**FACTORIAL USING FUNCTIONS:**

#include <iostream>

using namespace std;

int factorial(int n) {

if (n <= 1)

return 1;

else

return n \* factorial(n - 1);

}

int main() {

int number;

cout << "Enter a number: ";

cin >> number;

if (number < 0)

cout << "Factorial of a negative number doesn't exist." << endl;

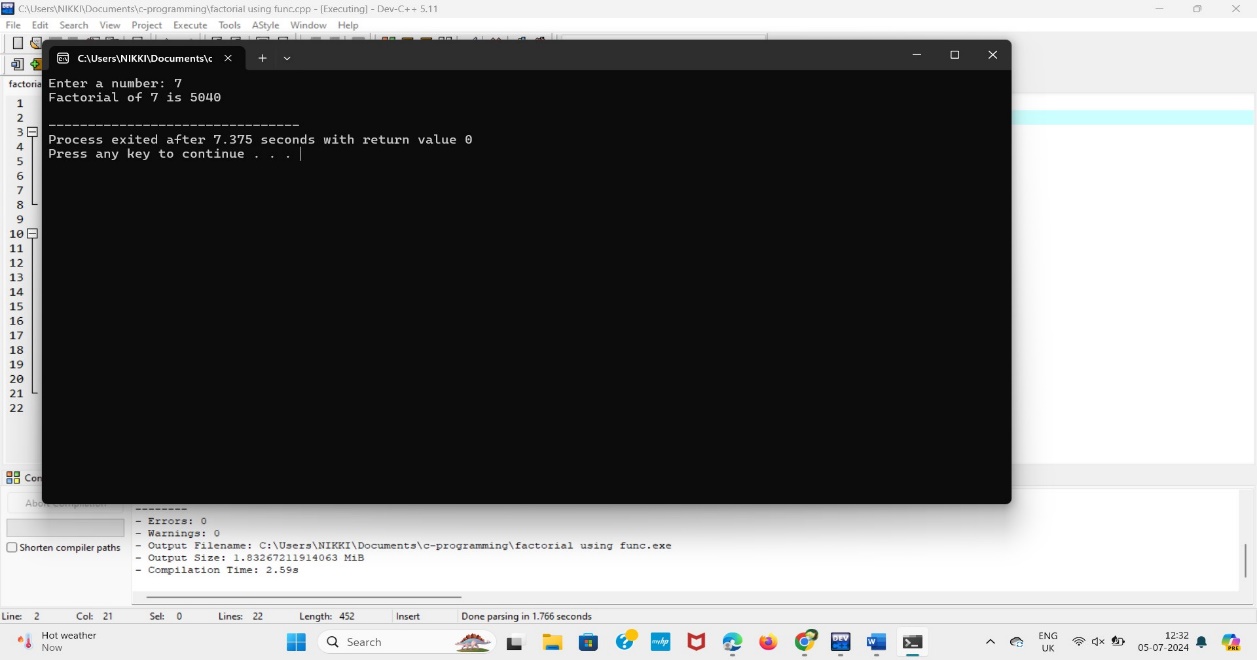
else

cout << "Factorial of " << number << " is " << factorial(number) << endl;

return 0;

}

**OUTPUT:**



**PRIME NUMBER USING FUNCTION:**

#include <iostream>

using namespace std;

bool isPrime(int n) {

if (n <= 1)

return false;

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

if (n % i == 0)

return false;

}

return true;

}

int main() {

int number;

cout << "Enter a number: ";

cin >> number;

if (isPrime(number))

cout << number << " is a prime number." << endl;

