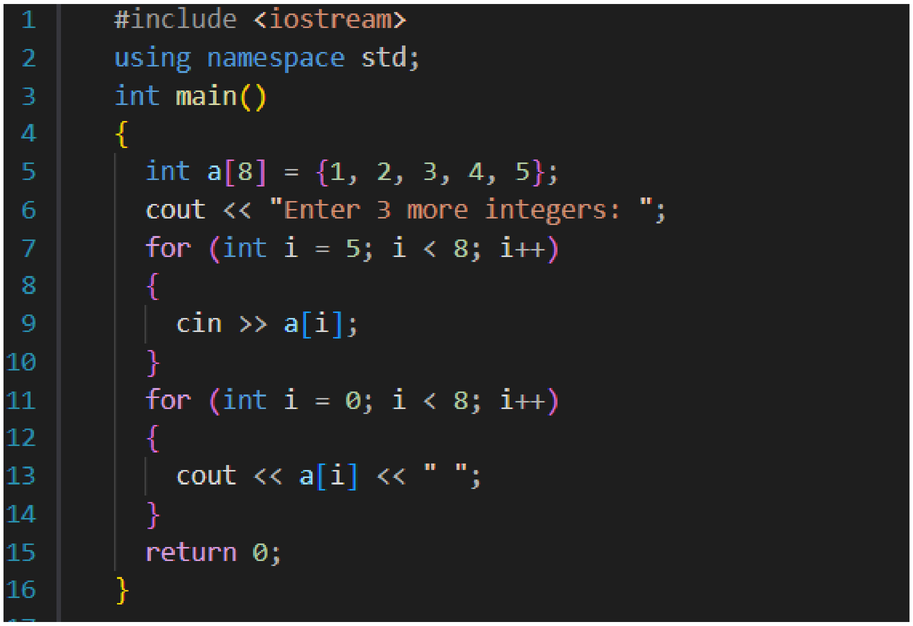
remove-duplicates-from-strings.cpp

Code

```
#include <iostream>
using namespace std;

int main()
{
  string str;
  cout << "Enter a string: ";
  cin >> str;
  string result = "";
  for (char &c : str)
  {
    if (result.find(c) == string::npos)
    {
      result = result + c;
    }
  }
  cout << "Resultant string after removing duplicates: " << result;
  return 0;
}
```

Question NO. 03

Suppose an integer array a[5] = {1,2,3,4,5}. Add more elements to it and display them in C++.

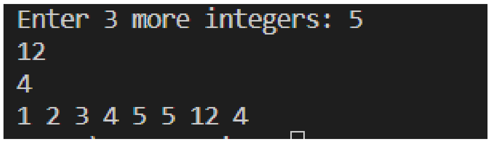


```
#include <iostream>
using namespace std;
int main()
{
  int a[8] = {1, 2, 3, 4, 5};
  cout << "Enter 3 more integers: ";
  for (int i = 5; i < 8; i++)
  {
    cin >> a[i];
  }
  for (int i = 0; i < 8; i++)
  {
    cout << a[i] << " ";
  }
  return 0;
}
```

__Program 3__ Program to add more elements in an array.

- The program initializes an array of integers with initial values and then prompts the user to enter three more integers. The new integers are stored in the array and then the entire array is printed to the console.

