******

**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 1**

**Name: Ayesha Ayaz**

**CMS:** **481761**

**Question 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.**

This code takes two strings as input from the user, compares them, and performs string rotation on the first string if they are equal.

#include <iostream>

using namespace std;

int main() {

string string1, string2;

cout<<"enter two strings"<<endl;

cin>>string1>>string2; //successfully took 2 string input from user

if(string1!=string2) { //checking if both strings are equal or not

cout<<"both strings are not equal";

}

else {

int string1length=0;

while (string1[string1length] != '\0') {

string1length++; //successfully counted length of string1

}

for (int i = 0; i < string1length / 2; ++i) {

int temp = string1[i];

string1[i] = string1[string1length - i - 1];

string1[string1length - i - 1] = temp;

}

cout << "Rotated string: " << string1 << endl; //successfully rotates string1

}

return 0;

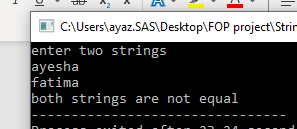
}

In more details this code takes input from user which is declared as *string1* and *string2*

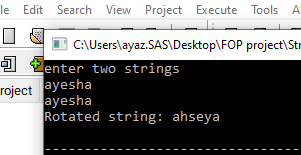
Then it checks if both the strings are equal if they aren’t equal it displayes that both strings aren’t equal if they are equal it finds the length of string1 using a while loop which iterates until the null character after a string is found. It then rotates the characters in string1 by swapping characters from the beginning to the middle with characters from the end to the middle. Finally after rotating string1 it outputs string1.

**Screenshots of output :**

1. If the strings are unequal



1. If both the strings are equal



**Question 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.**

This code removes duplicate characters from the input string, transforms all characters to lowercase, and outputs the resultant string that consists of unique characters

#include <iostream>

using namespace std;

int main() {

string anything,resultant=" ";

cout<<"enter a string"<<endl;

cin>>anything; //successfully took a string input from user

for (char &ch : anything) {

if (ch >= 'A' && ch <= 'Z') {

ch = ch - 'A' + 'a';

} } //making all characters lowercase

int anything\_length=0;

while (anything[anything\_length] != '\0') { //counting length of user inputted string

anything\_length++;

}

for (int i = 0; i < anything\_length ; ++i) {

bool isDuplicate = false;

for (int j = 0; j < anything\_length;++j) {

if (i != j && anything[j] == anything[i]) { // Removing duplicates from string

isDuplicate = true;

break;

}}

if (isDuplicate == false){

resultant += anything[i];

}

}

cout<<resultant;

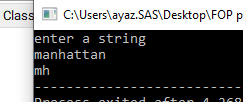
return 0;

}

This code takes input from user, a string that may contain both lowercase and uppercase characters. It processes the input string to ensure all characters are converted to lowercase letters. After that, it counts the length of the string. then utilizes nested loops to iterate through each character of the input string and identify duplicates. It checks for duplicate characters by comparing each character with all other characters in the string. If a character is found to be a duplicate, it is skipped, and the loop moves on to the next character. Throughout this process, a new string called resultant is constructed, storing only the unique characters from the original input string. Once all duplicates have been eliminated, the program prints the resultant string that contains only unique characters from the user-inputted string.

**Screenshots of output :**

User input string being *manhattan*



**Question 3**

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

this code demonstrates how to combine and display elements from an existing array *(a)* and additional elements *(aplus)* by creating a new array of a size large enough to hold all the elements and copying them accordingly into the new array for display.

#include <iostream>

using namespace std;

int main() {

int a[5] = {1, 2, 3, 4, 5};

int aplus[] = {6, 7, 8, 9, 10}; // Elements to be added

int combinedsize = 10;

int aplusa[combinedsize]; //creating new array with combined size

for (int i = 0; i < 5; ++i) {

aplusa[i] = a[i];

}

for (int i = 0; i < 10; ++i) {

aplusa[i + 5] = aplus[i]; //adding both old and new elements to the new array

}

cout << "Updated array elements: ";

for (int i = 0; i < combinedsize; ++i) {

cout << aplusa[i] << " "; //displaying the new array

}

return 0;

}

This code defines an integer array *a* with 5 initial elements It then creates another integer array *aplus* with 5 additional elements that need to be added to the original array.To accommodate the additional elements, a new array *aplusa* of size 10 (combined size of both arrays *a* and *aplus*) is declared. The program then copies the elements from the original array a into the beginning of the new array aplusa using a for loop. Then, it adds the elements from the aplus array to the end of the new array aplusa using another for loop. Finally, the program displays the updated array elements by iterating through the combined array aplusa and printing each element.

**Screenshots of output :**

A black and white screen with white text

Description automatically generated

**Question 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.**

This code makes the user to input a number. It then proceeds to find and display all prime numbers that are less than or equal to the entered number.

