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)

"Assignment No.1"

Name: **Muhammad Areeb Arshad**

CMS: **461157**

Question No. 1

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.

Solution

```cpp
#include <iostream>
#include <string>
using namespace std;
int main()
{
    string str1;
    string str2;
    cout<<"Write the 1st string: ";
    cin>>str1;
    cout<<"Write the 2nd string: ";
    cin>>str2;
    int length1 = str1.length();
    int length2 = str2.length();
    bool notequal = false;
    char temp;
    for(int i =0 ; i < length1 ; i++){
        if(length1 != length2 || str1 != str2){
            notequal = true;
            break;
        }
    }
    if (notequal){
        cout<<"String are not equal"<<endl;
    }
    if(notequal != true){
        for (int i =0 ,j =length1 -1 ; i < j ; i++,j--){
            temp = str1[i];
            str1[i]=str1[j];
            str1[j]=temp;
        }
        cout<<"The rotated string is : "<<str1;
    }
    return 0;
}
```

Explanation

This revised C++ code prompts users for two strings, checks if they're equal in length and content. If unequal, it outputs a message stating so. Otherwise, if the strings are equal, it rotates the first string by reversing its characters and displays the rotated string. The code addresses issues by simplifying the equality check and separating it from the string rotation logic, improving efficiency and readability.

Solution Code

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

int main() {
    // Declaration of string variables to store user input
    string str1;
    string str2;

    // Prompting the user to input strings
    cout << "Write the 1st string: ";
    cin >> str1;
    cout << "Write the 2nd string: ";
    cin >> str2;

    // Getting the lengths of both strings
    int length1 = str1.length();
    int length2 = str2.length();

    // Variable to indicate inequality between strings
    bool notequal = false;
    char temp;
```

```
    // Loop to check for inequality in length or content of the strings
    for (int i = 0; i < length1; i++) {
        // If lengths are unequal or contents are different, set notequal flag to true and break
        if (length1 != length2 && str1 != str2) {
            notequal = true;
            break;
        }
    }

    // Display message if strings are not equal
    if (notequal) {
        cout << "String are not equal" << endl;
    }

    // If strings are equal, perform string rotation
    if (notequal != true) {
        // Loop to rotate the string
        for (int i = 0, j = length1 - 1; i < j; i++, j--) {
            // Swap characters to rotate the string
            temp = str1[i];
            str1[i] = str1[j];
            str1[j] = temp;
        }
        // Display the rotated string
        cout << "The rotated string is : " << str1;
    }

    return 0;
}
```

CPP File

Rotate String.cpp

Output

```
Write the 1st string: AreebArshad
Write the 2nd string: AreebArshad
The rotated string is : dahsrAbeerA
```

```
Write the 1st string: sara
Write the 2nd string: usef
String are not equal
```

Question No. 2

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.

Solution

```cpp
#include <iostream>
#include <string>
using namespace std;
int main()
{
    string str;
    cout << "Enter any string: ";
    getline (cin,str);
    cout << endl;
    int length = str.length();
    for (int i = 0; i < length; i++){
        bool diff = true;
        for (int j = 0; j < length; j++){
            if (i != j && str[i] == str[j]){
                diff = false;
                break;
            }
        }
        if (diff){
            cout << str[i];
        }
    }
}
```

Explanation

This code takes a string input from the user and outputs the unique characters present in the entered string, preserving their original order. It achieves this by iterating through each character and checking if it appears elsewhere in the string, printing only the characters that occur just once. The code aims to find and display unique characters from the provided string.

Solution Code

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

int main() {
    // Declaring a string variable to store user input
    string str;

    // Asking the user to input a string
    cout << "Enter any string: ";

    // Using getline to accept a string with spaces and store it in 'str'
    getline(cin, str);

    cout << endl;

    // Getting the length of the input string
    int length = str.length();

    // Loop to iterate through each character in the string
    for (int i = 0; i < length; i++) {
        // Initializing a boolean variable to check if the character is unique
```

```
        bool diff = true;

        // Nested loop to compare the current character with others in the string
        for (int j = 0; j < length; j++) {
            // Checking if the current character matches any other character in the string
            if (i != j && str[i] == str[j]) {
                // If it matches and it's not the same character index, set 'diff' to false
                diff = false;
                break; // Break the loop as the character is not unique
            }
        }

        // If 'diff' is still true, it means the character is unique in the string
        if (diff) {
            // Display the unique character
            cout << str[i];
        }
    }
    return 0;
}
```

