| NAME | RAJ SANJAY JADHAV |
| --- | --- |
| CLASS | BE-3-CSE |
| BATCH | B |
| ROLL NO | 512024 |
| PRN | 2019033800129402 |

**PRACTICAL-2**

**NUMBERS,BRANCHES AND LOOPS**

**GIT REPOSITORY :** [**PRACTICAL-2**](https://github.com/Rajjadhav1710/.NET_Practical/tree/main/Practical-2)

**1) Manipulate Integral And Floating Point Numbers In C#.**

**CODE:**

using System;

namespace MyApplication{

class WorkingWithNumbers{

static void Main(string[] args){

Console.WriteLine("\n--------Working With Integers--------\n");

WorkWithIntegers();

Console.WriteLine("\n--------Check Order Precedence--------\n");

OrderPrecedence();

Console.WriteLine("\n--------Test Limits--------\n");

TestLimits();

Console.WriteLine("\n--------Work With Doubles--------\n");

WorkWithDoubles();

Console.WriteLine("\n--------Work With Decimal--------\n");

WorkWithDecimal();

Console.WriteLine("\n--------Calculate Area Of Circle Having R=2.50--------\n");

calcAreaOfCircle(2.50);

}

static void WorkWithIntegers(){

int a=18;

int b=6;

int c=a+b;

Console.WriteLine(c);

// subtraction

c = a - b;

Console.WriteLine(c);

// multiplication

c = a \* b;

Console.WriteLine(c);

// division

c = a / b;

Console.WriteLine(c);

// Mixing variables and constant numbers

c = a + b - 12 \* 17;

Console.WriteLine(c);

}

static void OrderPrecedence(){

int a = 5;

int b = 4;

int c = 2;

int d = a + b \* c;

Console.WriteLine(d);

d = (a + b) \* c;

Console.WriteLine(d);

d = (a + b) - 6 \* c + (12 \* 4) / 3 + 12;

Console.WriteLine(d);

int e = 7;

int f = 4;

int g = 3;

int h = (e + f) / g;

Console.WriteLine(h);

}

static void TestLimits(){

int a = 7;

int b = 4;

int c = 3;

int d = (a + b) / c;

int e = (a + b) % c;

Console.WriteLine($"quotient: {d}");

Console.WriteLine($"remainder: {e}");

int max = int.MaxValue;

int min = int.MinValue;

Console.WriteLine($"The range of integers is {min} to {max}");

int what = max + 3;

Console.WriteLine($"An example of overflow: {what}");

}

static void WorkWithDoubles(){

double a = 5;

double b = 4;

double c = 2;

double d = (a + b) / c;

Console.WriteLine(d);

double e = 19;

double f = 23;

double g = 8;

double h = (e + f) / g;

Console.WriteLine(h);

double max = double.MaxValue;

double min = double.MinValue;

Console.WriteLine($"The range of double is {min} to {max}");

double third = 1.0 / 3.0;

Console.WriteLine(third);

double num1=1234;

double num2=9999;

Console.WriteLine(num1/num2);

}

static void WorkWithDecimal(){

// The decimal type has a smaller range but greater precision than double

decimal min = decimal.MinValue;

decimal max = decimal.MaxValue;

Console.WriteLine($"The range of the decimal type is {min} to {max}");

double a = 1.0;

double b = 3.0;

Console.WriteLine(a / b);

decimal c = 1.0M;

decimal d = 3.0M;

Console.WriteLine(c / d);

// The M suffix on the numbers is how you indicate that a constant should use the decimal type. Otherwise, the compiler assumes the double type.

// math using the decimal type has more digits to the right of the decimal point.

}

static void calcAreaOfCircle(double r){

double area=Math.PI\*r\*r;