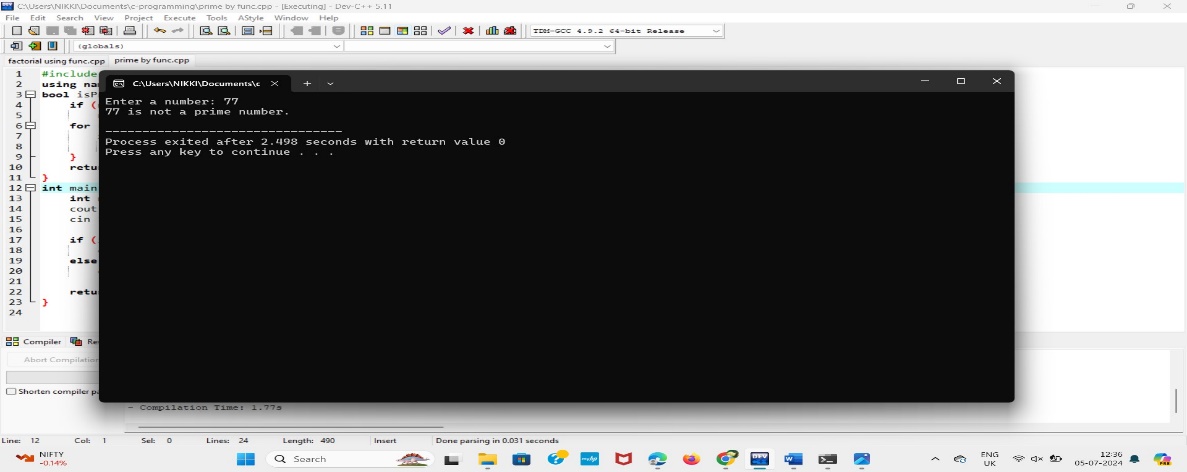
else

cout << number << " is not a prime number." << endl;

return 0;

}

**OUTPUT:**



**REVERSING THE NUMBER USING FUNCTION:**

#include <iostream>

using namespace std;

int reverseNumber(int n) {

int reversedNumber = 0;

while (n != 0) {

int digit = n % 10;

reversedNumber = reversedNumber \* 10 + digit;

n /= 10;

}

return reversedNumber;

}

int main() {

int number;

cout << "Enter a number: ";

cin >> number;

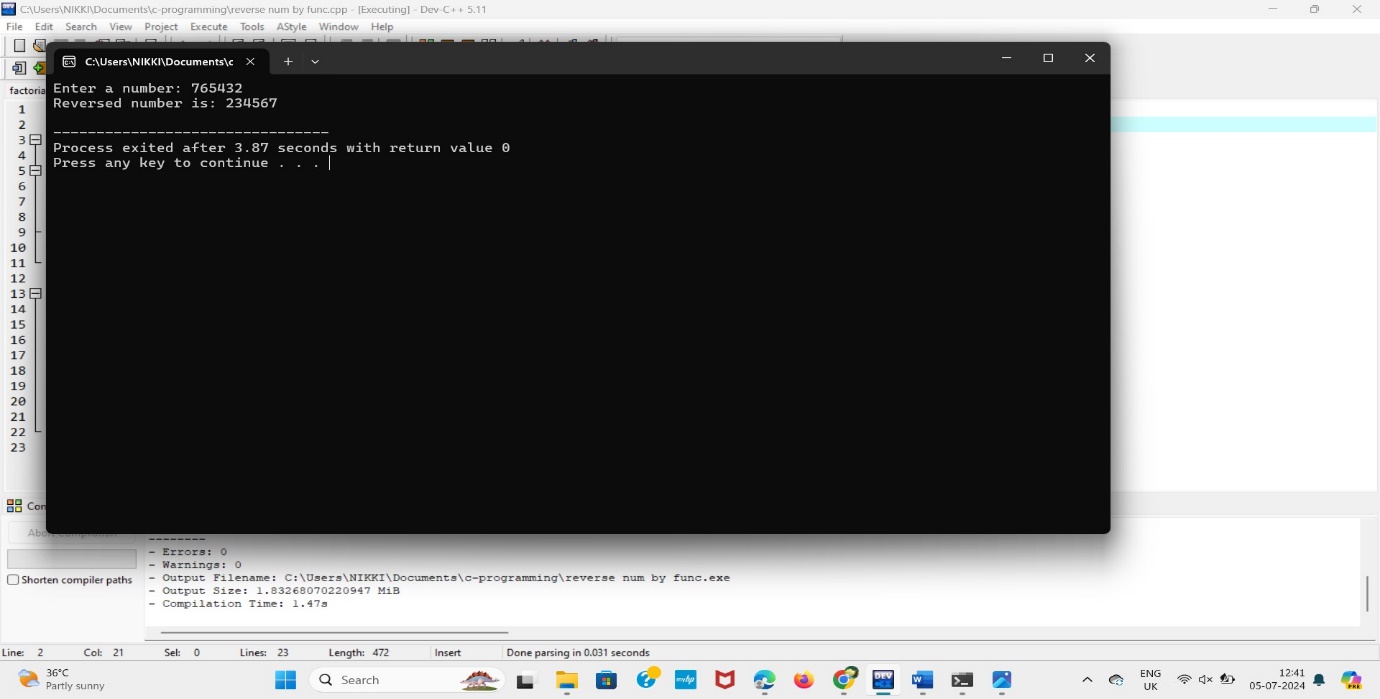
int reversed = reverseNumber(number);

