



***School of Mechanical & Manufacturing Engineering (SMME),
National University of Science and Technology (NUST),***

Program: BE Aerospace Engineering

Section: AE-01

Session: Fall 2023

Semester: 1st

Course: Fundamentals of Programming

Course code: CS-114

Lab Assignment

Name: **Syed Amna Hasnain**

CMS ID: **467234**

Question no 1:

Write a C++ program, take two strings as input from the 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>
#include <string>
using namespace std;
int main()
{
    string str1, str2;

    cout << "Enter the first string: ";
    cin >> str1;

    cout << "Enter the second string: ";
    cin >> str2;

    if (str1 == str2)
    {
        char temp = str1[0];
        str1[0] = str1[str1.length() - 1];
        str1[str1.length() - 1] = temp;

        cout << "After swapping, the strings are now unequal: "<< endl;
        cout << "String 1: " << str1 << endl;
        cout << "String 2: " << str2 << endl;
    } else {
        cout << "The entered strings are already unequal"<<endl;
    }

    return 0;
}
```

Output:

```
Enter the first string: Hello
Enter the second string: Hello
After swapping, the strings are now unequal:
String 1: oellH
String 2: Hello

-----
Process exited after 8.097 seconds with return value 0
Press any key to continue . . .
```

Explanation:

This simple C++ program takes two strings as input from the user, checks if they are equal, and if so, modifies one of the strings by rotating its characters to make them unequal. The rotation is done manually without using any functions. The program then outputs the modified strings or informs the user if the strings were already unequal.

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.

```
#include <iostream>
#include <string>
using namespace std;
int main() {
    string originalString;
    cout << "Enter a string: ";
    cin >> originalString;

    string uniqueString;

    for (char currentChar : originalString) {
        if (uniqueString.find(currentChar) == std::string::npos) {
            uniqueString += currentChar;
        }
    }

    cout << "Resultant string after removing duplicates: " <<
        uniqueString << std::endl;

    return 0;
}
```

Output:

```
Enter a string: aleezay
Resultant string after removing duplicates: alezy
```

Explanation:

This C++ program removes duplicate characters from a user-entered string and outputs the resulting string without duplicates. The program prompts the user to enter a string. The input string is stored in the variable `originalString`.

A new string, `uniqueString`, is declared to store the characters without duplicates. A for loop iterates through each character (`currentChar`) in the `originalString`.

Inside the loop, it checks whether the current character is already present in uniqueString using find.If the character is not found, it is added to uniqueString, effectively removing duplicates.The program prints the resultant string after removing duplicates using cout.The program returns 0, indicating successful execution.

Question no 3:

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

    cout << "Original array elements: ";
    for (int i = 0; i < 5; ++i) {
        cout << originalArray[i] << " ";
    }

    cout << endl;
    int additionalElements[] = {6, 7, 8, 9, 10};

    cout << "Updated array elements: ";


    for (int i = 0; i < 5; ++i) {
        cout << originalArray[i] << " ";
    }

    for (int i = 0; i < sizeof(additionalElements) / sizeof(int); ++i) {
        cout << additionalElements[i] << " ";
    }

    cout << endl;

    return 0;
}
```

Output:

 C:\Users\FineCom\Documents\Untitled1.exe

```
Original array elements: 1 2 3 4 5
Updated array elements: 1 2 3 4 5 6 7 8 9 10
```

```
-----
Process exited after 0.3953 seconds with return value 0
Press any key to continue . . .
```

Explanation:

This simple C++ program initialise an array originalArray with 5 elements, displays its contents, adds more elements from additionalElements to it, and then displays the updated array. The variable names are beginner-friendly, and the program serves as a basic example of array manipulation and display in C++.