CPP File

Q No 2.cpp

Output

```
Enter any string: TheseAssignmentsAreSoTimeConsuming

htrSCu
```

Question No. 3

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

Solution

```cpp
#include <iostream>
using namespace std;
int main()
{
    int arrA[5] = {1,2,3,4,5};
    int arrB[5];
    cout<<"Type in the elements you would like to add: "<<endl<<endl;
    for (int i =0 ; i <5 ; i++)
    {
        cin>>arrB[i];
    }
    cout<<"The array with the elements added: ";
    cout<<"{";
    for (int i = 0 ; i<5 ; i++)
    {
        cout<<arrA[i]<<",";
    }
    for ( int i = 0; i <4; i++)
    {
        cout<<arrB[i]<<",";
    }
    cout<<arrB[4];
    cout<<"}";
}
```

Explanation

This C++ code initializes an array ‘arrA’ with fixed values and prompts the user to input five elements to populate another array ‘arrB’. After the user inputs values, it displays a combined array, presenting ‘arrA’ followed by ‘arrB’, effectively merging both arrays into a single output. The final output showcases the elements of ‘arrA’ first, followed by the elements of ‘arrB’, maintaining their input order.

Solution Code

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

int main() {
   // Initializing an array 'arrA' with fixed values
   int arrA[5] = {1, 2, 3, 4, 5};

   // Initializing an empty array 'arrB'
   int arrB[5];

   // Prompting the user to input elements for 'arrB'
   cout << "Type in the elements you would like to add: " << endl << endl;

   // Loop to accept user input for 'arrB'
   for (int i = 0; i < 5; i++) {
      cin >> arrB[i];
   }

   // Displaying the merged array
   cout << "The array with the elements added: ";
   cout << "{";  // Output opening curly brace for the combined array
```

```
    // Displaying elements of 'arrA'
    for (int i = 0; i < 5; i++) {
        cout << arrA[i] << ",";  // Output each element of 'arrA' followed by a comma
    }

    // Displaying elements of 'arrB', except the last element
    for (int i = 0; i < 4; i++) {
        cout << arrB[i] << ","; // Output each element of 'arrB' followed by a comma
    }

    cout << arrB[4];  // Output the last element of 'arrB'

    cout << "}";  // Output closing curly brace for the combined array

    return 0;
}
```

CPP File

Add elemenets.cpp

Output

```
Type in the elements you would like to add:

6
7
8
9
10
The array with the elements added: {1,2,3,4,5,6,7,8,9,10}
--------------------------------
```

Question No. 4

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.

Solution

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

int main() {
    int N;
    cout << "Type any number 'N': ";
    cin >> N;

    if (N <=1) {
        cout << "There is no Prime Number less than 1!";
    } else if (N ==2) {
        cout << "The largest prime number less than or equal to 'N' is: 2";
    } else {
        for (int num = N-1; num >=2; num--) {
            bool Prime = true;
            for (int i= 2; i*i <= num; i++) {
                if (num % i ==0) {
                    isPrime = false;
                    break;
                }
            }
            if (Prime) {
                cout << "The largest prime number less than or equal to 'N' is: " << num;
                break;
            }
        }
    }
    return 0;
}

```

Explanation

This C++ code prompts the user to input a number 'N'. It then identifies and prints the largest prime number that is less than or equal to 'N'. The code utilizes loops to check each number from 'N - 1' downwards until it finds the largest prime number, considering edge cases where 'N' is less than or equal to 1 or 'N' is equal to 2.

This loop is a method to determine whether a number 'num' is prime or not. It utilizes the fact that factors of a number always appear in pairs: if 'num' is divisible by 'i', then it's also divisible by 'num / i'.