cout << "Reversed number is: " << reversed << endl;

return 0;

}

**OUTPUT:**



**minimum and maximum element in an array using function:**

#include <iostream>

using namespace std;

int findMin(int arr[], int size) {

int min = arr[0];

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

if (arr[i] < min)

min = arr[i];

}

return min;

}

int findMax(int arr[], int size) {

int max = arr[0];

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

if (arr[i] > max)

max = arr[i];

}

return max;

}

int main() {

int size;

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

cin >> size;

int arr[size];

cout << "Enter the elements of the array: ";

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

cin >> arr[i];

}

int min = findMin(arr, size);

int max = findMax(arr, size);

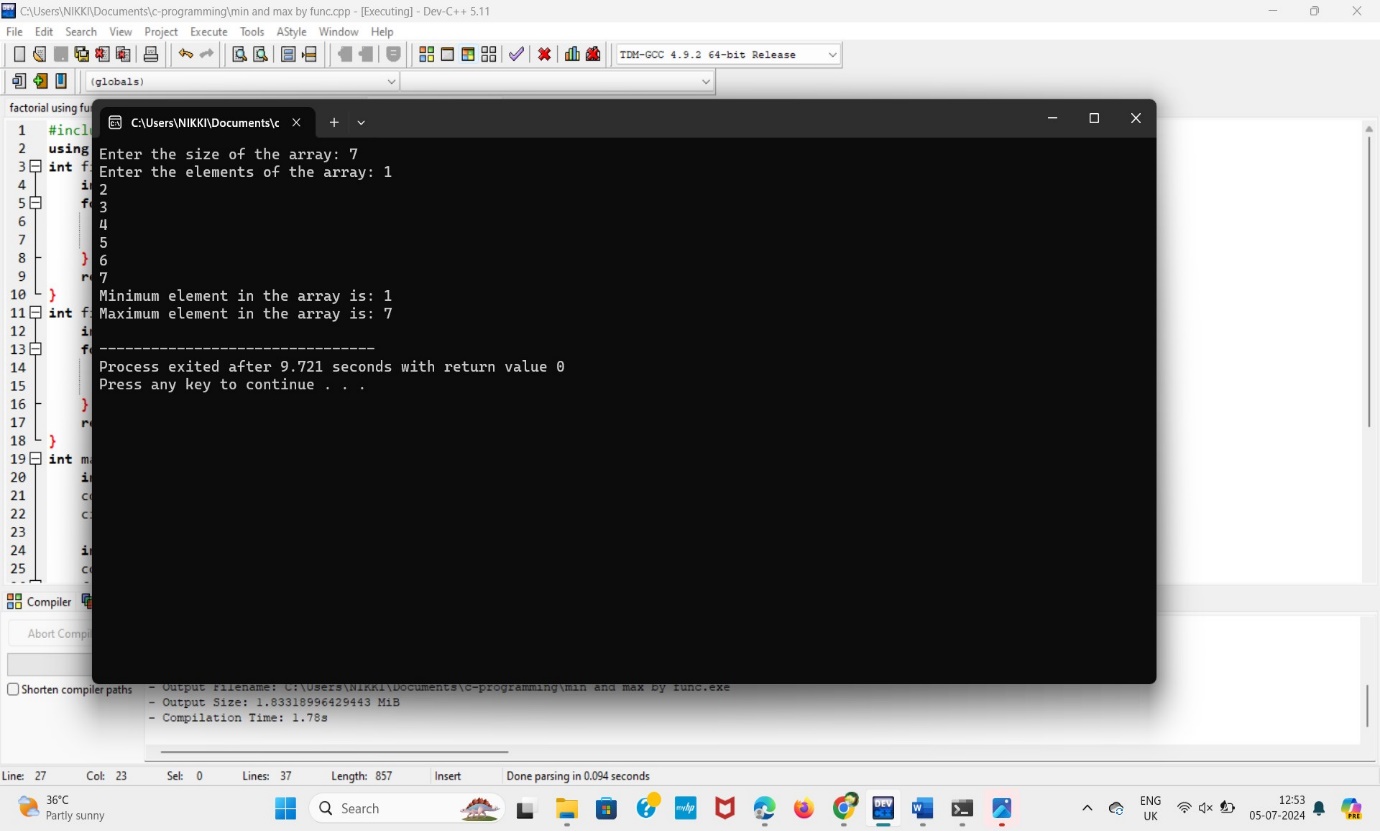
cout << "Minimum element in the array is: " << min << endl;

cout << "Maximum element in the array is: " << max << endl;

return 0;

}

**OUTPUT:**



**GCD of two number using function:**

#include <bits/stdc++.h>

using namespace std;

int gcd(int a, int b) {

int result = min(a, b);

while (result > 0) {

if (a % result == 0 && b % result == 0) {

break;

}

result--;

}

return result;

}

int main() {

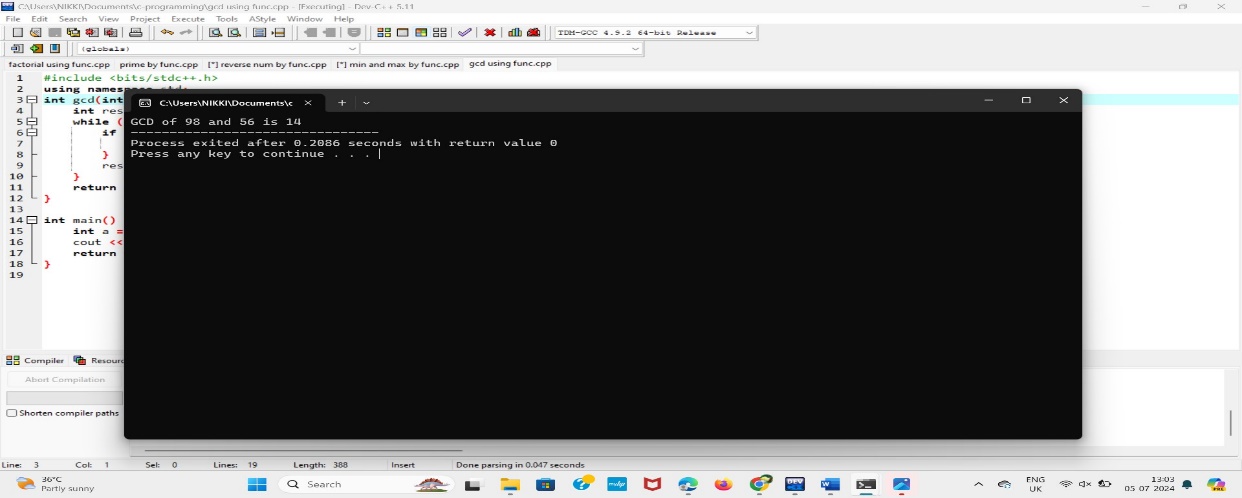
int a = 98, b = 56;

cout << "GCD of " << a << " and " << b << " is " << gcd(a, b);

return 0;

}

**OUTPUT:**



**Function to count the no.of elements in a string:**

#include <iostream>

#include <algorithm>

#include <string>

int main() {

std::string str = "helloworld";

char target = 'e';

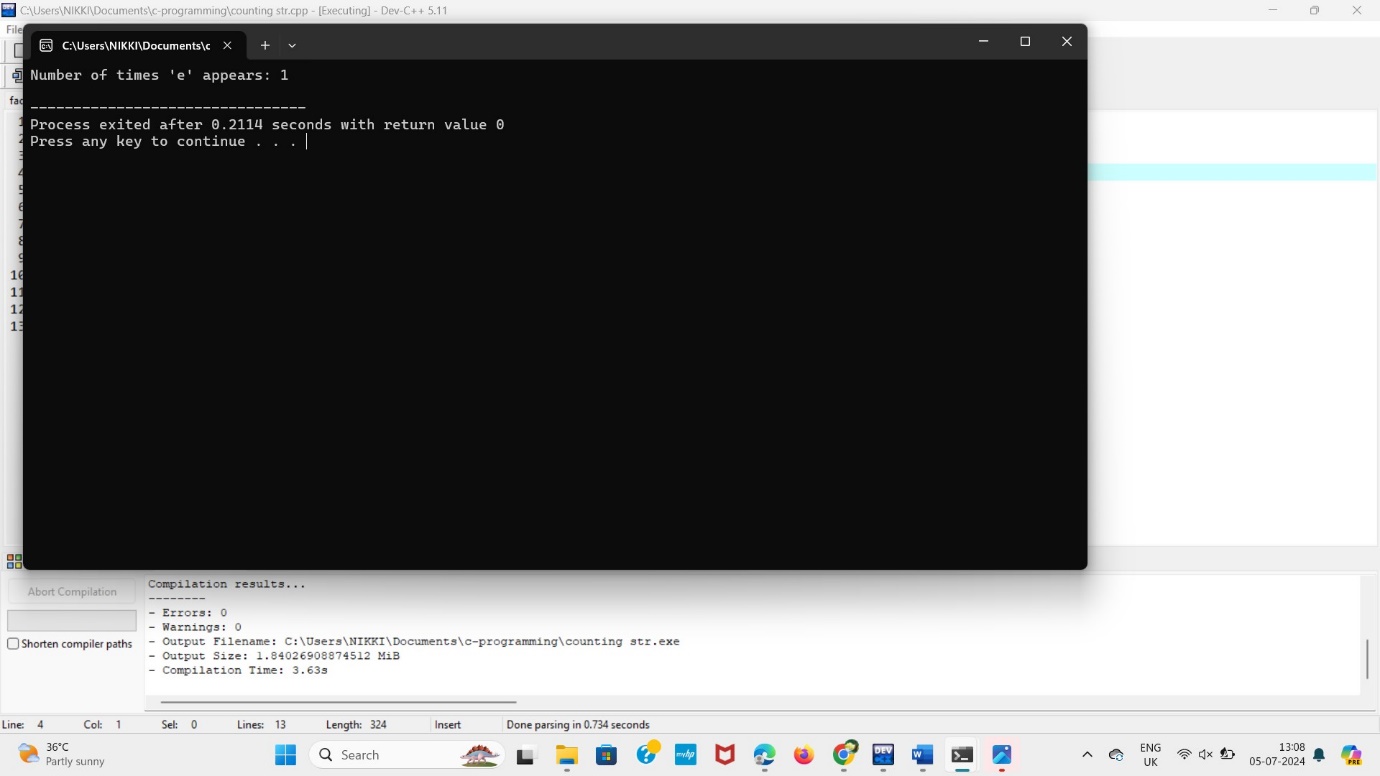
int occurrences = std::count(str.begin(), str.end(), target);

std::cout << "Number of times '" << target << "' appears: " << occurrences << std::endl;

return 0;

}

**OUTPUT:**



**Convert Celsius and Fahrenheit using function:**

#include <iostream>

using namespace std;

int main() {

float fahrenheit, celsius;

cout << "Enter the temperature in Celsius: ";

cin >> celsius;

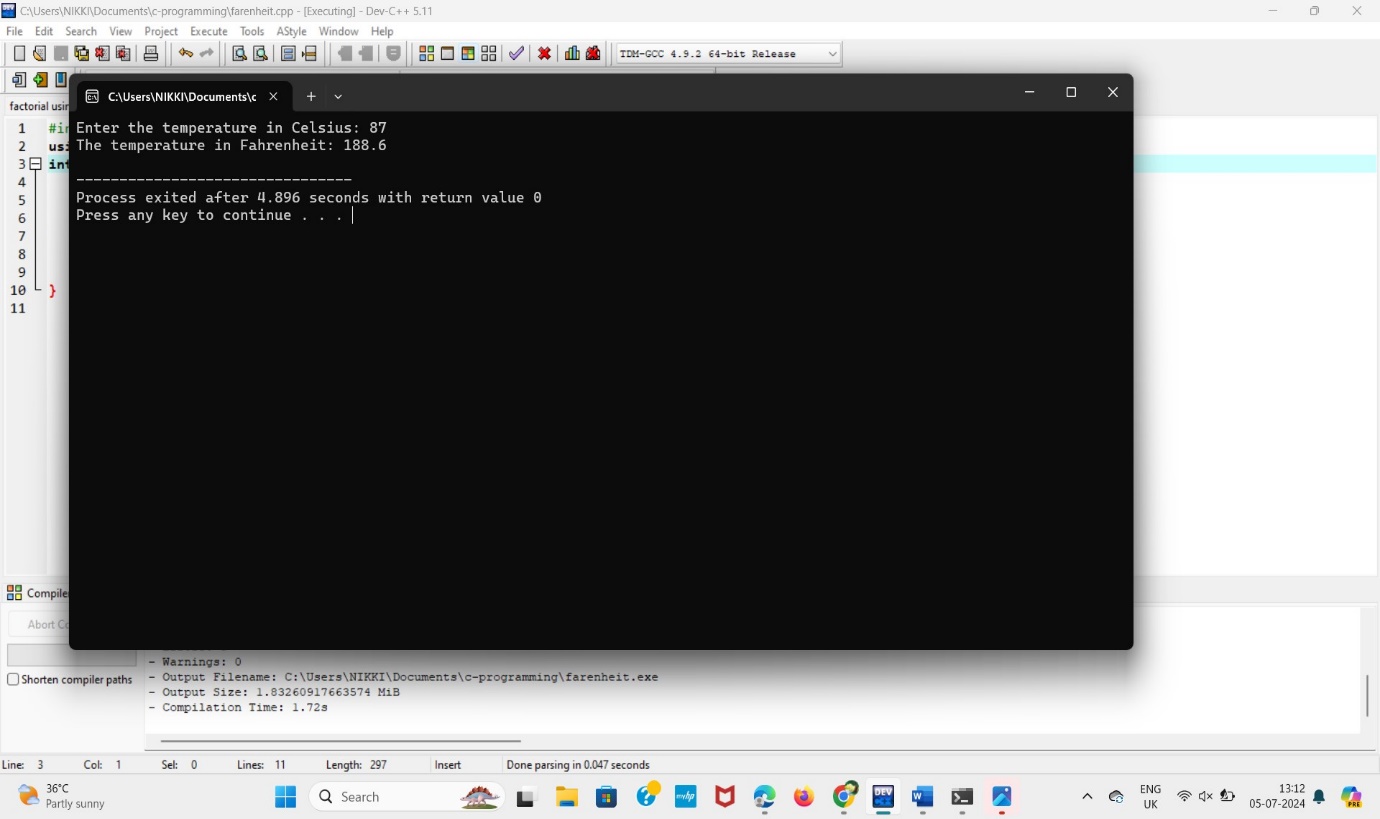
fahrenheit = (celsius \* 9.0) / 5.0 + 32;

cout << "The temperature in Fahrenheit: " << fahrenheit << endl;

return 0;

}

**OUTPUT:**



**The area of a circle using function:**

#include <iostream>

const double pi = 3.14159265358979323846;

double calculateArea(double radius) {

return pi \* radius \* radius;

}

int main() {

double radius;

std::cout << "Please enter the radius of the circle: ";

std::cin >> radius;

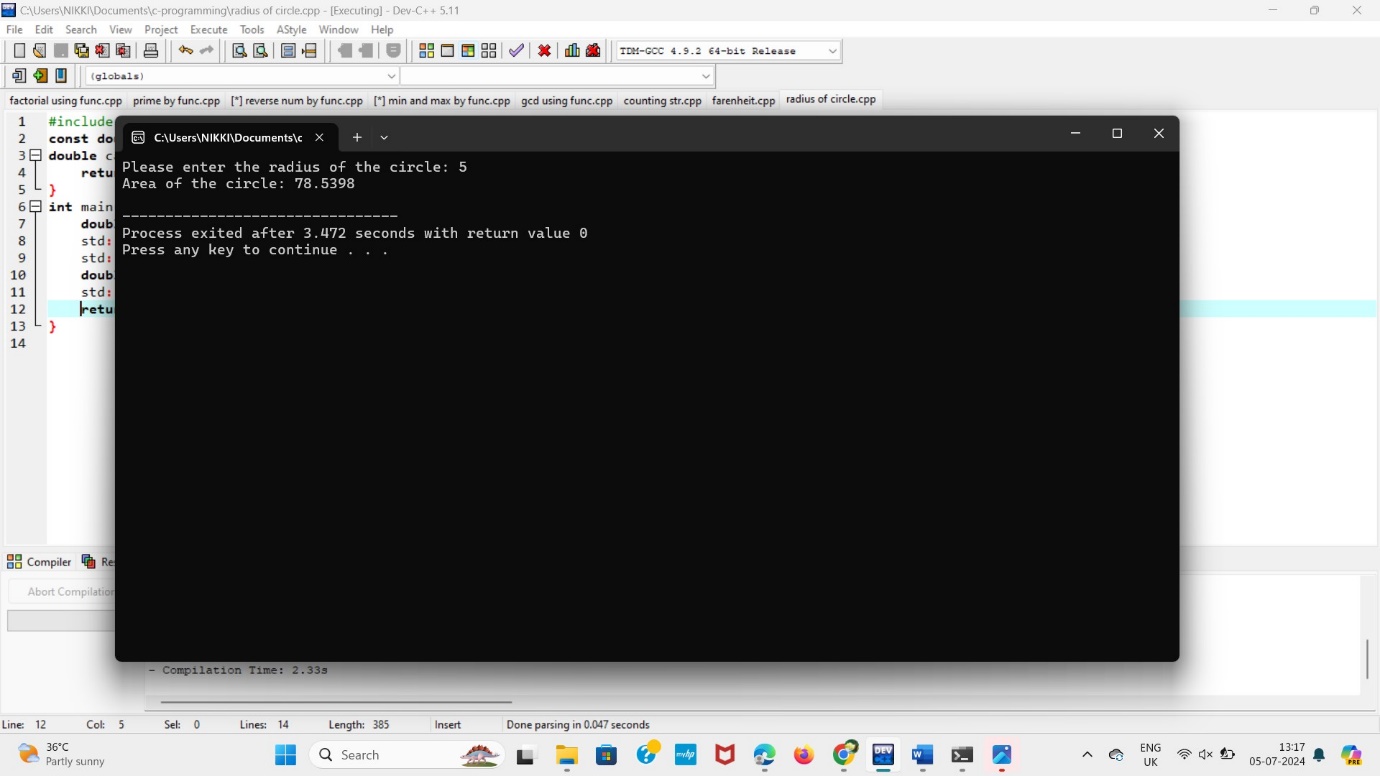
double area = calculateArea(radius);

std::cout << "Area of the circle: " << area << std::endl;

return 0;

}

**OUTPUT:**



**Check whether the string is palindrome or not:**

#include <bits/stdc++.h>

using namespace std;

string isPalindrome(string S)

{

string P = S;

reverse(P.begin(), P.end());

if (S == P) {

return "Yes";

}

else {

return "No";

}

}

int main()

{

string S = "ABCDCBA";

cout << isPalindrome(S);

return 0;

}

**OUTPUT:**

