******

***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***

***Name: Bashayer Amjad***

***CMS: 466710***

**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.

**Code:**

#include <iostream>

using namespace std;

int main() {

string str1, str2;

cout << "Enter string 1: ";

getline(cin, str1);

cout << "Enter string 2: ";

getline(cin, str2);

if (str1==str2) {

int finish, start;

start=0;

finish=str2.length()-1;

while (start<finish) {

char temp;

temp=str2[finish];

str2[finish]=str2[start];

str2[start]=temp;

start++;

finish--;

}

cout << "String 1: " << str1 << endl;

cout << "String 2: " << str2 << endl;

} else {

cout << "Strings are not equal.";

}

return 0;

}

A screenshot of a computer

Description automatically generated

Figure : Output Q1

A screenshot of a computer

Description automatically generated

Figure 2: Output Q1

**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.

**Code:**

#include <iostream>

#include <string>

using namespace std;

int main() {

string inputString;

cout << "Enter a string: ";

getline(cin, inputString);

string resultString = "";

for (auto it = inputString.begin(); it != inputString.end(); ++it) {

char ch = \*it;

char lowercaseCh = tolower(ch);

bool isDuplicate = false;

for (auto resultIt = resultString.begin(); resultIt != resultString.end(); ++resultIt) {

if (tolower(\*resultIt) == lowercaseCh) {

isDuplicate = true;

break;

}

}

if (!isDuplicate) {

resultString += ch;

}

}

cout << "Resultant string after removing duplicates: " << resultString << endl;

return 0;

}

A screenshot of a computer

Description automatically generated

Figure 3: Output Q2

A screenshot of a computer

Description automatically generated

Figure 4: Output Q2

**Question 3**

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

#include <iostream>

using namespace std;

int main() {

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

int size = 5;

int newSize;

cout << "Enter new size of array: ";

cin >> newSize;

int newArray[newSize];

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

newArray[i] = a[i];

}

for (int j = size; j < newSize; j++) {

newArray[j] = a[size - 1] + j - size + 1;

}

cout << "Updated array: ";

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

cout << newArray[i] << " ";

}

cout << endl;

return 0;

