21.GCD of two numbers using do-while loop.

Program:

#include <iostream>

using namespace std;

int main() {

int num1, num2;

cout << "Enter two positive integers: ";

cin >> num1 >> num2;

// Ensure num1 is the greater number

if (num2 > num1) {

int temp = num1;

num1 = num2;

num2 = temp;

}

int gcd = num2; // Start with the smaller number as the initial GCD candidate

do {

if (num1 % gcd == 0 && num2 % gcd == 0) {

break; // If both numbers are divisible by gcd, we have found the GCD

}

gcd--; // Otherwise, decrease gcd and try again

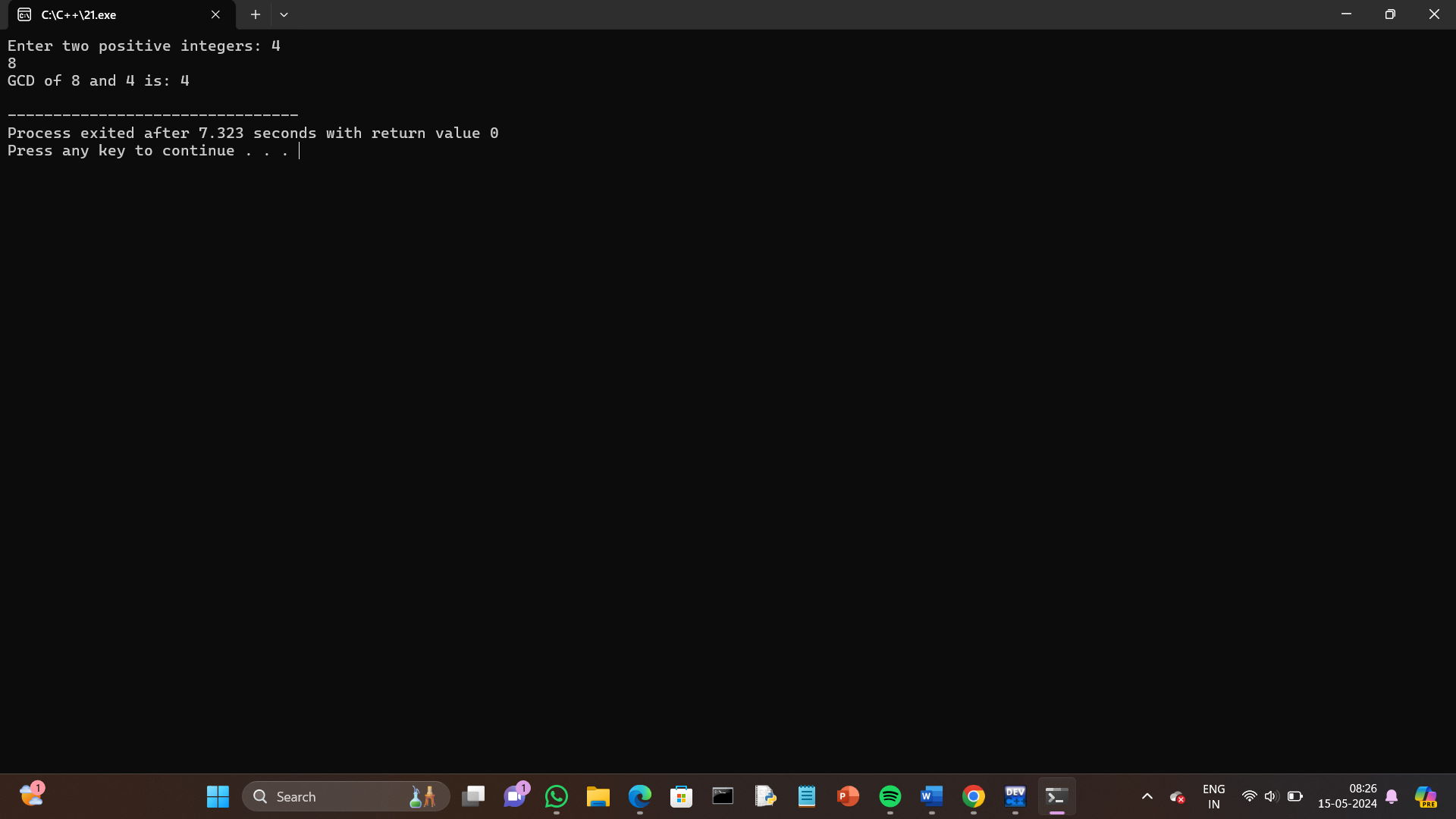
} while (gcd > 0);

cout << "GCD of " << num1 << " and " << num2 << " is: " << gcd << endl;

return 0;

}

Output:



22.Check whether the number is perfect or not.