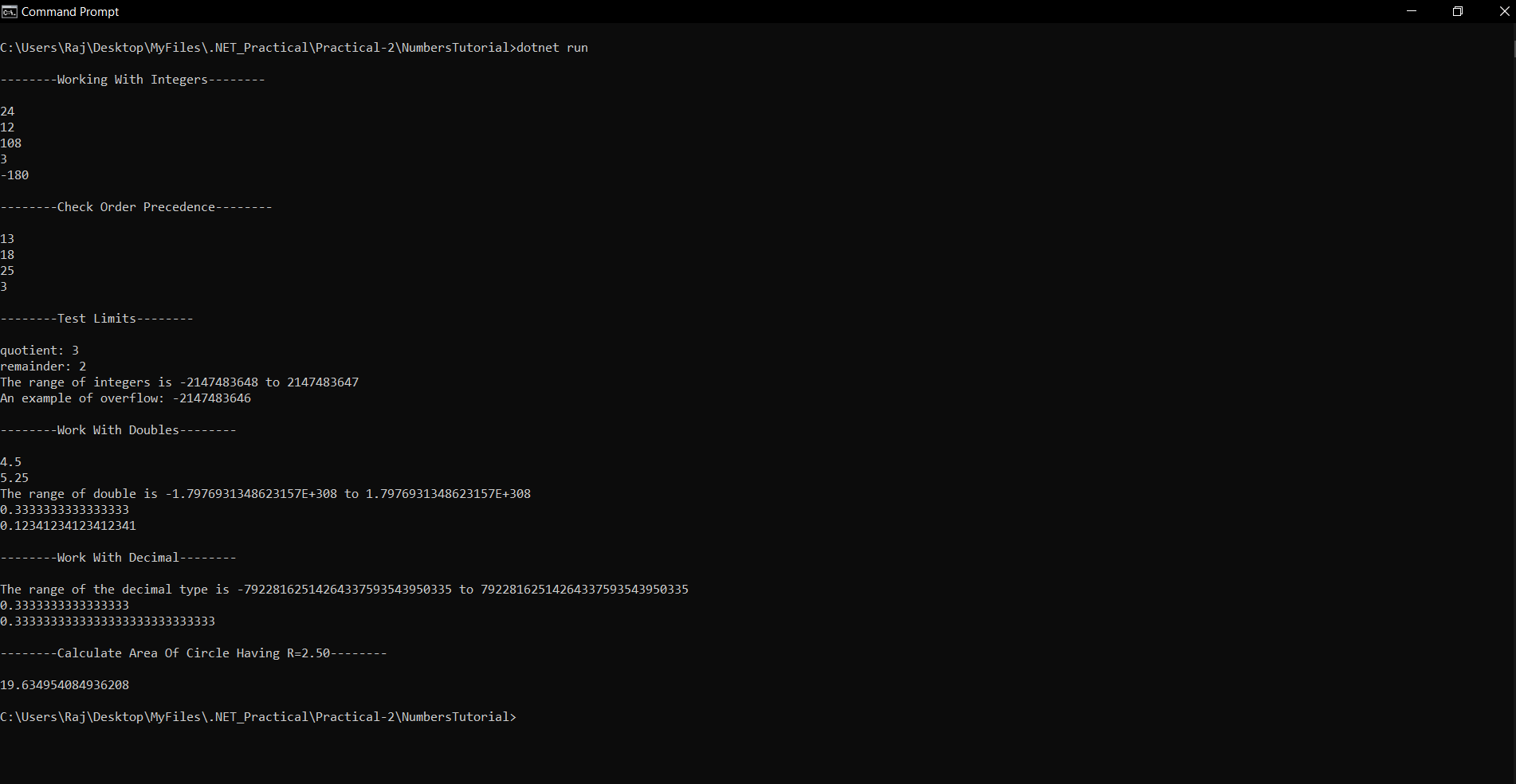
Console.WriteLine(area);