}

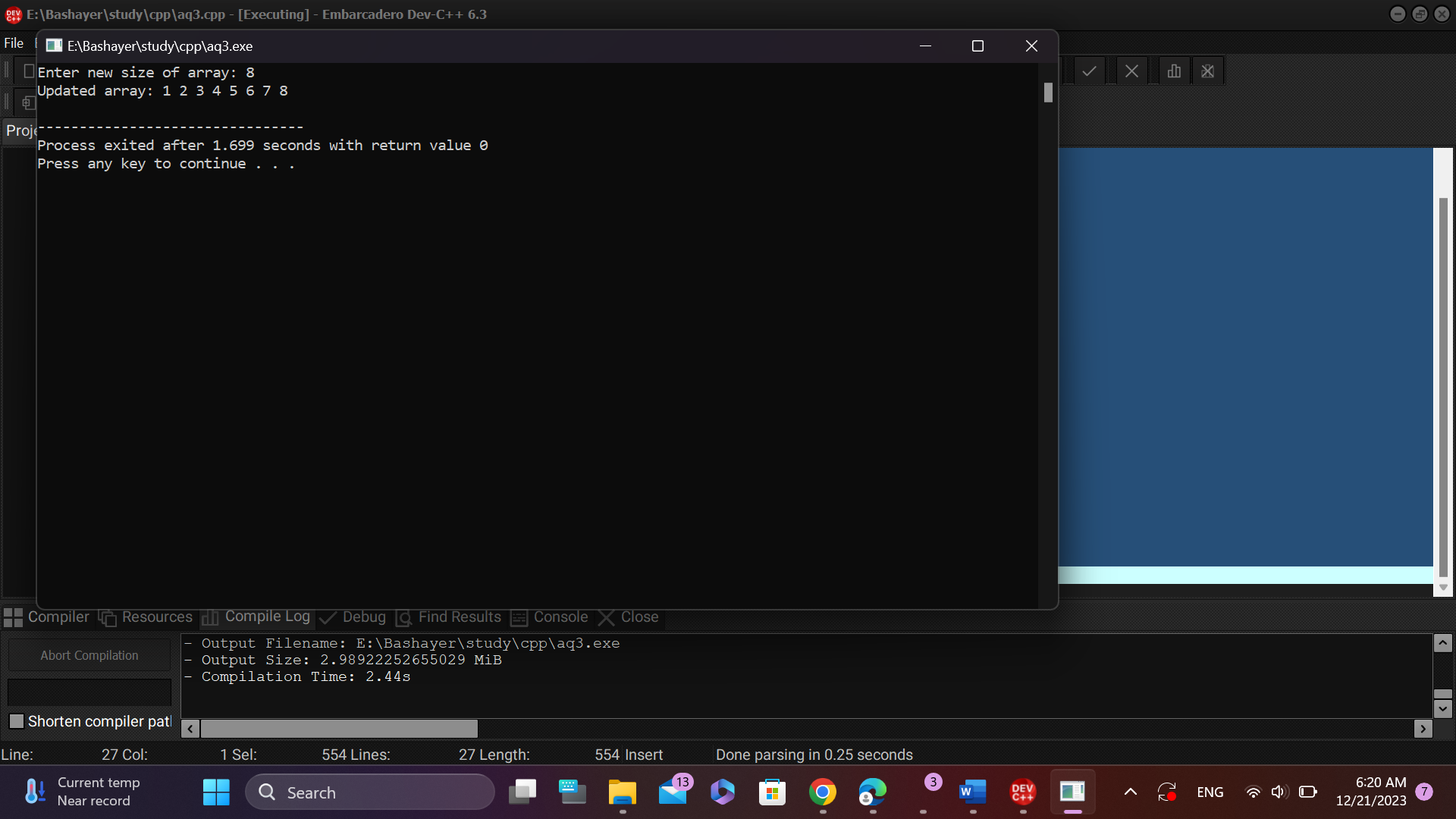


Figure 5: Q3 Output

A screenshot of a computer

Description automatically generated

Figure 6: Q3 Output

**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.

**Code:**

#include <iostream>

using namespace std;

int main() {

int N, i, j, isPrime;

cout << "Enter a positive integer: ";

cin >> N;

if (N <= 1) {

cout << "Invalid input. Please enter a positive integer greater than 1." << endl;

return 1;

}

i = N - 1;

while (i > 1) {

isPrime = true;

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

if (i % j == 0) {

isPrime = false;

break;

}

}

if (isPrime) {

cout << "The largest prime number less than " << N << " is " << i << endl;

break;

}

i--;

}

return 0;