Program:

#include <iostream>

using namespace std;

int main() {

int num, sum = 0;

cout << "Enter a positive integer: ";

cin >> num;

// Calculate the sum of proper divisors of the number

for (int i = 1; i <= num / 2; ++i) {

if (num % i == 0) {

sum += i;

}

}

// Check if the sum of divisors is equal to the number

if (sum == num && num != 0) {

cout << num << " is a perfect number." << endl;

} else {

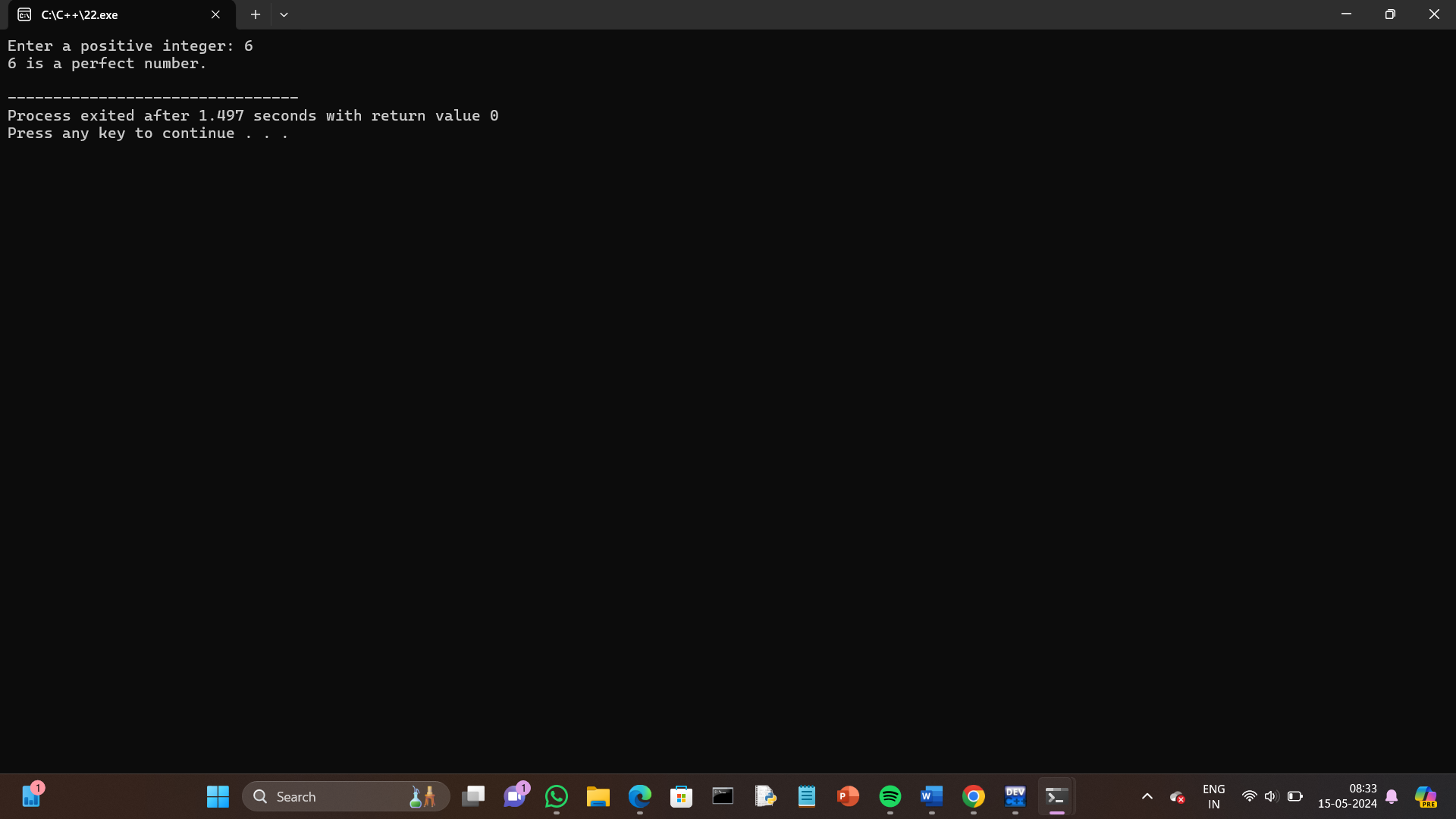
cout << num << " is not a perfect number." << endl;

}

return 0;

}

Output:



23.Armstrong number.

Program:

#include <iostream>

#include <cmath> // For pow function

using namespace std;

int main() {

int num, originalNum, remainder, result = 0, n = 0;

cout << "Enter an integer: ";

cin >> num;

originalNum = num;

// Count the number of digits

while (originalNum != 0) {

originalNum /= 10;

++n;

}

originalNum = num;

// Compute the sum of each digit raised to the power of n

while (originalNum != 0) {

remainder = originalNum % 10;

result += pow(remainder, n);

originalNum /= 10;

}

// Check if the result is equal to the original number

if (result == num) {

cout << num << " is an Armstrong number." << endl;