Solution Code

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

int main() {
   int N;
   cout << "Type any number 'N': ";
   cin >> N;

   // Check if 'N' is less than or equal to 1
   if (N <= 1) {
      cout << "There is no Prime Number less than 1!";
   }
   // Check if 'N' is equal to 2
   else if (N == 2) {
      cout << "The largest prime number less than or equal to 'N' is: 2";
   }
   else {
      // Loop to find the largest prime number less than or equal to 'N'
      for (int num = N - 1; num >= 2; num--) {
         bool Prime = true;
```

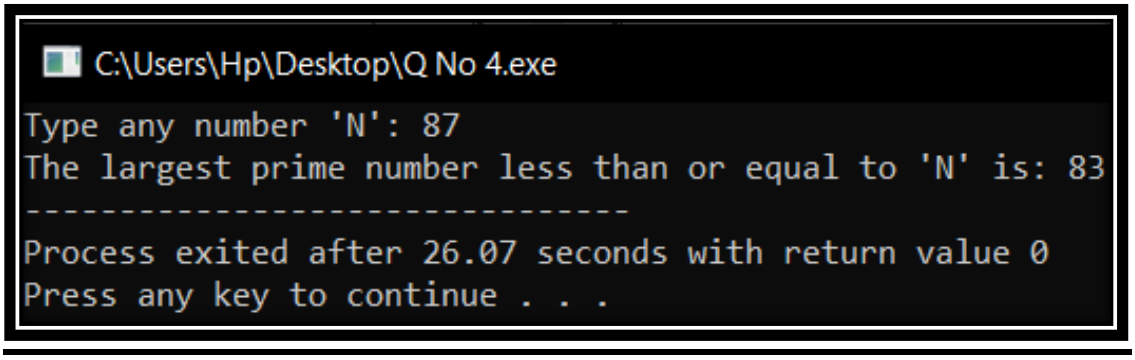
```
        // Check for prime by checking divisibility from 2 to sqrt(num)
        for (int i = 2; i * i <= num; i++) {
            if (num % i == 0) {
                Prime = false;
                break;
            }
        }

        // If 'num' is prime, print it and break the loop
        if (Prime) {
            cout << "The largest prime number less than or equal to 'N' is: " << num;
            break;
        }
    }
  }
  return 0;
}
```

CPP File

Q No 4.cpp

Output



Question No. 5

Implement Bubble Sort on an array of 6 integers.

Solution

```cpp
#include <iostream>
using namespace std;
int main()

{
    int arr[6] = {83,219,30,43,32,431};
    int i;
    int j;
    int temp;
    for (i = 0; i<6; i++){
        for (j = i; j<6; j++ ){
            if (arr[i]<arr[j]){
                temp = arr[i];
                arr[i] = arr[j];
                arr[j] = temp;
            }
        }
    }
    for (int k = 0; k<6; k++){
        cout<<arr[k]<<" ";
    }
    return 0;
}
```

Explanation

Certainly! This C++ code implements the Bubble Sort algorithm to sort an array 'arr' containing six integer elements in descending order. It compares adjacent elements and swaps them if they are in the wrong order, thereby arranging the elements from highest to lowest. Finally, it displays the sorted array.

Solution Code

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

int main() {
   // Initialize an array 'arr' with integer values
   int arr[6] = {83, 219, 30, 43, 32, 431};
   int i;
   int j;
   int temp;

   // Loop to perform sorting in descending order using the Bubble Sort algorithm
   for (i = 0; i < 6; i++) {
      for (j = i; j < 6; j++) {
         // Compare adjacent elements and swap them if they are in the wrong order
         if (arr[i] < arr[j]) {
            temp = arr[i];
            arr[i] = arr[j];
            arr[j] = temp;
         }
      }
   }

   // Display the sorted array in descending order
```