}

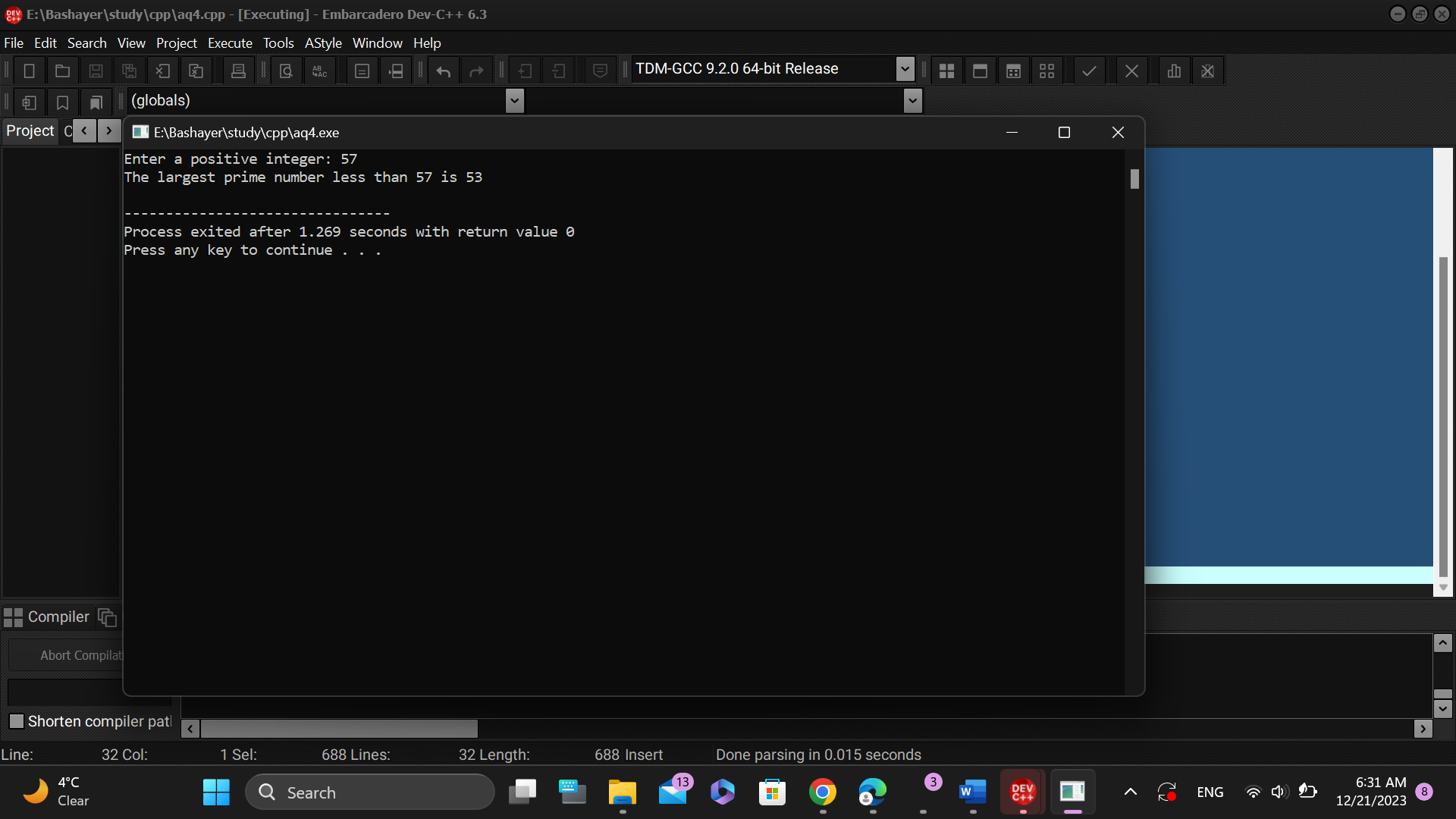


Figure 7: Q4 Output

A screenshot of a computer

Description automatically generated

Figure 8: Q4 Output

**Question 5**

Implement Bubble Sort on an array of 6 integers.

**Code:**

#include <iostream>

using namespace std;

int main() {

const int size = 6;

int arr[size];

cout << "Enter 6 integers for the array:" << endl;

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

cin >> arr[i];

}

for (int i = 0; i < size - 1; ++i) {

for (int j = 0; j < size - i - 1; ++j) {

if (arr[j] > arr[j + 1]) {

int temp = arr[j];

arr[j] = arr[j + 1];

arr[j + 1] = temp;

}

}

}

cout << "Sorted array: ";

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

cout << arr[i] << " ";

}

cout << endl;

return 0;