} else {

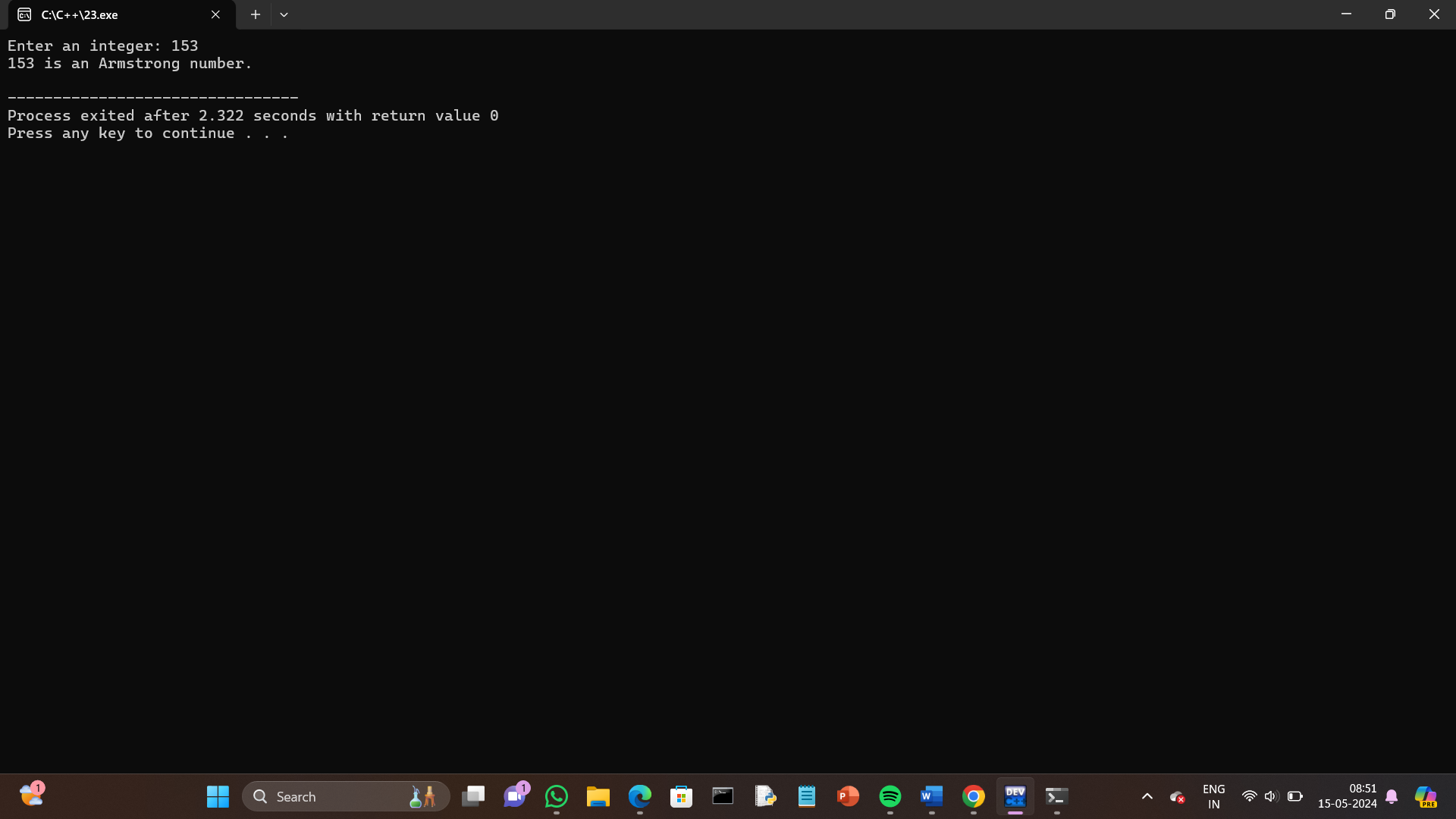
cout << num << " is not an Armstrong number." << endl;

}

return 0;

}

Output:



24.Harshad number.

Program:

#include <iostream>

using namespace std;

int main() {

int num, originalNum, remainder, sum = 0;

cout << "Enter an integer: ";

cin >> num;

originalNum = num;

// Calculate the sum of digits

while (originalNum != 0) {

remainder = originalNum % 10;

sum += remainder;

originalNum /= 10;

}

// Check if the number is divisible by the sum of its digits

if (num % sum == 0) {

cout << num << " is a Harshad number." << endl;