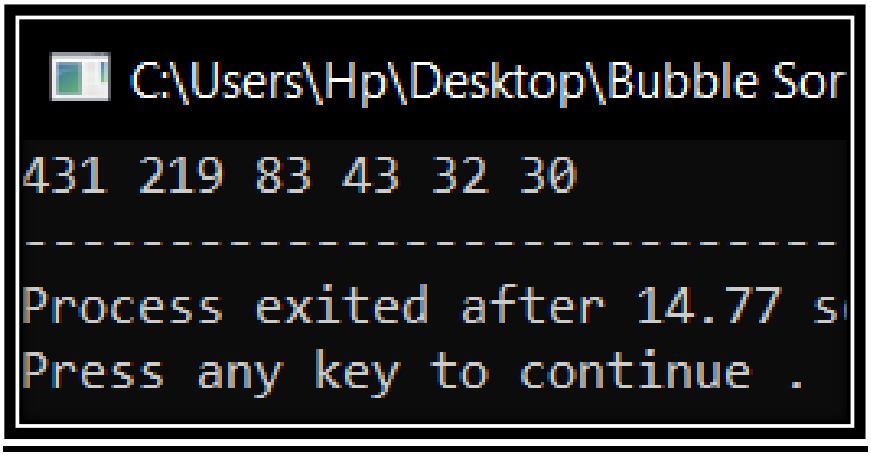
```
    for (int k = 0; k < 6; k++) {
        cout << arr[k] << " ";
    }

    return 0;
}
```

CPP File

Bubble Sort
Assignment.cpp

Output



Bubble Sorted Output of an array in descending order.

Question No. 6

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

Astronautical Problem

This Aerospace Problem involves the user inputting three value of the rocket at a specific point above the Earth along with the angle of its burnout engines to calculate several factors like Orbit type, Whether the Spacecraft would be in orbit or not? And to find out its Time Period, Radius of the orbit, Area of the Orbit & The Orbital Length as well.

Explanation

These are the possible paths that the Spacecraft can take, depending on the three initial conditions which are:

- The Burnout Velocity? (In Km/s)
- The Height of the rocket? (in Km)
- The Angle of Burnout? (in Degrees)

The program will calculate all the conditions mentioned above according to the data given by the user. The formulas used for the program are explained ahead.

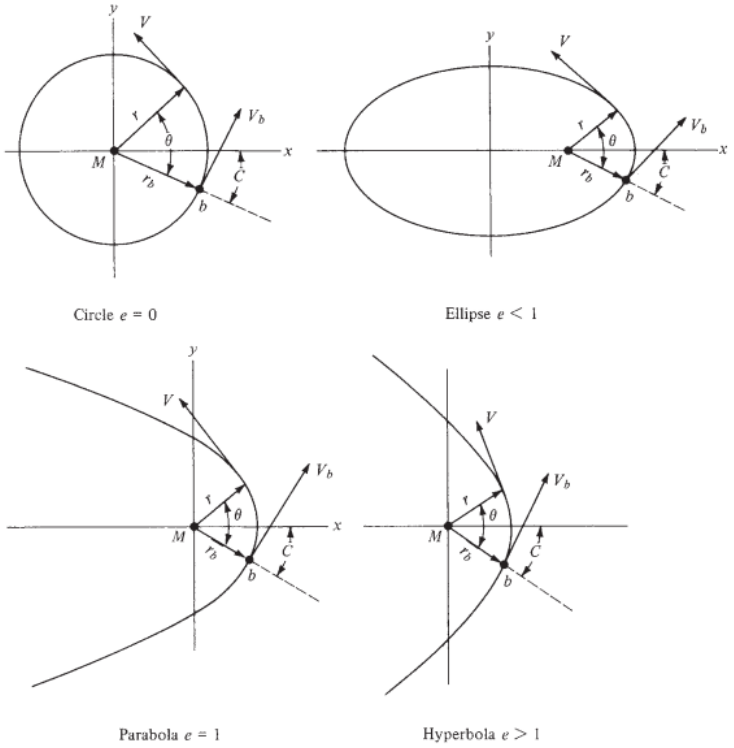
Orbit Conditions:

If $e = 0$, the path is a circle.

If $e < 1$, the path is an ellipse.

If $e = 1$, the path is a parabola.

If $e > 1$, the path is a hyperbola.



The four types of orbits and trajectories, illustrating the relation of the burnout point and phase angle with the axes of symmetry.