}

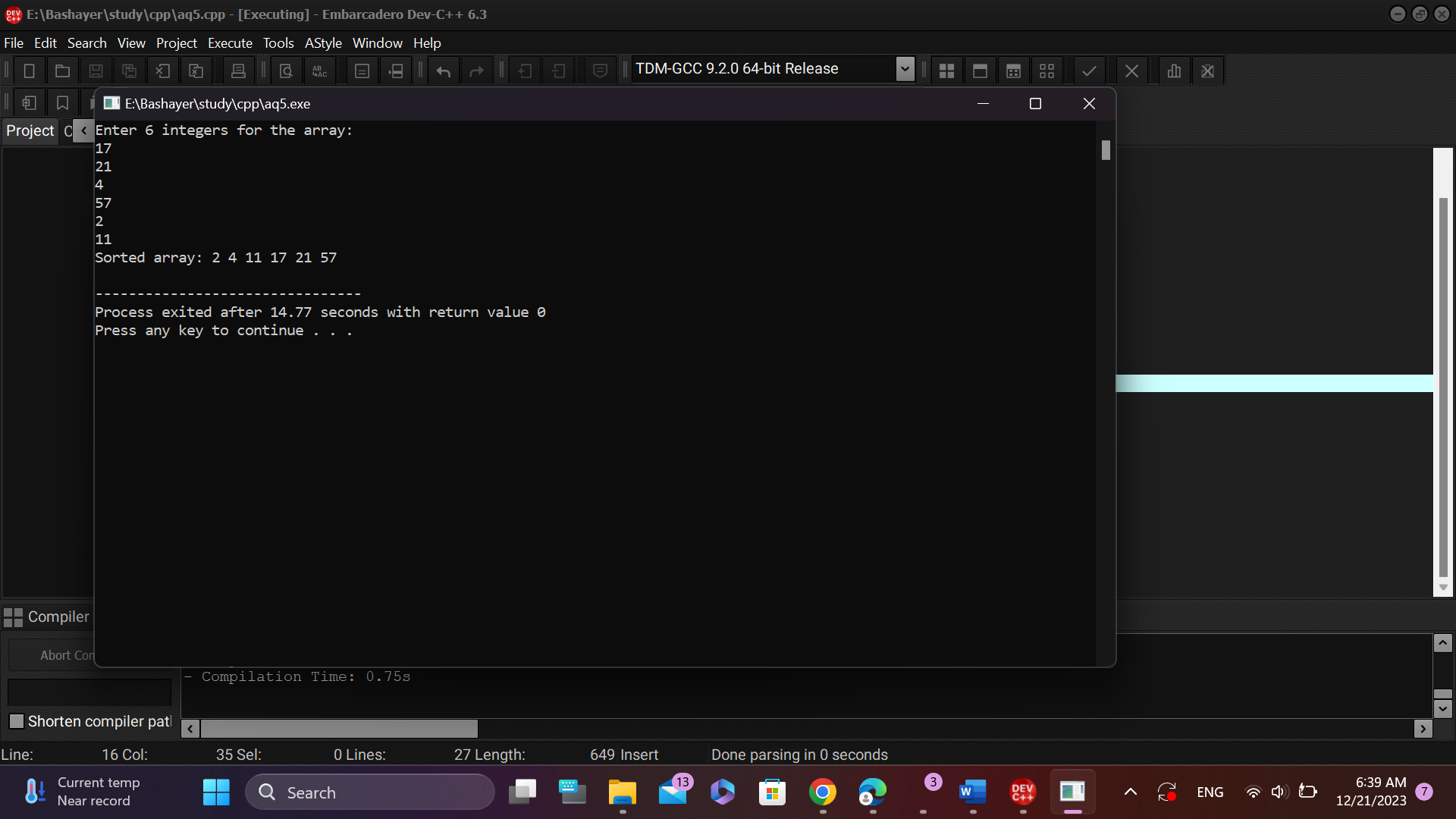


Figure 9: Q5 Output

**Question 6**

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

**Code:**

#include <iostream>

#include <string>

#include <cctype>

using namespace std;

bool isValidPassword(const string& password) {

bool hasLowercase = false, hasUppercase = false, hasDigit = false, hasSpecial = false;

string allowedSpecialChars = "!@\_-;:.?";

for (char c : password) {

if (islower(c)) {

hasLowercase = true;

} else if (isupper(c)) {

hasUppercase = true;

} else if (isdigit(c)) {

hasDigit = true;

} else if (allowedSpecialChars.find(c) != string::npos) {

hasSpecial = true;

}

}

return password.length() >= 8 && hasLowercase && hasUppercase && hasDigit && hasSpecial;

}

int main() {

string password, confirmPassword;

string allowedSpecialChars = "!@\_-;:.?";

// Password setup

cout << "Enter a new password: ";

// Enhanced password strength check

while (getline(cin, password)) {

if (!isValidPassword(password)) {

cout << "Password must meet the following requirements:\n"

<< "- At least 8 characters long\n"

<< "- Contain at least one lowercase letter\n"

<< "- Contain at least one uppercase letter\n"

<< "- Contain at least one digit\n"

<< "- Contain at least one of the following special characters: "

<< allowedSpecialChars << endl

<< "Please try again: ";

} else {

break; // Password is valid

}

}

// Password confirmation

do {

cout << "Confirm password: ";

getline(cin, confirmPassword);

if (confirmPassword != password) {

cout << "Passwords do not match. Please try again." << endl;

}

} while (confirmPassword != password);

cout << "Password set successfully!" << endl;

// Password entry

string enteredPassword;

int attempts = 0;

do {

cout << "Enter password: ";

cin >> enteredPassword;

attempts++;

if (enteredPassword == password) {

cout << "Password correct!" << endl;

break;