} else {

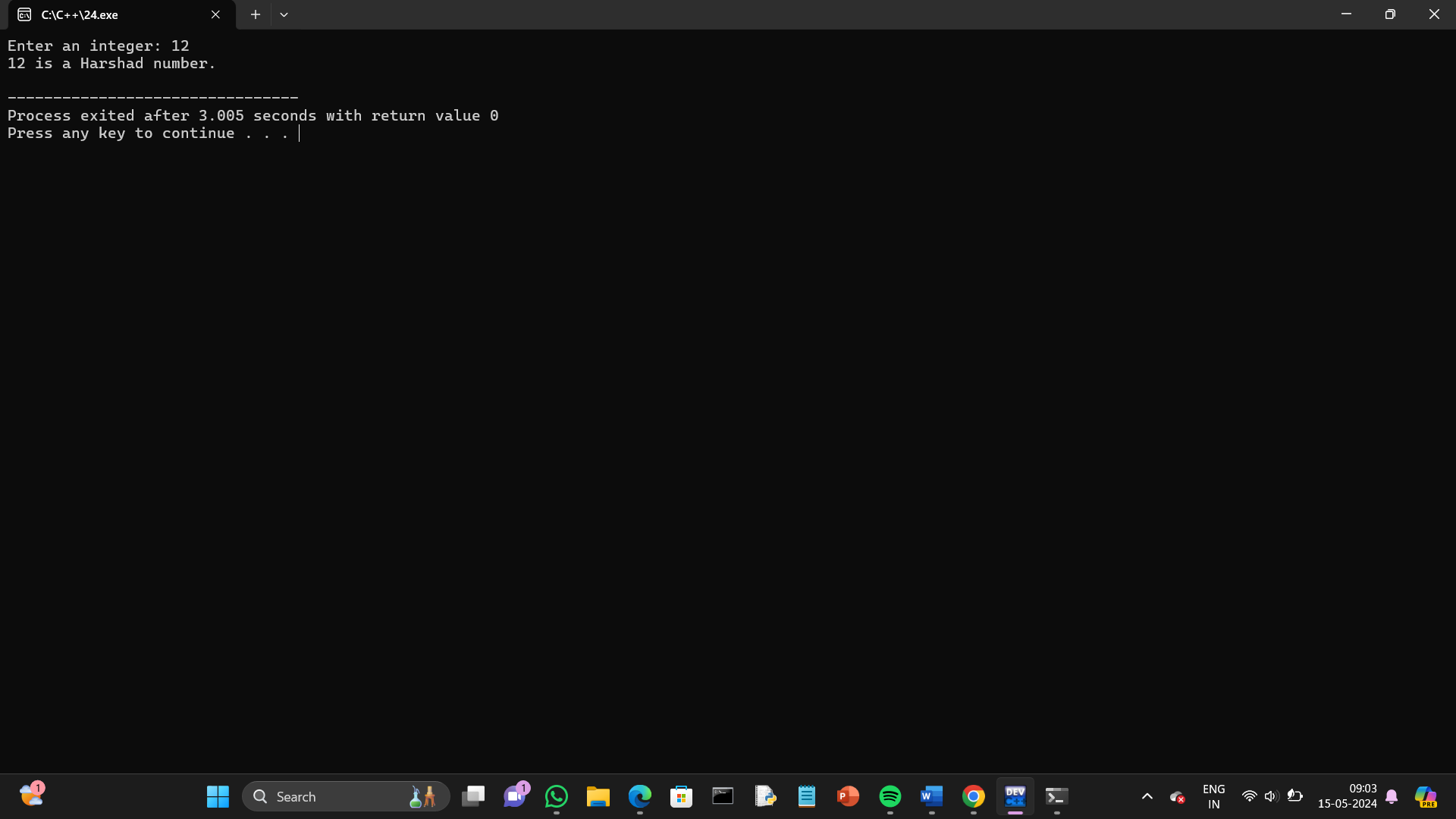
cout << num << " is not a Harshad number." << endl;

}

return 0;

}

Output:



25.Happy number.

Program:

#include <iostream>

using namespace std;

int squareSumOfDigits(int num) {

int sum = 0;

while (num > 0) {

int digit = num % 10;

sum += digit \* digit;

num /= 10;

}

return sum;

}

bool isHappy(int num) {

int slow = num, fast = num;

do {

slow = squareSumOfDigits(slow);

fast = squareSumOfDigits(squareSumOfDigits(fast));

} while (slow != fast);

return slow == 1;

}

int main() {

int num;

cout << "Enter a number: ";

cin >> num;

if (isHappy(num)) {

cout << num << " is a happy number." << endl;