Formulae:

$$e = \sqrt{1 + \frac{2h^2 H}{mk^4}}$$

Description:

Here, ‘e’ is the eccentricity of the path that the spacecraft will make, which contains the following parameters:

1. ‘h’ is the Angular Momentum per unit mass of the spacecraft.
2. ‘H’ is the difference between Kinetic & Potential Energy.
3. ‘m’ is the mass constant of spacecraft.
4. K is a constant.

Derivation:

- **‘h’** is the Angular Momentum per unit mass of the spacecraft. It is measured by the following:

$$h = rV_\theta = r_b V \cos\beta_b$$

- Now, V is the given velocity & Beta is the Angle of Burnout.
- $\mathbf{R_b}$ is the burnout radius, calculated from the center of the Earth, which is the sum of the radius of the Earth $\mathbf{R_e}$ & the given Height $\mathbf{h_G}$:

$$r_b = r_e + h_G$$

- 'H' is the difference of the Kinetic and Potential Energy of the Aircraft from the point of burnout given as:

where $H/m = (T - |\Phi|)/m$:

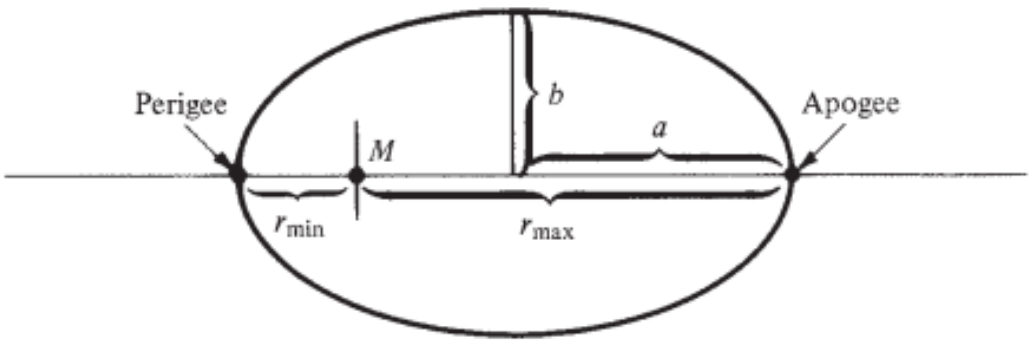
$$\frac{T}{m} = \frac{V^2}{2}$$

$$\left|\frac{\Phi}{m}\right| = \frac{GM}{r_b} = \frac{k^2}{r_b}$$

- Now, with specification, 'm' is considered 1 for a point spacecraft, 'T' is the Kinetic Energy & 'φ' is the Potential Energy from the point to of Burnout.
- 'k' is the constant equal to the product of 'G' called the gravitational constant & 'M' is the mass of the Earth.

The general measurement are all considered with these formulae and parameters. Now moving on to the case of ellipse which are the most basic orbits in planetary system of space.

Apogee & Perigee:



'Perigee' is the closest point & 'Apogee' is the furthest point of a body in an elliptical orbit. These are important for determining the variable velocities in the orbit and for 'Planetary Acceleration' used for unpowered guidance of a Spacecraft in Space, like the one's used for Voyager missions.

Parameters:

- **'r_{min}' is** the Perigee distance. It's formula is:

$$r_{\min} = \frac{h^2/k^2}{1+e}$$

- **'r_{max}' is** the Apogee distance. It's formula is:

$$r_{\max} = \frac{h^2/k^2}{1-e}$$

- **'a'** is the major axis. It's formula is:

$$a = \frac{1}{2}(r_{\max} + r_{\min}) = \frac{1}{2}\frac{h^2}{k^2}\left(\frac{1}{1-e} + \frac{1}{1+e}\right) = \frac{h^2/k^2}{1-e^2}$$

- **'b'** is the minor axis. It's formula is:

$$b = a(1-e^2)^{1/2}$$

- The area of the elliptical orbit is :

$$A = \pi a b$$

- The Distance of Orbit, or the Circumference 'C' of orbit is:

For Elliptical Orbit:

$$C = 2\pi a\sqrt{1-e^2}$$

For Elliptical Orbit:

$$\mathbf{C = 2\pi a^1}$$

Here, either 'a' or 'b' could be used as they are equal in circular orbit.

Time Period for Circular Orbit:

$$\mathbf{T = \frac{2\,\pi\,rmax}{h}}$$

Time Period for Elliptical Orbit:

$$T = \frac{2\,\pi\,a\,b}{h}$$

Now, the program will calculate the eccentricity first and then check the condition of the orbit. After that, it will give the measurements of the trajectory of the Spacecraft according to the eccentricity.

Solution

```
#include <iostream>
#include <cmath>
using namespace std;
int main()
{
    cout<<"What's the Burnout Velocity? (In Km/s): ";
    double V;
    cin>>V;
    cout<<"What's the Height of the rocket? (in Km)?: ";
    double Height;
    cin>>Height;
    cout<<"What's the angle of burnout? (in Degress): ";
    double Angle;
    cin>>Angle;
    cout<<endl;

    double Re = 6.4 * pow(10,6);
    Height = Height * pow(10,3);
    double Rb = Re + Height;
    double k = sqrt(3.986 * pow(10,14));
    Angle = Angle * 3.14 / 180;
    V = V * pow(10,3);
    double h = Rb * V * cos(Angle);
    double KE = pow(V,2)/2;
    double PE = pow(k,2)/Rb;
    double H = KE - PE;

    double e = sqrt(1 + (2 * pow(h,2) * H)/pow(k,4));

    double Apogee = pow(h,2)/(pow(k,2) * (1 - e));
    double Perigee = pow(h,2)/(pow(k,2) * (1 + e));
    double a = pow(h,2)/(pow(k,2) * (1 - pow(e,2)));
    double b = a * sqrt(1 - pow(e,2));
    double AreaC= 3.14159 * Apogee;
    double AreaE = 3.14159 * a * b;
    double Ccircle = 2 * 3.14159 * a;
    double Cellipse = 2 * 3.14159 * a * sqrt(1 - pow(e,2));
    double TimeC = (2 * AreaC)/h;
    double TimeE = (2 * AreaE)/h;
    cout<<"The eccentricity is: "<<e<<endl<<endl;
```

These are all the formulas used, written in form of a code in C++.

```
    if ((e == 0)
    {
        cout<<"The orbit is Circular."<<endl;
        cout<<"Area of the Corbit is: "<<AreaC<<" m^2"<<endl;
        cout<<"Radius of the orbit is: "<<a<<" m"<<endl;
        cout<<"Orbital Circumference is: "<<Ccircle<<" m"<<endl;
        cout<<"Time Period of orbit: "<<TimeC/3600<<" Hr(s)";
    }
    if (e > 0 && e < 1)
    {
        cout<<"The orbit is eliptical."<<endl;
        cout<<"The area of the eliptical orbit is: "<<AreaE<<" m^2"<<endl;
        cout<<"Radius of the orbit is variable as it's an ellipse."<<endl;
        cout<<"Apogee is: "<<Apogee<<" m"<<endl;
        cout<<"Perigee is: "<<Perigee<<" m"<<endl;
        cout<<"Orbital Circumference is: "<<Cellipse<<" m"<<endl;
        cout<<"Time Period of Orbit: "<<TimeE/3600<<" Hr(s)";
    }
    if (e == 1)
    {
        cout<<"The Spaceship would fy off into space as the orbit is a Parabola."<<endl;
        cout<<"Area, Radius, Time Period & Orbital Circumference can't be found as the orbit won't be completed."<<endl;
    }
    if (e >1)
    {
        cout<<"The Spaceship would fy off into space as the orbit is Hyperbolic"<<endl;
        cout<<"Area, Radius, Time Period and Orbital Circumference can't be found as the orbit won't be completed."<<endl;
    }
```

These are the outputs displayed according to the initial conditions.

Solution Code

```
#include <iostream>
#include <cmath>
using namespace std;
int main()
{
    cout<<"What's the Burnout Velocity? (In Km/s): ";
    double V;
    cin>>V;
    cout<<"What's the Height of the rocket? (in Km)?: ";
    double Height;
    cin>>Height;
    cout<<"What's the angle of burnout? (in Degress): ";
    double Angle;
    cin>>Angle;
```

```
cout<<endl;

double Re = 6.4 * pow(10,6);
Height = Height * pow(10,3);
double Rb = Re + Height;
double k = sqrt(3.986 * pow(10,14));
Angle = Angle * 3.14 / 180;
V = V * pow(10,3);
double h = Rb * V * cos(Angle);
double KE = pow(V,2)/2;
double PE = pow(k,2)/Rb;
double H = KE - PE;

double e = sqrt(1 + (2 * pow(h,2) * H)/pow(k,4));

double Apogee = pow(h,2)/(pow(k,2) * (1 - e));
double Perigee = pow(h,2)/(pow(k,2) * (1 + e));
double a = pow(h,2)/(pow(k,2) * (1 - pow(e,2)));
double b = a * sqrt(1 - pow(e,2));
double AreaC= 3.14159 * Apogee;
double AreaE = 3.14159 * a * b;
double Ccircle = 2 * 3.14159 * a;
double Cellipse = 2 * 3.14159 * a * sqrt(1 - pow(e,2));
double TimeC = (2 * AreaC)/h;
double TimeE = (2 * AreaE)/h;
cout<<"The eccentricity is: "<<e<<endl<<endl;

if ((e == 0)
{
```

```
            cout<<"The orbit is Circular."<<endl;
            cout<<"Area of the Corbit is: "<<AreaC<<" m^2"<<endl;
            cout<<"Radius of the orbit is: "<<a<<" m"<<endl;
            cout<<"Orbital Circumference is: "<<Ccircle<<" m"<<endl;
            cout<<"Time Period of orbit: "<<TimeC/3600<<" Hr(s)";
        }
        if (e > 0 && e < 1)
        {
            cout<<"The orbit is elliptical."<<endl;
            cout<<"The area of the elliptical orbit is: "<<AreaE<<" m^2"<<endl;
            cout<<"Radius of the orbit is variable as it's an ellipse."<<endl;
            cout<<"Apogee is: "<<Apogee<<" m"<<endl;
            cout<<"Perigee is: "<<Perigee<<" m"<<endl;
            cout<<"Orbital Circumference is: "<<Cellipse<<" m"<<endl;
            cout<<"Time Period of Orbit: "<<TimeE/3600<<" Hr(s)";
        }
        if (e == 1)
        {
            cout<<"The Spaceship would fly off into space as the orbit is a Parabola."<<endl;
            cout<<"Area, Radius, Time Period & Orbital Circumference can't be found as the
orbit won't be completed."<<endl;
        }
        if (e >1)
        {
            cout<<"The Spaceship would fy off into space as the orbit is Hyperbolic"<<endl;
            cout<<"Area, Radius, Time Period and Orbital Circumference can't be found as the
orbit won't be completed."<<endl;
        }
        }
```

CPP File

Astronautical
Problem.cpp

Output

```
What's the Burnout Velocity? (In Km/s): 3
What's the Height of the rocket? (in Km)?: 860
What's the angle of burnout? (in Degress): 3

The eccentricity is: 0.836569

The orbit is eliptical.
The area of the eliptical orbit is: 2.69099e+13 m^2
Radius of the orbit is variable as it's an ellipse.
Apogee is: 7.26195e+06 m
Perigee is: 646221 m
Orbital Circumference is: 1.36112e+07 m
Time Period of Orbit: 0.687349 Hr(s)
--------------------------------
```

```
What's the Burnout Velocity? (In Km/s): 75
What's the Height of the rocket? (in Km)?: 8765432
What's the angle of burnout? (in Degress): 95

The eccentricity is: 10685

The Spaceship would fy off into space as the orbit is Hyperbolic
Area, Radius, Time Period and Orbital Circumference can't be found as the orbit won't be completed.
```

```
What's the Burnout Velocity? (In Km/s): 9
What's the Height of the rocket? (in Km)?: 860
What's the angle of burnout? (in Degress): 90

The eccentricity is: 1

The Spaceship would fy off into space as the orbit is a Parabola.
Area, Radius, Time Period & Orbital Circumference can't be found as the orbit won't be completed.
```