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.

```
#include <iostream>
using namespace std;
bool isPrime(int number) {
    if (number <= 1) {
        return false;
    }

    for (int i = 2; i < number; ++i) {
        if (number % i == 0) {
            return false;
        }
    }
    return true;
}
int main() {
    int N;
    cout << "Enter a positive integer N: ";
    cin >> N;

    while (N > 1) {
        if (isPrime(N)) {
            cout << "Largest prime number less than or equal to " <<
```

```

        N << ": " << N << std::endl;
        return 0;
    }
    --N;
}
cout << "No prime number found less than or equal to 1." << endl
;

return 0;
}

```

Output:

Output

```

/tmp/kDwo50xUx2.o
Enter a positive integer N: 2
Largest prime number less than or equal to 2: 2

```

Explanation:


This C++ program finds and prints the largest prime number less than or equal to a positive integer `N` entered by the user. It defines a function `isPrime` to check if a number is prime. It uses a `while` loop to decrement `N` until a prime number is found. The program then outputs the largest prime number less than or equal to the user-input value. If no prime number is found (if the user enters 1), it prints a corresponding message.

Question no 5:

Implement Bubble Sort on an array of 6 integers.

```
#include <iostream>
using namespace std;
int main() {
    int arr[] = {6, 3, 1, 7, 5, 2}; // Example array of 6 elements
    int n = sizeof(arr) / sizeof(arr[0]); // Get the size of the array
    for (int i = 0; i < n-1; i++) {
        bool swapped = false;
        for (int j = 0; j < n-i-1; j++) {
            if (arr[j] > arr[j+1]) {
                // Swap elements if out of order
                int temp = arr[j];
                arr[j] = arr[j+1];
                arr[j+1] = temp;
                swapped = true;
            }
        }
        if (!swapped) {
            break;
        }
    }
    cout << "Sorted array: ";
    for (int i = 0; i < n; i++) {
        cout << arr[i] << " ";
    }
    cout << endl;
    return 0;
}
```

Output:

 C:\Users\FineCom\Documents\Untitled1.exe

```
Sorted array: 1 2 3 5 6 7
```

```
-----
```

```
Process exited after 0.6464 seconds with return value 0
```

```
Press any key to continue . . .
```

Explanation:

- This C++ code implements the Bubble Sort algorithm to sort an array of integers in ascending order.
- An array `arr` is declared and initialised with six integers.
- The variable `n` is calculated to store the size of the array.
- The variable `n` is calculated to store the size of the array.
- The expression `sizeof(arr)` returns the total number of bytes occupied by the array, and `sizeof(arr[0])` returns the size of one element in the array.
- Outer Loop: The outer loop iterates `n-1` times, where `n` is the number of elements in the array. This loop represents each pass through the array.

- Inner loop: The inner loop iterates through the unsorted portion of the array. In each pass, it compares adjacent elements and swaps them if they are in the wrong order.
- Swapping: If the element at index j is greater than the element at index $j+1$, they are swapped. A boolean variable swapped is set to true to indicate that a swap has occurred.
- If no swaps occurred in a pass, the array is already sorted, and the algorithm can exit early.
- Finally, the sorted array is printed to the console.

Question no 6:

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

```
#include <iostream>
using namespace std;
int main()
{
    double distance, fuel, fuelEfficiency;
    cout << "Enter the distance traveled (in kilometers): ";
    cin >> distance;
    cout << "Enter the fuel consumed (in liters): ";
    cin >> fuel;

    fuelEfficiency = distance / fuel;
    cout << "Fuel Efficiency: " << fuelEfficiency << " kilometers per liter" << endl;
    return 0;
}
```

Output:

```
C:\Users\FineCom\Documents\Untitled1.exe
Enter the distance traveled (in kilometers): 50
Enter the fuel consumed (in liters): 2
Fuel Efficiency: 25 kilometers per liter

-----
Process exited after 10.74 seconds with return value 0
Press any key to continue . . .
```


Explanation:

- Declare three double-precision floating-point variables: distance for the distance travelled, fuel for the amount of fuel consumed, and fuelEfficiency to store the calculated fuel efficiency.
- Now take user input for the distance and store it in the distance variable.
- Take user input for the fuel consumed and store it in the fuel variable.
- Calculate fuel efficiency by dividing the distance by the amount of fuel consumed.
- Finally, the calculated fuel efficiency in kilometres per litre is printed to the console.