} else {

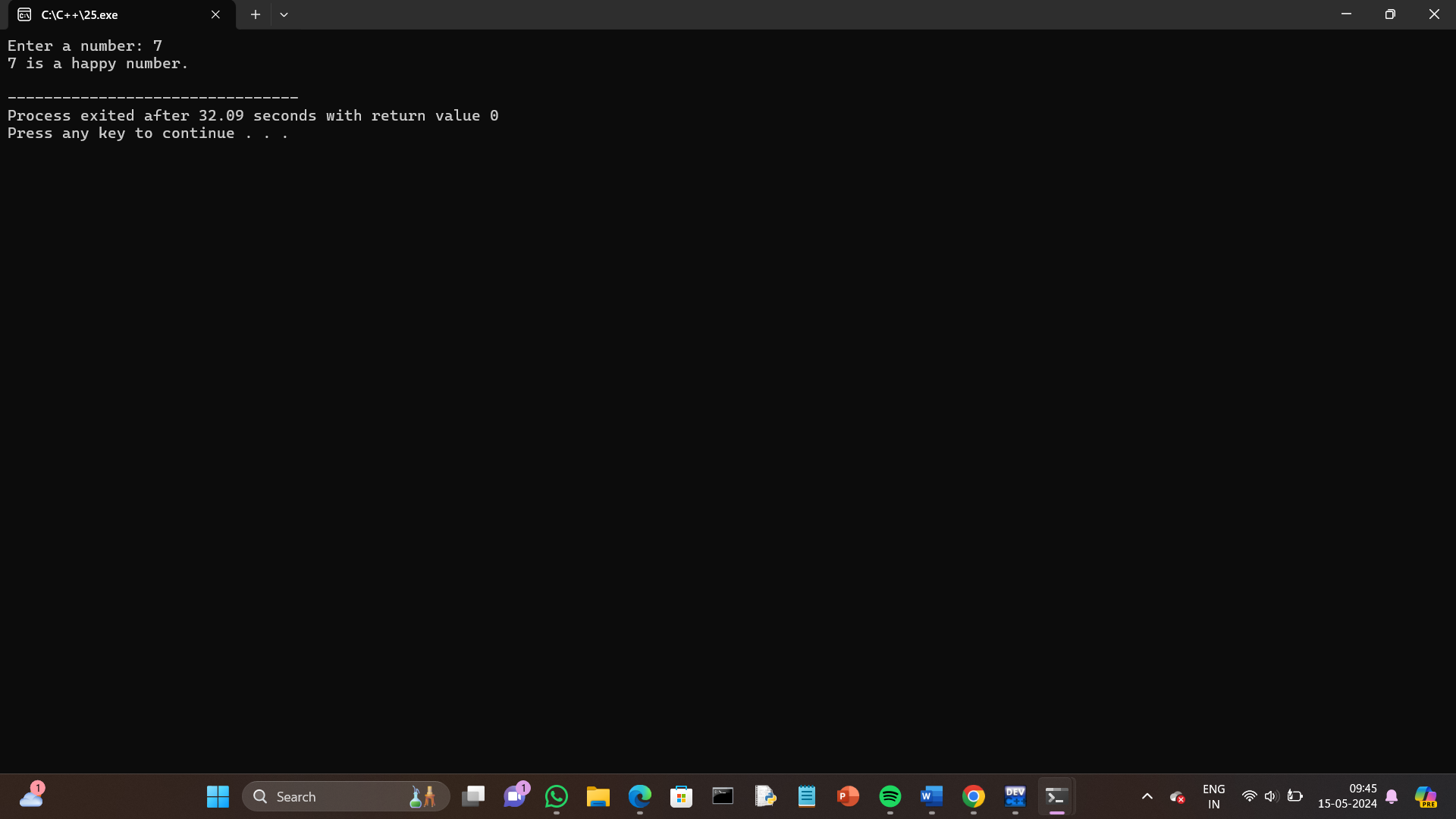
cout << num << " is not a happy number." << endl;

}

return 0;

}

Output:



26.strong number.

Program:

#include <iostream>

using namespace std;

// Function to calculate factorial of a number

int factorial(int n) {

int fact = 1;

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

fact \*= i;

}

return fact;

}

int main() {

int num, originalNum, remainder, sum = 0;

cout << "Enter an integer: ";

cin >> num;

originalNum = num;

// Calculate the sum of the factorial of the digits

while (originalNum != 0) {

remainder = originalNum % 10;

sum += factorial(remainder);

originalNum /= 10;

}

// Check if the sum of factorials is equal to the original number

if (sum == num) {

cout << num << " is a strong number." << endl;

} else {

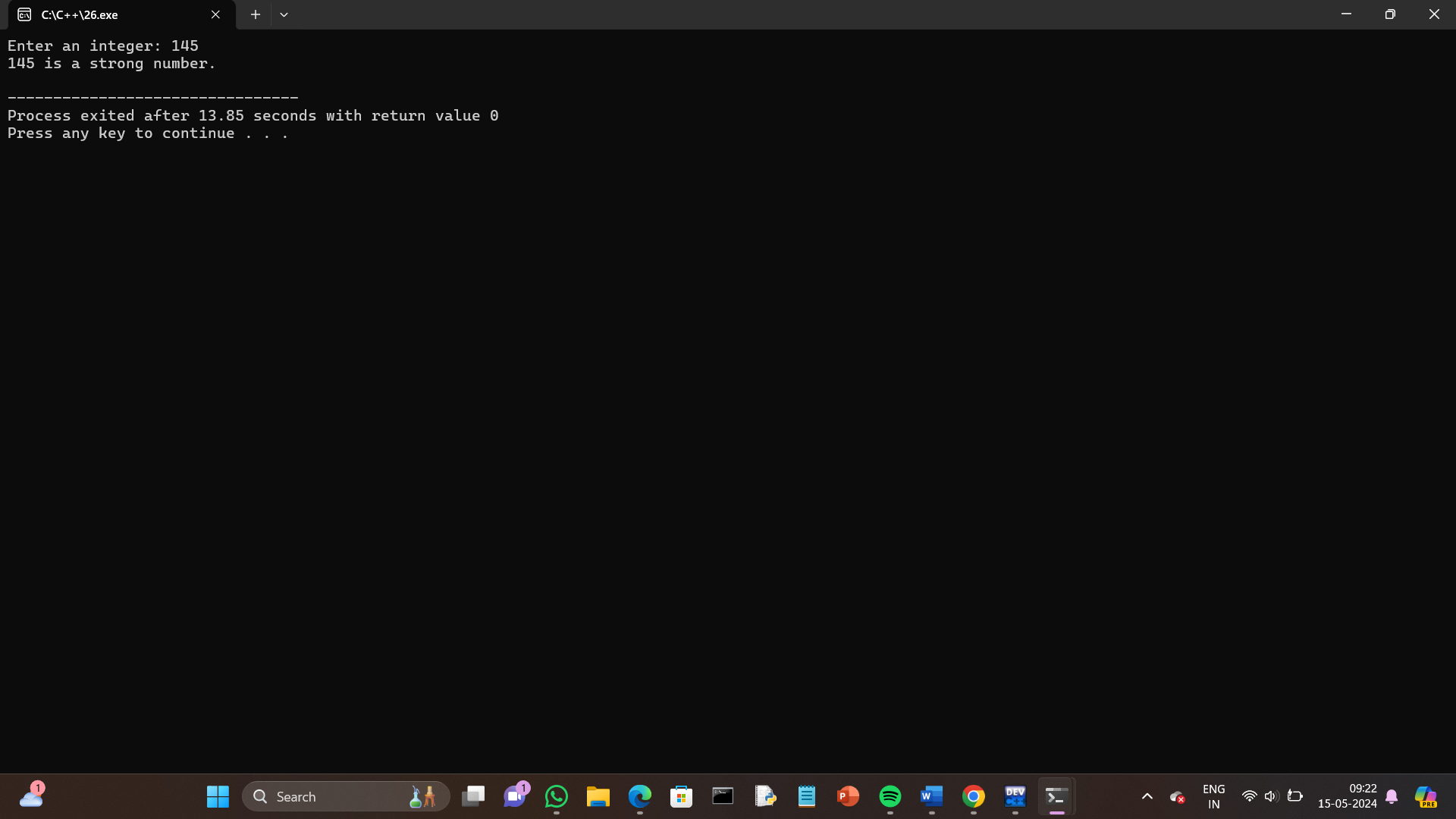
cout << num << " is not a strong number." << endl;