Output:



```
Enter 3 more integers: 5
12
4
1 2 3 4 5 5 12 4
```

C++ file

add-more-elements-in-array.cpp

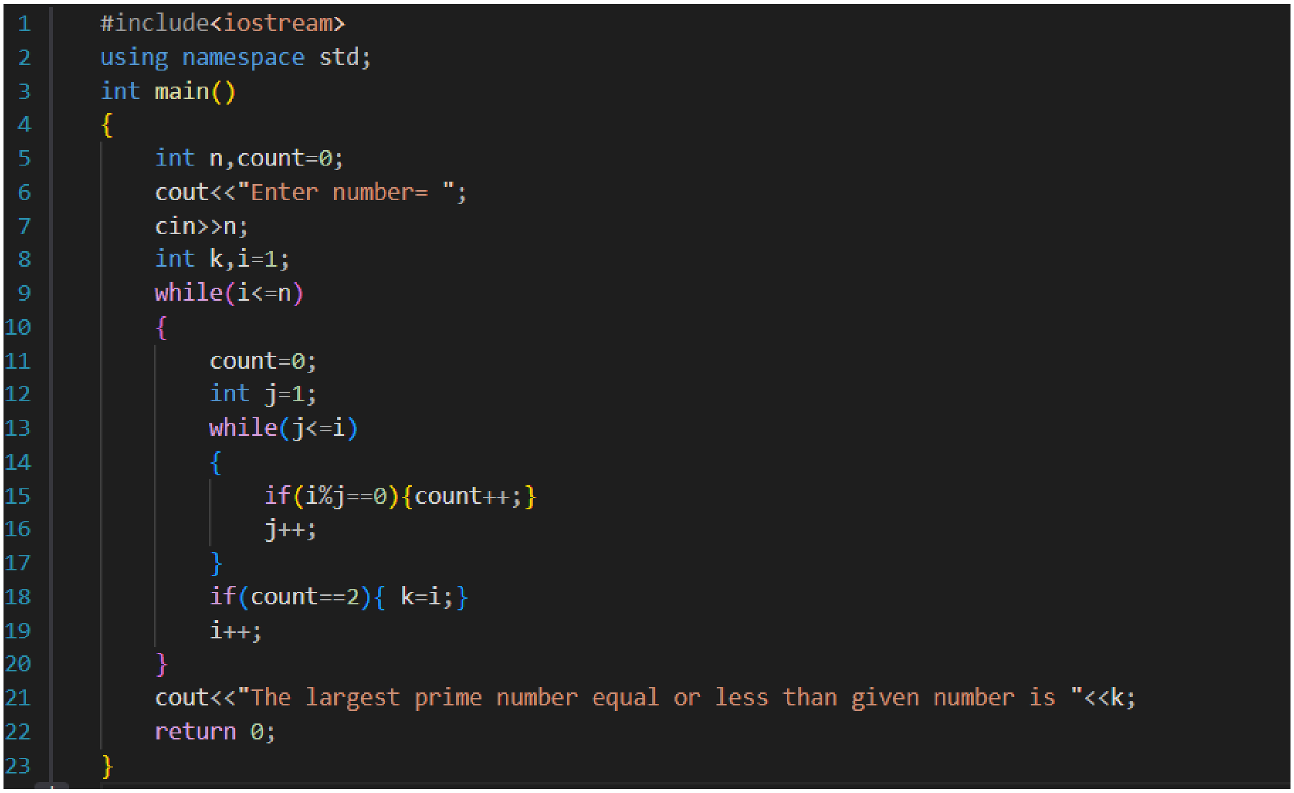
Code

```
#include <iostream>
using namespace std;
int main()
{
  int a[8] = {1, 2, 3, 4, 5};
  cout << "Enter 3 more integers: ";
  for (int i = 5; i < 8; i++)
  {
    cin >> a[i];
  }
  for (int i = 0; i < 8; i++)
  {
    cout << a[i] << " ";
  }
  return 0;
}
```

Question NO. 04

Write a C++ program that uses a while loop to find the largest prime number less than a given positive integer N. Your program should take the value of N as input from the user and then find the largest prime number less than or equal to N. You are not allowed to use any library or pre-existing functions to check for prime numbers.

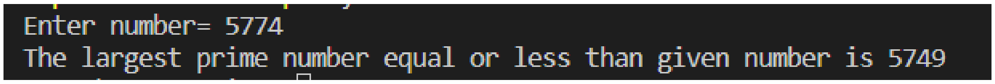
- The program finds the largest prime number that is less than or equal to a given input number. It starts by asking the user to input a number. Then, it uses a nested loop to check each number from 1 up to the given number to see if it is a prime number. If a number is found to be prime, it is stored in the variable k. Finally, the program prints out the largest prime number found.



```cpp
#include<iostream>
using namespace std;
int main()
{
    int n,count=0;
    cout<<"Enter number= ";
    cin>>n;
    int k,i=1;
    while(i<=n)
    {
        count=0;
        int j=1;
        while(j<=i)
        {
            if(i%j==0){count++;}
            j++;
        }
        if(count==2){ k=i;}
        i++;
    }
    cout<<"The largest prime number equal or less than given number is "<<k;
    return 0;
}
```

Program 4 *Program to check nearest prime number.*

Output:



```
Enter number= 5774
The largest prime number equal or less than given number is 5749
```

C++ file

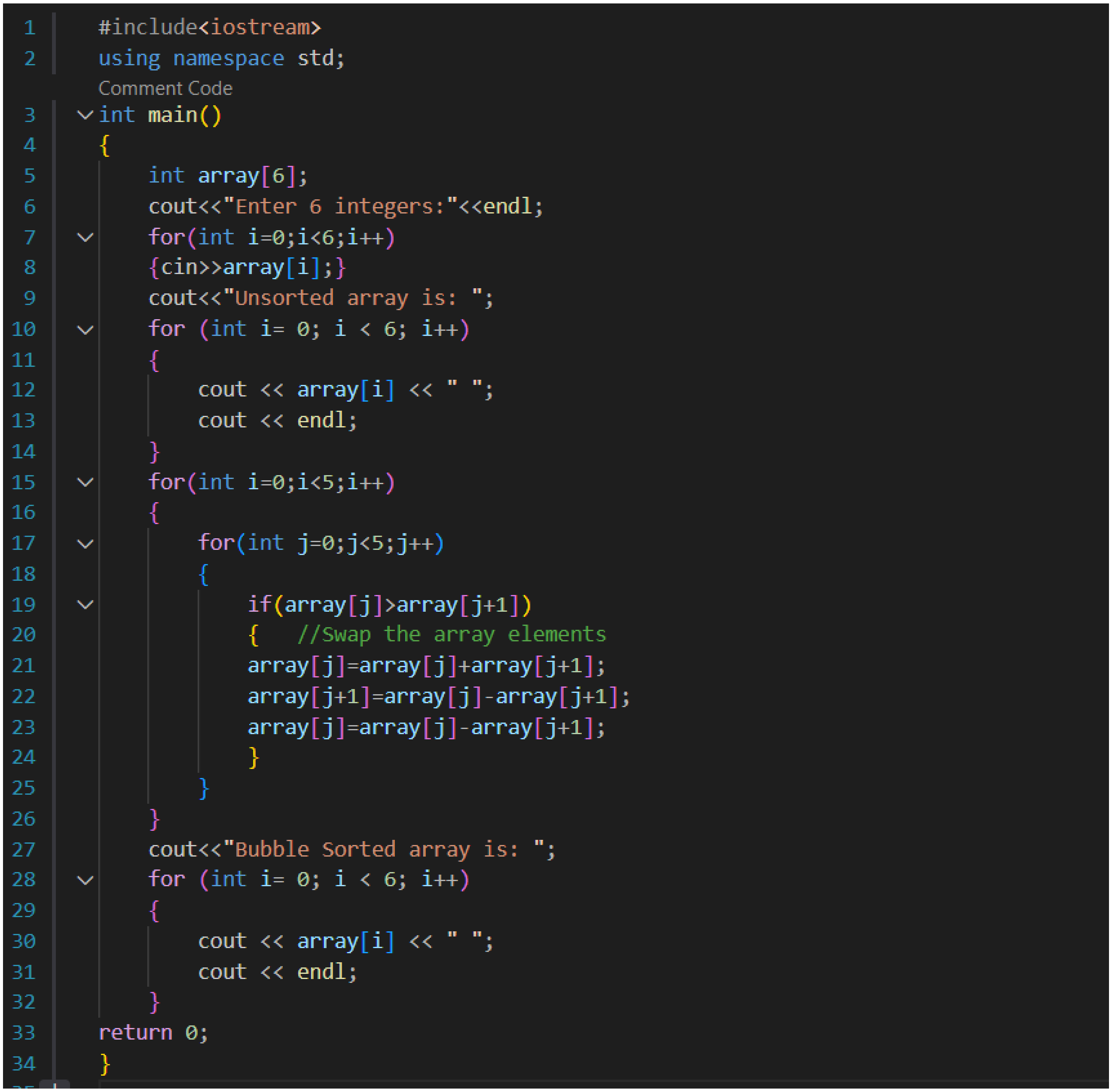
to-find-nearest-prime-no-from-input.cpp

Code

```
#include <iostream>
using namespace std;
int main()
{
    int n,count=0;
    cout<<"Enter number= ";
    cin>>n;
    int k,i=1;
    while(i<=n)
    {
        count=0;
        int j=1;
        while(j<=i)
        {
            if(i%j==0){count++;}
            j++;
        }
        if(count==2){ k=i;}
        i++;
    }
    cout<<"The largest prime number equal or less than given number is "<<k;
    return 0;
}
```

Question NO. 05

Implement Bubble Sort on an array of 6 integers.



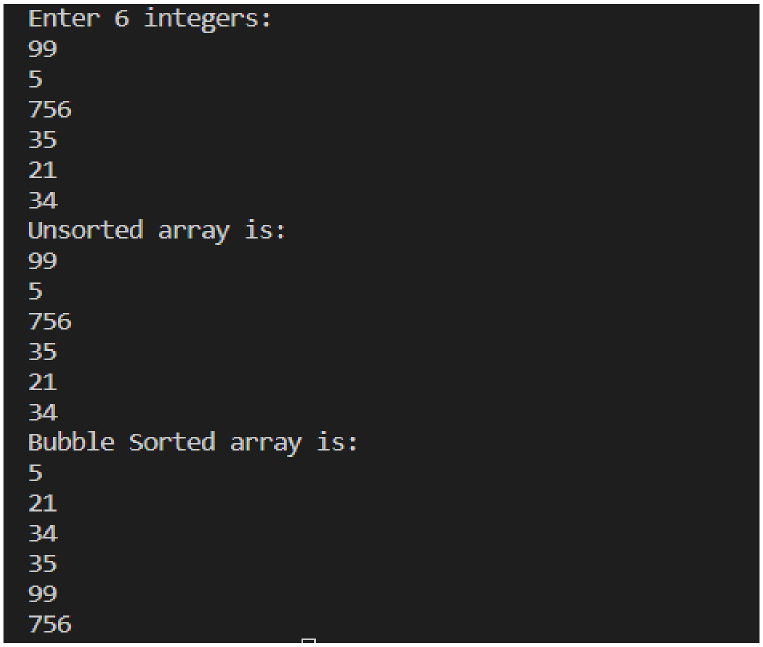
```
#include<iostream>
using namespace std;
int main()
{
    int array[6];
    cout<<"Enter 6 integers:"<<endl;
    for(int i=0;i<6;i++)
    {cin>>array[i];}
    cout<<"Unsorted array is: ";
    for (int i= 0; i < 6; i++)
    {
        cout << array[i] << " ";
        cout << endl;
    }
    for(int i=0;i<5;i++)
    {
        for(int j=0;j<5;j++)
        {
            if(array[j]>array[j+1])
            {   //Swap the array elements
            array[j]=array[j]+array[j+1];
            array[j+1]=array[j]-array[j+1];
            array[j]=array[j]-array[j+1];
            }
        }
    }
    cout<<"Bubble Sorted array is: ";
    for (int i= 0; i < 6; i++)
    {
        cout << array[i] << " ";
        cout << endl;
    }
return 0;
}
```

__Program 5__ Program to implement bubble sort an integer array.

- This code sorts an array of 6 integers using the bubble sort algorithm. First, it asks the user to input 6 integers and stores them in an array. Then it displays the unsorted array. After that, it sorts the array using the bubble sort algorithm and displays the sorted array. The bubble sort algorithm compares adjacent elements and swaps them if they are in the wrong order, repeating this process until the entire array is sorted. This code demonstrates the use of arrays, loops, and conditional statements in C++.

Output:

The output of above program is as follows:



```
Enter 6 integers:
99
5
756
35
21
34
Unsorted array is:
99
5
756
35
21
34
Bubble Sorted array is:
5
21
34
35
99
756
```

C++ file

bubble-sorted-in-integer-array.cpp

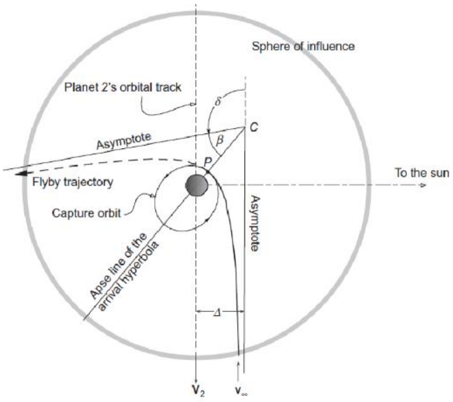
Code

```
#include <iostream>
using namespace std;
int main()
{
    int array[6];
    cout<<"Enter 6 integers:"<<endl;
    for(int i=0;i<6;i++)
    {cin>>array[i];}
    cout<<"Unsorted array is: "<<endl;
    for (int i= 0; i < 6; i++)
    {
        cout << array[i] << " ";
        cout << endl;
    }
    for(int i=0;i<5;i++)
    {
        for(int j=0;j<5;j++)
        {
            if(array[j]>array[j+1])
            {   //Swap the array elements
            array[j]=array[j]+array[j+1];
            array[j+1]=array[j]-array[j+1];
            array[j]=array[j]-array[j+1];
            }
        }
    }
    cout<<"Bubble Sorted array is: "<<endl;
    for (int i= 0; i < 6; i++)
    {
        cout << array[i] << " ";
        cout << endl;
    }
return 0;
}
```

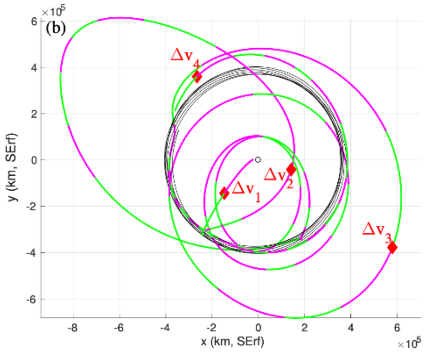
Question NO. 06

Solve any Aerospace/Real Life Problem using C++ Programming.

One real-life problem that could be solved using C++ programming in the aerospace industry is trajectory optimization for space missions. C++ can be used to develop programs that calculate and optimize trajectories for spacecraft to reach their destinations with minimal fuel consumption and time.



By simulating the motion of the spacecraft and using optimization techniques, C++ programs can help mission planners and engineers design optimal trajectories for missions to other planets, satellites, or deep space exploration.



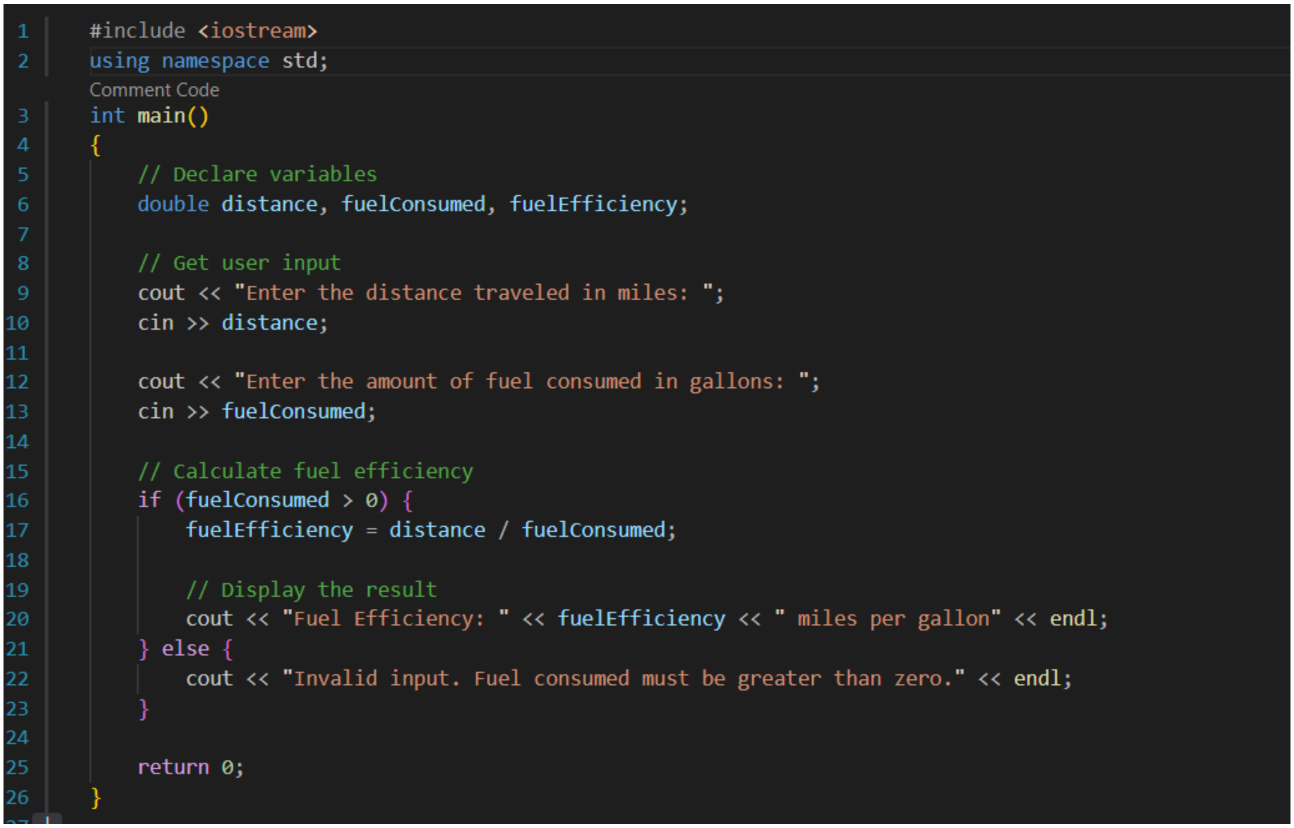
Additionally, C++ can be used to model and simulate the behavior of various systems onboard the spacecraft, such as control systems, propulsion systems, and environmental conditions, enabling engineers to analyze and optimize the overall mission performance.

Thus, we can also find trajectory simulation for a projectile motion with air resistance. In this scenario, we'll model the motion of a projectile (like a rocket or a missile) accounting for air resistance.

We can also calculate the fuel efficiency of a car or an airplane.

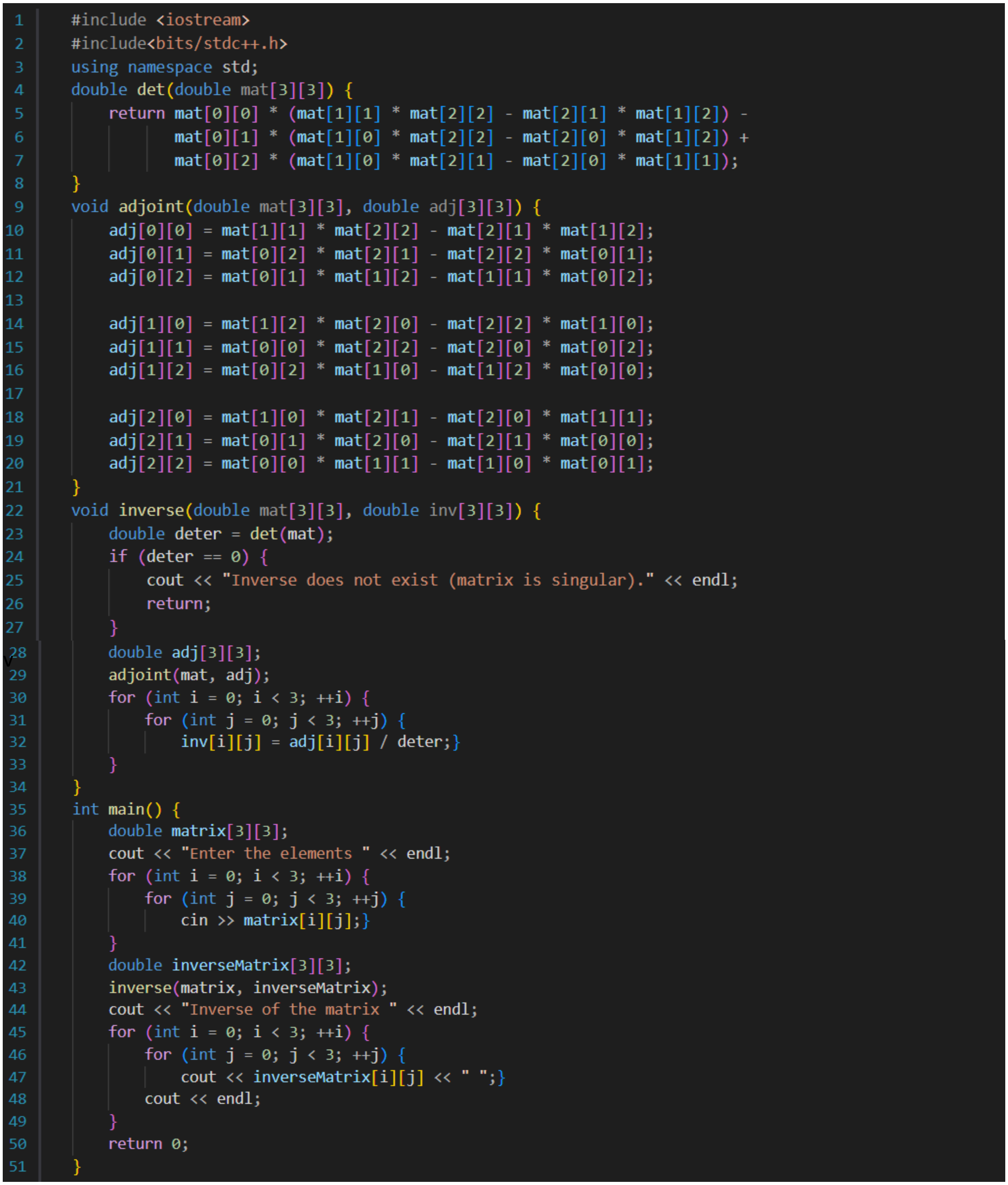
Write a C++ program to calculate the fuel efficiency (miles per gallon) of an airplane. The program takes the distance traveled in miles and the amount of fuel consumed in gallons as input and output the fuel efficiency.

For calculating fuel efficiency



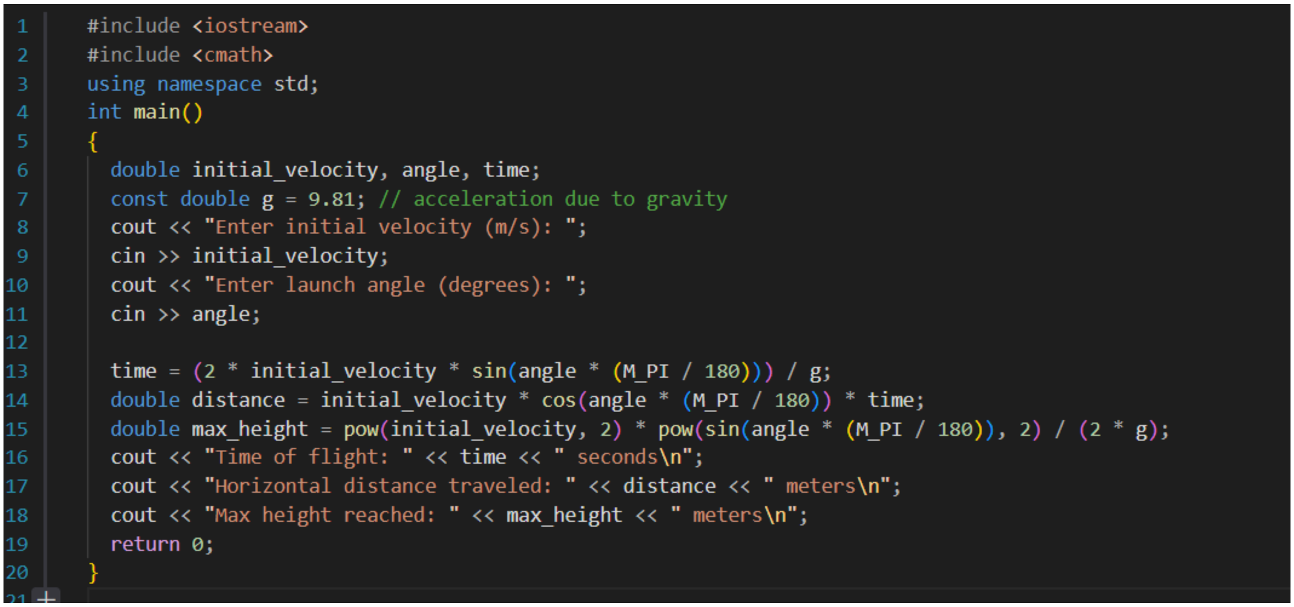
```cpp
#include <iostream>
using namespace std;
int main()
{
    // Declare variables
    double distance, fuelConsumed, fuelEfficiency;

    // Get user input
    cout << "Enter the distance traveled in miles: ";
    cin >> distance;

    cout << "Enter the amount of fuel consumed in gallons: ";
    cin >> fuelConsumed;

    // Calculate fuel efficiency
    if (fuelConsumed > 0) {
        fuelEfficiency = distance / fuelConsumed;

        // Display the result
        cout << "Fuel Efficiency: " << fuelEfficiency << " miles per gallon" << endl;
    } else {
        cout << "Invalid input. Fuel consumed must be greater than zero." << endl;
    }

    return 0;
}
```

For calculating inverse of a matrix



```cpp
#include <iostream>
#include<bits/stdc++.h>
using namespace std;
double det(double mat[3][3]) {
    return mat[0][0] * (mat[1][1] * mat[2][2] - mat[2][1] * mat[1][2]) -
           mat[0][1] * (mat[1][0] * mat[2][2] - mat[2][0] * mat[1][2]) +
           mat[0][2] * (mat[1][0] * mat[2][1] - mat[2][0] * mat[1][1]);
}
void adjoint(double mat[3][3], double adj[3][3]) {
    adj[0][0] = mat[1][1] * mat[2][2] - mat[2][1] * mat[1][2];
    adj[0][1] = mat[0][2] * mat[2][1] - mat[2][2] * mat[0][1];
    adj[0][2] = mat[0][1] * mat[1][2] - mat[1][1] * mat[0][2];

    adj[1][0] = mat[1][2] * mat[2][0] - mat[2][2] * mat[1][0];
    adj[1][1] = mat[0][0] * mat[2][2] - mat[2][0] * mat[0][2];
    adj[1][2] = mat[0][2] * mat[1][0] - mat[1][2] * mat[0][0];

    adj[2][0] = mat[1][0] * mat[2][1] - mat[2][0] * mat[1][1];
    adj[2][1] = mat[0][1] * mat[2][0] - mat[2][1] * mat[0][0];
    adj[2][2] = mat[0][0] * mat[1][1] - mat[1][0] * mat[0][1];
}
void inverse(double mat[3][3], double inv[3][3]) {
    double deter = det(mat);
    if (deter == 0) {
        cout << "Inverse does not exist (matrix is singular)." << endl;
        return;
    }
    double adj[3][3];
    adjoint(mat, adj);
    for (int i = 0; i < 3; ++i) {
        for (int j = 0; j < 3; ++j) {
            inv[i][j] = adj[i][j] / deter;}
    }
}
int main() {
    double matrix[3][3];
    cout << "Enter the elements " << endl;
    for (int i = 0; i < 3; ++i) {
        for (int j = 0; j < 3; ++j) {
            cin >> matrix[i][j];}
    }
    double inverseMatrix[3][3];
    inverse(matrix, inverseMatrix);
    cout << "Inverse of the matrix " << endl;
    for (int i = 0; i < 3; ++i) {
        for (int j = 0; j < 3; ++j) {
            cout << inverseMatrix[i][j] << " ";}
        cout << endl;
    }
    return 0;
}
```

For calculating time of flight, horizontal distance and range of projectile



```cpp
#include <iostream>
#include <cmath>
using namespace std;
int main()
{
  double initial_velocity, angle, time;
  const double g = 9.81; // acceleration due to gravity
  cout << "Enter initial velocity (m/s): ";
  cin >> initial_velocity;
  cout << "Enter launch angle (degrees): ";
  cin >> angle;

  time = (2 * initial_velocity * sin(angle * (M_PI / 180))) / g;
  double distance = initial_velocity * cos(angle * (M_PI / 180)) * time;
  double max_height = pow(initial_velocity, 2) * pow(sin(angle * (M_PI / 180)), 2) / (2 * g);
  cout << "Time of flight: " << time << " seconds\n";
  cout << "Horizontal distance traveled: " << distance << " meters\n";
  cout << "Max height reached: " << max_height << " meters\n";
  return 0;
}
```