}

}

}

**OUTPUT:**

****

**2) Conditional Logic With Branch And Loop Statements.**

**CODE:**

**using System;**

**// ExploreIf();**

**namespace MyApplication**

**{**

**class WorkingWithBranchesAndLoops**

**{**

**static void Main(string[] args)**

**{**

**Console.WriteLine("\n------ Explore If..Else -------\n");**

**ExploreIf();**

**Console.WriteLine("\n------ Explore Loops -------\n");**

**ExploreLoops();**

**Console.WriteLine("\n------ find the sum of all integers 1 through 20 that are divisible by 3 -------\n");**

**findSum(1,20);**

**}**

**static void ExploreIf()**

**{**

**// Make decisions using the if statement**

**Console.WriteLine("\n\*If Statement With One Condition\n");**

**int a = 5;**

**int b = 3;**

**if (a + b > 10)**

**{**

**Console.WriteLine("The answer is greater than 10");**

**}**

**else**

**{**

**Console.WriteLine("The answer is not greater than 10");**

**}**

**Console.WriteLine("\n\*If Statement With Two Conditions Combined Using AND\n");**

**int c = 4;**

**if ((a + b + c > 10) && (a == b))**

**{**

**Console.WriteLine("The answer is greater than 10");**

**Console.WriteLine("And the first number is equal to the second");**

**}**

**else**

**{**

**Console.WriteLine("The answer is not greater than 10");**

**Console.WriteLine("Or the first number is not equal to the second");**

**}**

**// The == symbol tests for equality**

**Console.WriteLine("\n\*If Statement With Two Conditions Combined Using OR\n");**

**if ((a + b + c > 10) || (a == b))**

**{**

**Console.WriteLine("The answer is greater than 10");**

**Console.WriteLine("Or the first number is equal to the second");**

**}**

**else**

**{**

**Console.WriteLine("The answer is not greater than 10");**

**Console.WriteLine("And the first number is not equal to the second");**

**}**

**}**

**static void ExploreLoops()**

**{**

**Console.WriteLine("\n\*While Loop\n");**

**// Use loops to repeat operations**

**int counter = 0;**

**while (counter < 10)**

**{**

**Console.WriteLine($"Hello World! The counter is {counter}");**

**counter++;**

**}**

**// The while loop tests the condition before executing the code following the while.**

**Console.WriteLine("\n\*Do...While Loop\n");**

**// The do ... while loop executes the code first, and then checks the condition.**

**int counter2 = 0;**

**do**

**{**

**Console.WriteLine($"Hello World! The counter is {counter2}");**

**counter2++;**

**} while (counter2 < 10);**

**Console.WriteLine("\n\*For Loop\n");**

**for (int index = 0; index < 10; index++)**

**{**

**Console.WriteLine($"Hello World! The index is {index}");**

**}**

**Console.WriteLine("\n\*Matrix Using Nested Loops\n");**

**for (int row = 1; row < 11; row++)**

**{**

**for (char column = 'a'; column < 'k'; column++)**

**{**

**Console.WriteLine($"The cell is ({row}, {column})");**

**}**

**}**

**}**

**static void findSum(int i, int j)**

**{**

**int sum = 0;**

**for (int val = i; val <= j; val++)**

**{**

**if (val % 3 == 0)**

**sum += val;**

**}**

**Console.WriteLine(sum);**

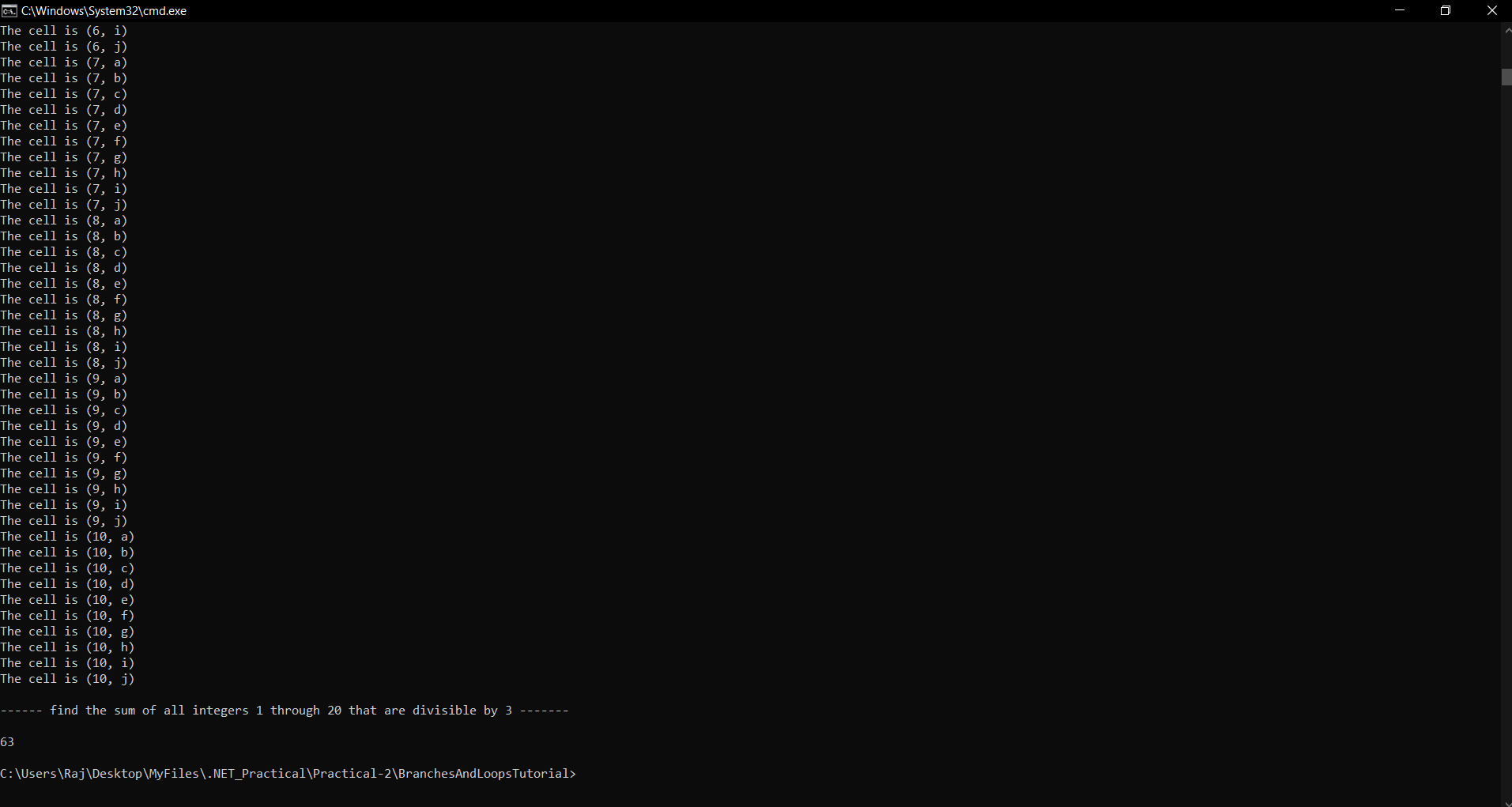
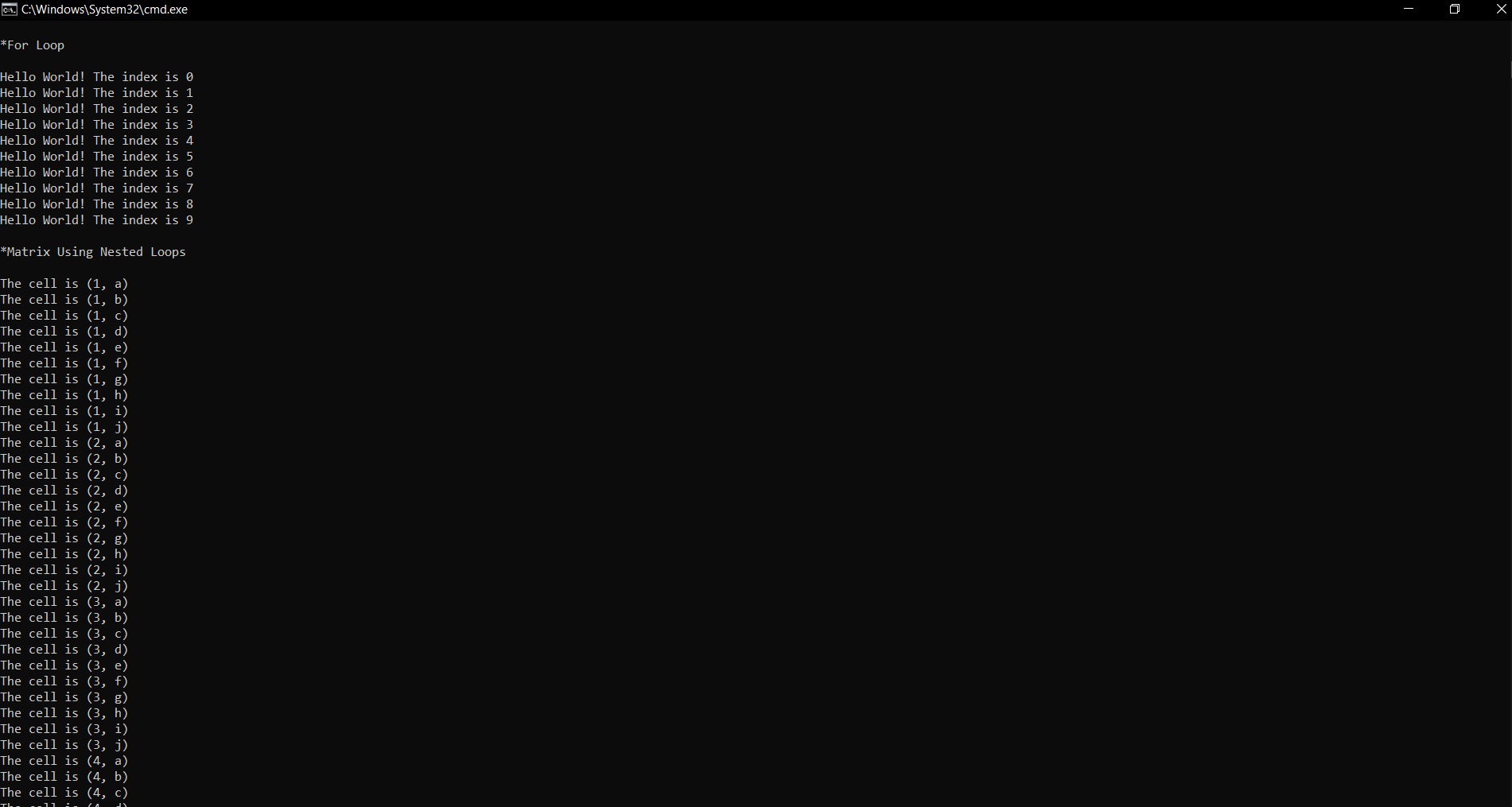
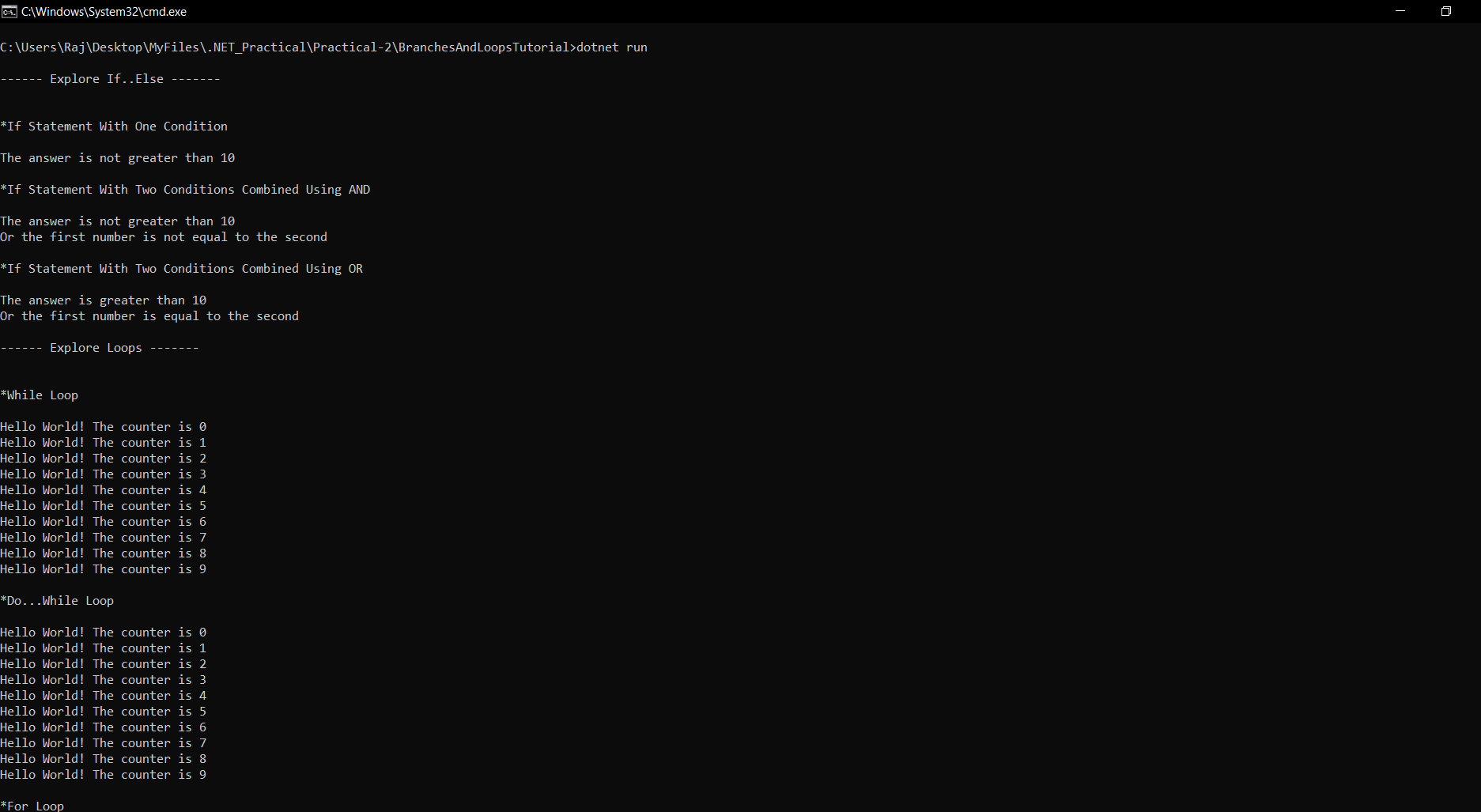
**}**

**}**

**}**

**// findSum(1, 20);**

**OUTPUT:**

****