}

return 0;

}

Output:



27.buzz number.

Program:

#include <iostream>

using namespace std;

int main() {

int num;

cout << "Enter an integer: ";

cin >> num;

// Check if the number is divisible by 7 or ends with 7

if (num % 7 == 0 || num % 10 == 7) {

cout << num << " is a Buzz number." << endl;

} else {

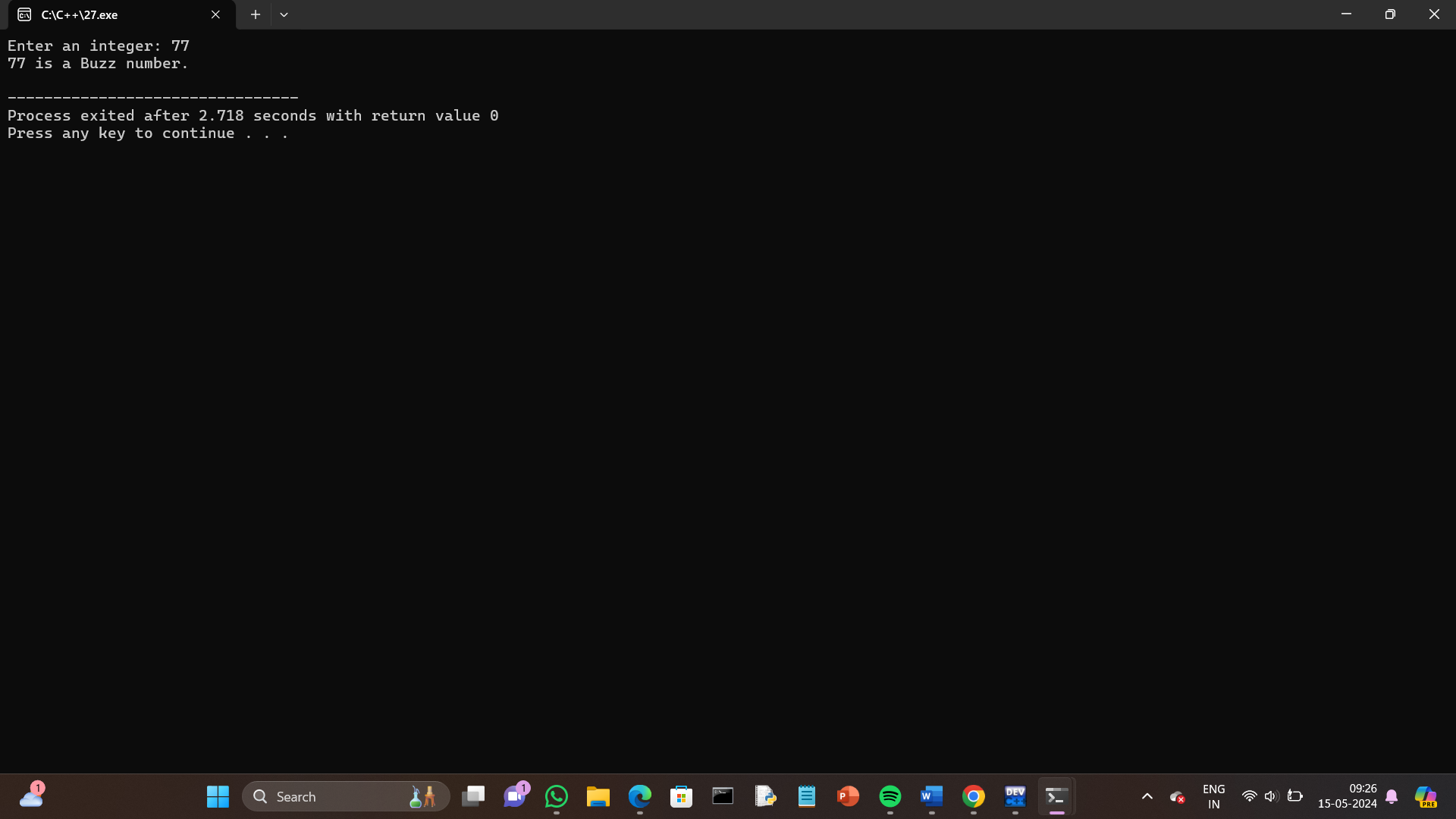
cout << num << " is not a Buzz number." << endl;

}

return 0;

}

Output:



28.neon number.