} else {

cout << "Incorrect password. " << (3 - attempts) << " attempts remaining." << endl;

// Clear input stream to avoid infinite loop on invalid input

cin.clear();

}

} while (attempts < 3);

if (attempts == 3) {

cout << "Too many incorrect attempts. Access denied." << endl;

}

return 0;

}

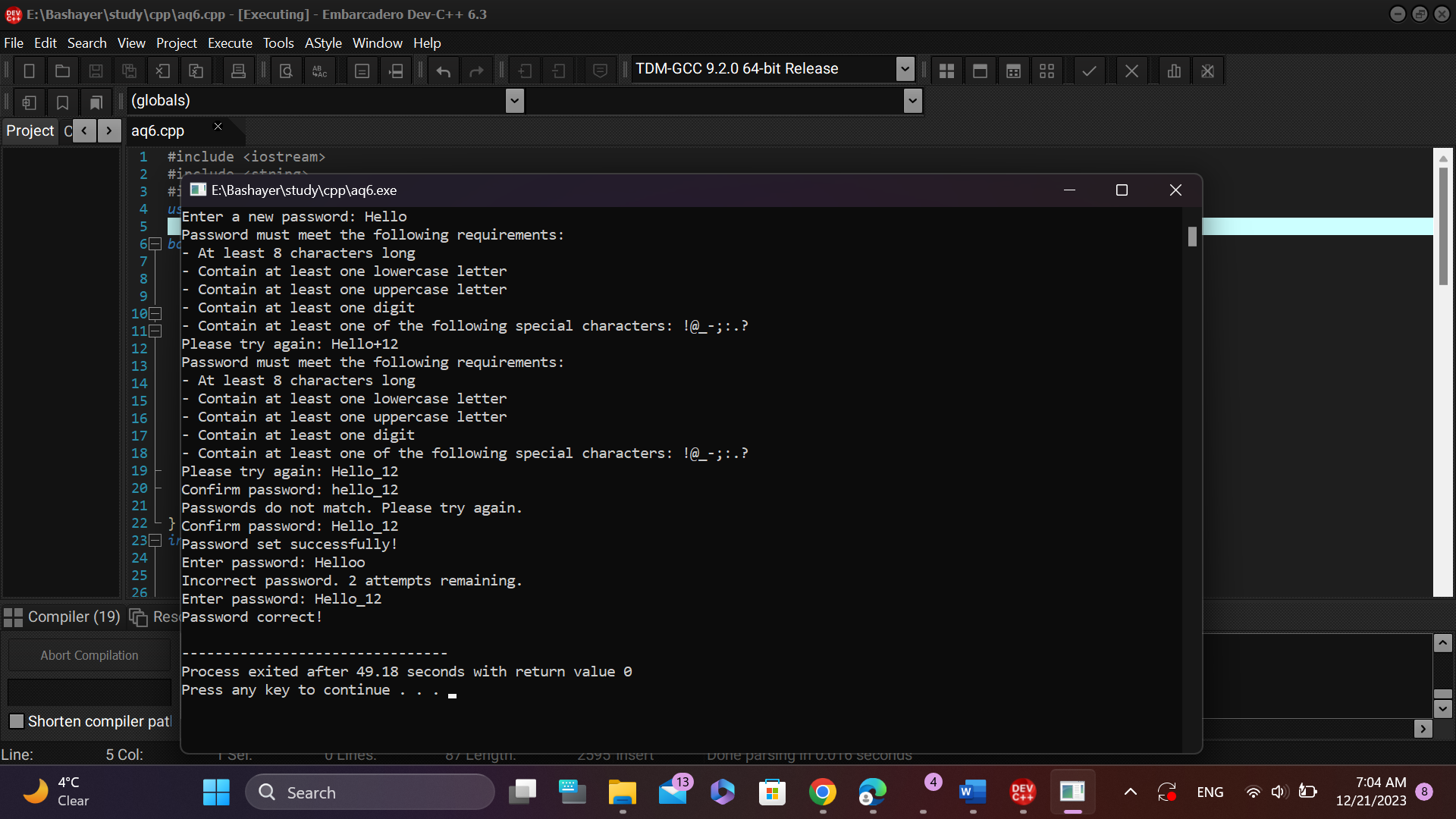


Figure10: Q6 Output