#include <iostream>

using namespace std;

int main() {

int number;

cout << "Enter a number: "; //user inputs number

cin >> number;

cout << "Prime numbers below and equal " << number << " are: ";

for (int i = 2; i <= number; ++i) {

bool Prime = true;

for (int j = 2; j \* j <= i; ++j) {

if (i % j == 0) { checking primenumbers below user inputted number

Prime = false;

break;

}}

if (Prime) {

cout << i << " "; //displayying prime number below user entered number

} }

return 0;

}

The program uses nested for loops to iterate through numbers starting from 2 up to the user-entered number (number). For each number within this range, it checks whether it is a prime number or not. For each number i in the range, it performs a primality check using another loop that goes from 2 up to the square root of i. If at any point i is divisible by any number between 2 and the square root of i, it sets the Prime flag to false, indicating that i is not a prime number. If the Prime flag remains true after the inner loop completes, it means that the number i is a prime number, and it gets displayed. Finally, the program displays all the prime numbers that are less than or equal to the user-entered number.

**Screenshots of output :**

A screenshot of a computer

Description automatically generated

**Question 5**

**Implement Bubble Sort on an array of 6 integers**

this code effectively implements the bubble sort algorithm to arrange the elements of the array x in ascending order and displays the sorted array elements using a for loop.

#include <iostream>

using namespace std;

int main() {

int temp,i,j;

int x[6]={14,2,9,1,89,24};

for(int i=0; i<=5; i++)

{

for(int j=i+1; j<=5; j++)

{

if(x[i]>x[j])

{

temp=x[i];

x[i]=x[j];

x[j]=temp;

//successfully sorted it in ascending porder using bubble sort

} }

}

for (int k=0; k<=5;k++)

cout<<x[k]<<endl; //displaying the ascending order array using for loop

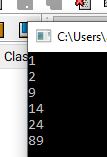
return 0;

}

Firstly this code initializes variables temp, i, and j The code then employs a nested loop structure to perform the bubble sort. The outer loop (for(int i=0; i<=5; i++)) iterates through each element of the array x. Within the outer loop, there's another loop (for(int j=i+1; j<=5; j++)) that compares adjacent elements of the array. If an element at index i is greater than the element at index j, it swaps them using a temporary variable temp. This process continues until the array is sorted in ascending order. After sorting, the program uses a for loop k one to display the elements of the array x in ascending order.finally the sorted elements of array x in ascending order are displayed on the console.

**Screenshots of output :**

Bubble sorted into ascending order



**Question 6**

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

Problem: Rocket Trajectory Calculation

This problem involves a combination of physics, mathematics, and programming. Solving it requires understanding the kinematics of projectile motion and implementing the formulas in a structured C++ program. This problem aims to test skills to translate real-world physics problems into a functioning program while applying proper coding practices in C++.

#include <iostream>

#include <cmath>

using namespace std;

int main() {

double initial\_velocity, launch\_angle, gravitational\_acceleration, air\_resistance\_coefficient, time\_interval;

double max\_height, t\_max, total\_flight\_time, horizontal\_distance, sin\_theta, cos\_theta, v0\_squared;

cout << "Enter initial velocity (m/s): ";

cin >> initial\_velocity;

cout << "Enter launch angle (radians): ";

cin >> launch\_angle;

cout << "Enter gravitational acceleration (m/s^2): ";

cin >> gravitational\_acceleration;

cout << "Enter air resistance coefficient: ";

cin >> air\_resistance\_coefficient;

cout << "Enter time interval for calculations (seconds): ";

cin >> time\_interval;

sin\_theta = sin(launch\_angle); //calculating theta related values

cos\_theta = cos(launch\_angle);

v0\_squared = initial\_velocity \* initial\_velocity; //calculating square value to make calculation easier

max\_height = (v0\_squared \* sin\_theta \* sin\_theta) / (2 \* gravitational\_acceleration); // Calculate maximum height

t\_max = (initial\_velocity \* sin\_theta) / gravitational\_acceleration; // Calculate time to reach maximum height

total\_flight\_time = 2 \* t\_max; // Calculate total time of flight

horizontal\_distance = initial\_velocity \* cos\_theta \* total\_flight\_time; // Calculate horizontal distance traveled

cout << endl;

cout << "Maximum height reached by the rocketis " << max\_height << " meters"<<endl;

cout << "Time taken to reach the maximum height is " << t\_max << " seconds"<<endl;

cout << "Total time of flight before the rocket hits the ground is " << total\_flight\_time << " seconds"<<endl;

cout << "Horizontal distance traveled by the rocket is " << horizontal\_distance << " meters"<<endl;

return 0;

}

This code utilizes user-input values to compute key parameters of a rocket's trajectory. It takes the user input values of the initial velocity, launch angle in radians, gravitational acceleration, air resistance coefficient, and time interval. Using trigonometric functions and standard mathematical operations, such as sin and cos, using the cmath library the program calculates the rocket's maximum height, time taken to reach that height, total flight time before landing, and horizontal distance traveled. Finally, it displays these calculated values using cout statements, providing essential insights into the rocket's flight path.

**Screenshots of output :**