Program:

#include <iostream>

using namespace std;

int main() {

int num;

int square, digit, sum = 0;

// Input number from user

cout << "Enter a number: ";

cin >> num;

// Calculate square of the number

square = num \* num;

// Calculate sum of digits of the square

while (square != 0) {

digit = square % 10;

sum += digit;

square /= 10;

}

// Check if sum of digits is equal to the original number

if (sum == num) {

cout << num << " is a neon number." << endl;

} else {

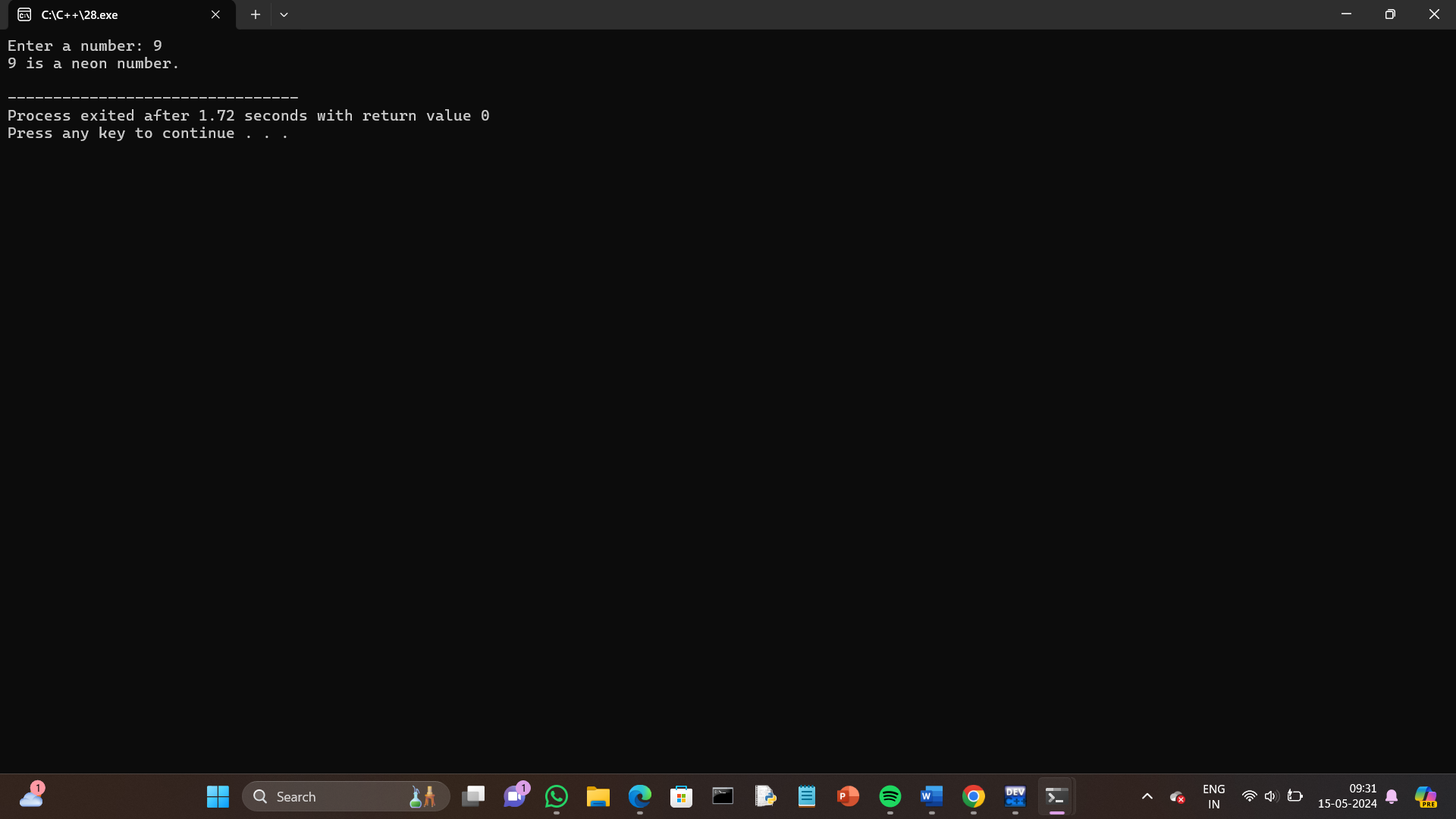
cout << num << " is not a neon number." << endl;

}

return 0;

}

Output:



29.abundant number.

Program:

#include <iostream>

using namespace std;

int main() {

int num, sum = 0;

// Input number from user

cout << "Enter a number: ";

cin >> num;

// Calculate the sum of proper divisors

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

if (num % i == 0) {

sum += i;

}

}

// Check if the sum is greater than the number itself

if (sum > num) {

cout << num << " is an abundant number." << endl;

} else {

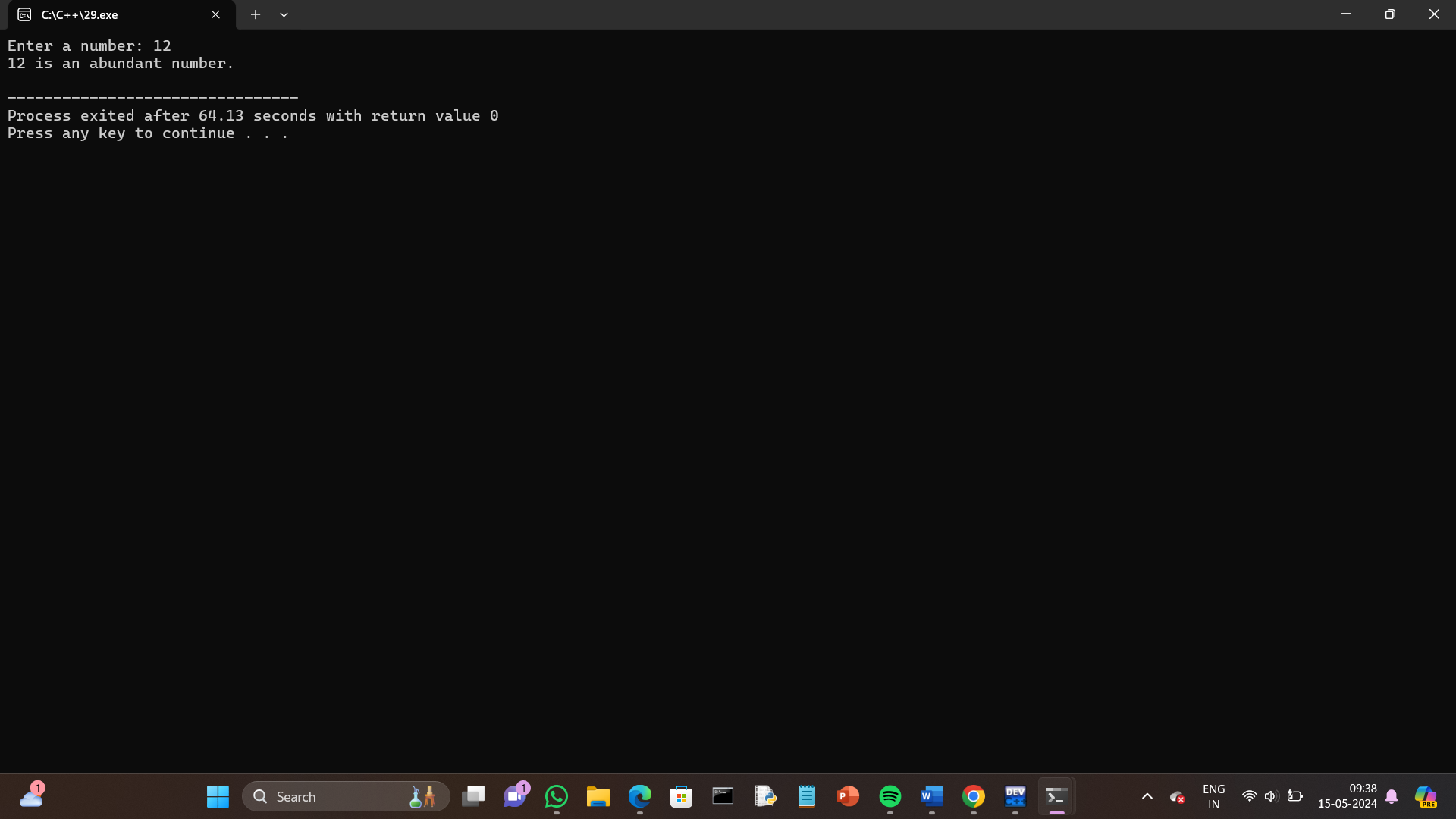
cout << num << " is not an abundant number." << endl;

}

return 0;

}

Output:



30.narcissistic number.

Program:

#include <iostream>

#include <cmath>

using namespace std;

int countDigits(int num) {

int count = 0;

while (num != 0) {

num /= 10;

count++;

}

return count;

}

int main() {

int num, originalNum, numDigits, sum = 0;

// Input number from user

cout << "Enter a number: ";

cin >> num;

originalNum = num;

numDigits = countDigits(num);

// Calculate the sum of digits raised to the power of numDigits

while (num != 0) {

int digit = num % 10;

sum += pow(digit, numDigits);

num /= 10;

}

// Check if the sum is equal to the original number

if (sum == originalNum) {

cout << originalNum << " is a narcissistic number." << endl;

} else {

cout << originalNum << " is not a narcissistic number." << endl;

}

return 0;

}

Output:

