|  |  |  |
| --- | --- | --- |
|  |  | **ISM 6225**  **Distributed Information systems** |

Assignment 1 – Programming Introduction

Primary objective: Develop familiarity with essential programming constructs

Secondary objective: develop comfort with using the IDE

*Estimated time: 15 hours*

## Introduction

Full-stack application development is an essential skill needed to succeed and even survive in business analytics and/ or information systems roles, especially as AI takes over many rudimentary tasks formerly performed by analysts. This assignment introduces the essential programming constructs such as variables, selection, loops, methods and arrays used to build such applications. Specifically, this assignment avoids the use of API methods and object-oriented programming. Those tasks are left for later assignments. This assignment also does not check for efficiency in program implementation. That is something you will develop over a lifetime in the profession. Rather, the focus is on simplicity, and creation of usable programs using industry best practices.

This is an individual assignment, to give every student the opportunity to develop the necessary skills to become a productive contributor to project teams in this class and beyond. One design goal for this assignment was to define an assignment that focused tightly on introductory programming structures, but very low probability that students would find ready-to-use solutions available online.

## Activity

In this assignment, you will define methods to do simple operations. Method signatures and hints are provided below. The methods are listed in the recommended sequence of development. A starter Program.cs file is included in the appendix.

## Submission

Push the code to GitHub and submit the URL. Also, get the output from a sample run that shows the use of all required methods, take the screenshot and upload/push it to GitHub.

## Grading scheme

Each method carries 1 point. There is also a bonus credit of 0.5 points for the overall assignment. You will be graded on the following aspects:

Logic (including appropriate organization of logic into methods) : 0.5

Handling all reasonable corner cases : 0.25

Descriptive comments explaining the logic to reviewer : 0.25

Self-reflection (learning from the assignment, and recommendations): 0.5 (Bonus credit)

## Method specifications

/\*

\* x – starting range, integer (int)

\* y – ending range, integer (int)

\*

\* summary : This method prints all the prime numbers between x and y

\* For example 5, 25 will print all the prime numbers between 5 and 25 i.e.

\* 5, 7, 11, 13, 17, 19, 23

\* Tip: Write a method isPrime() to compute if a number is prime or not.

\*

\* returns : N/A

\* return type : void

\*

\*/

public static void printPrimeNumbers(int x, int y)

/\*

\* para n – number of terms of the series, integer (int)

\*

\* summary : This method computes the series 1/2 – 2!/3 + 3!/4 – 4!/5 --- n \* where ! means factorial, i.e., 4! = 4\*3\*2\*1 = 24. Round off the results to

\* three decimal places.

\* Hint: Odd terms are all positive whereas even terms are all negative.

\* Tip: Write a method to compute factorial of n, call it whenever required.

\*

\* returns : result

\* return type : double

\*/

public static double getSeriesResult(int n)

/\*

\* n – non-negative number to be converted, integer (long)

\*

\* summary: This method converts a number from decimal (base 10) to binary (base 2).

\* Implementation: A number can be converted from decimal to binary in the following \* steps: 1)Divide it by 2. 2)Get the integer quotient for next iteration. 3)Get the \* remainder for binary digit. 4)Repeat the steps until the quotient is equal to 0.

\* For example, binary conversion for 15 is 1111

\*

\* Follow this link for detail explanation:

\* <https://www.rapidtables.com/convert/number/decimal-to-binary.html>

\*

\* returns : integer

\* return type : long

\*/

public static long decimalToBinary(long n)

/\*

\* n – non-negative number to be converted, integer (long)

\*

\* summary: This method converts a number from binary (base 2) to decimal (base 10).

\* Implementation: A number can be converted from binary to decimal in the following \* steps: 1)Start from the rightmost bit to the left. 2)Multiply each bit by 2^i where \* i starts from 0 and increases by 1 on iteration. 3)Add all the computations for the \* result.

\* For example, decimal conversion for 1101 = 1 \* 2^3 + 1 \* 2^2 + 0 \* 2^1 + 1 \* 2^0

\* = 1\*8 + 1\*4 + 0\*2 + 1\*1

\* = 8 + 4 + 0 + 1

\* = 13

\*

\* Follow this link for detail explanation:

\* <https://www.rapidtables.com/convert/number/binary-to-decimal.html>

\*

\* Tip: Write a method to compute 2^n, i.e, 2\*2\*2\*2---n. Call it whenever required. Do \* not use Math.Power()

\*

\* returns : integer

\* return type : long

\*/

public static long binaryToDecimal(long n)

/\*

\* n – number of lines for the pattern, integer (int)

\*

\* summary : This method prints a triangle using \*

\* For example n = 5 will display the output as:

\*

\*\*\*

\*\*\*\*\*

\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*

\*

\* returns : N/A

\* return type : void

\*/

public static void printTriangle(int n)

/\*

\* a – array of elements, integer (int)

\*

\* summary : This method computes the frequency of each element in the array

\* For example a = {1,2,3,2,2,1,3,2} will display the output as:

Number Frequency

1 2

2 4

3 2

\* returns : N/A

\* return type : void

\*/

public static void computeFrequency(int []a)

## Appendix: Program.cs

using System;  
  
namespace Assignment1\_S19  
{  
    class Program  
    {  
        public static void Main()  
        {  
            int a = 5, b = 15;  
            printPrimeNumbers(a, b);  
  
            int n1 = 5;  
            double r1 = getSeriesResult(n1);  
            Console.WriteLine("The sum of the series is: " + r1);

            long n2 = 15;  
            long r2 = decimalToBinary(n2);  
            Console.WriteLine("Binary conversion of the decimal number " + n2 + " is: " + r2);

            long n3 = 1111;  
            long r3 = binaryToDecimal(n3);  
            Console.WriteLine("Decimal conversion of the binary number " + n3 + " is: " + r3);

            int n4 = 5;  
            printTriangle(n4);  
  
            int[] arr = new int[] { 1, 2, 3, 2, 2, 1, 3, 2 };  
            computeFrequency(arr);

// write your self-reflection here as a comment

        }  
  
        public static void printPrimeNumbers(int x, int y)  
        {  
            try  
            {  
“using System;

namespace Printprimenumbers

{

class Program

{

public static void printPrimeNumbers()

{

{

int x, y; // we have created two integers x,y.

Console.WriteLine("Please enter a number to check if it is a prime number or not");//take input from the user

x = Int32.Parse(Console.ReadLine()); //insert that value to variable x by using ReadLine() function

for (y = 2; y < x; y++) //check wether x is greater than 2

{

if (x % y == 0) //check if x has any factors

{

Console.WriteLine("{0} is not a prime number", x);

break;

}

}

if (x == 1 || x<0)

{

Console.WriteLine("{0} is not a prime number", x);//display output

}

if (y == x) //if x is 2

{

Console.WriteLine("{0} is a prime number", x);

}

}

}

static void Main(string[] args)

{

try

{

printPrimeNumbers();//call method "printprimeNumbers()"

}

catch (Exception)//if user enters alphabets or symbols

{

Console.Write("Please enter only integers");//display

}

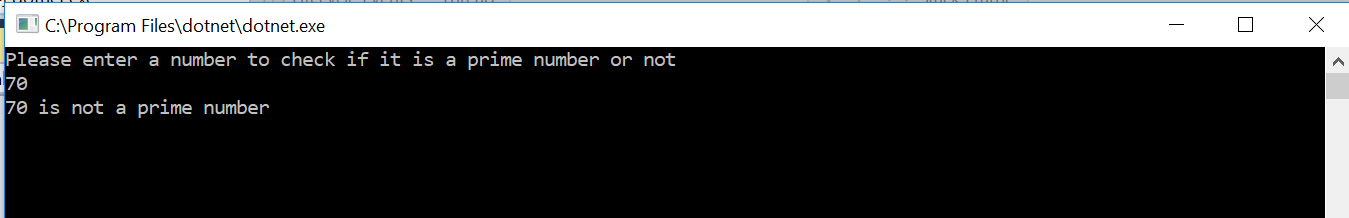
Console.ReadLine();//to hold output

}

} }

“

            }  
            catch  
            {  
                Console.WriteLine("Please enter only integers ");  
            }  
        }

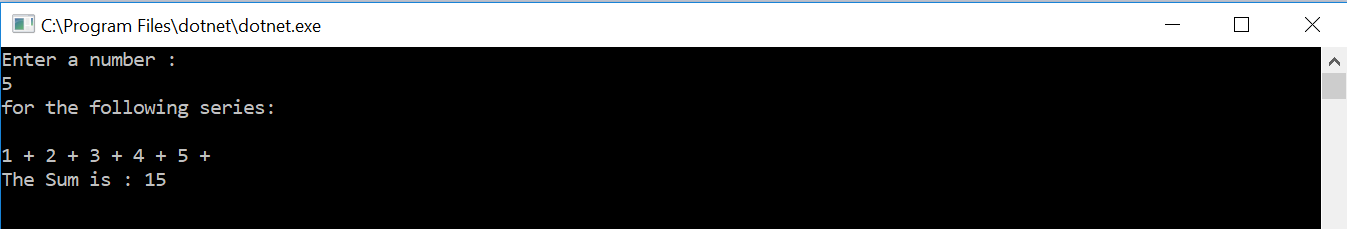


<https://github.com/saksham123/Printprimenumbers/tree/master/Printprimenumbers>

        public static double getSeriesResult(int n)  
        {  
            try  
            {

|  |
| --- |
| “using System; |
|  |  |
|  | namespace getseriesresult |
|  | { |
|  | class Program |
|  | { |
|  | public static double getSeriesResult()//method |
|  | { |
|  | int i, n, sum = 0, x = 1; |
|  | Console.WriteLine("Enter a number :"); // get number from user |
|  | n = Int32.Parse(Console.ReadLine()); // put the value in variable n |
|  | Console.Write("for the following series:"); |
|  | Console.Write(""); |
|  | Console.Write("\n\n"); |
|  |  |
|  | for (i = 1; i <= n; i++) |
|  | { |
|  | Console.Write("{0} + ", x); |
|  | sum = sum + x; //add the value in sum |
|  | x = (x) + 1;//increment the value of x till n |
|  | } |
|  | Console.Write("\nThe Sum is : {0}\n", sum);//display sum |
|  | Console.ReadLine(); |
|  | return 0; |
|  | } |
|  | static void Main(string[] args) |
|  | { |
|  | try |
|  | { |
|  | getSeriesResult();//call method |
|  | } |
|  | catch (Exception) |
|  | { |
|  | Console.Write("Please enter a whole number");//display if user enters anything other than whole number |
|  | } |
|  | Console.ReadLine(); |
|  | }//main end |
|  | }//class end” |
|  | } |

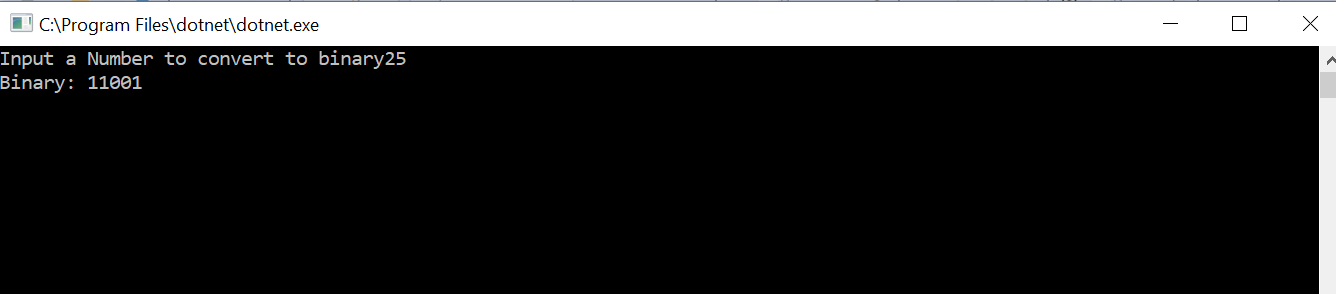
            }  
            catch  
            {  
                Console.WriteLine("Please enter a whole number”);  
            }  
  
            return 0;  
        }

  
  
<https://github.com/saksham123/getseriesresult/tree/master/getseriesresult>

 public static long decimalToBinary(long n)  
        {  
            try  
            {

|  |
| --- |
| “using System; |
|  |  |
|  | namespace solution3 |
|  | { |
|  | class Program |
|  | { |
|  | public static long decimalToBinary() |
|  | { |
|  | string x, result; |
|  |  |
|  | Console.Write("Input a Number to convert to binary");//input number from the user |
|  | x = Console.ReadLine();//put number into x |
|  | int y = Convert.ToInt32(x);//convert to integer |
|  | result = ""; |
|  | while (y>1)//while loop where num value is greater than 1 |
|  | { |
|  | int remainder = y % 2;//check remainder when divided by 2 |
|  | result = Convert.ToString(remainder) + result;//enter result from right to left |
|  | y /= 2;// change value of num to qutioent when num is divided by 2 |
|  | } |
|  | result = Convert.ToString(y) + result;//add num the last qutioent 1 to result |
|  | Console.WriteLine("Binary: {0}", result);//display the binary value |
|  | Console.ReadLine(); |
|  | return 0; |
|  | } |
|  | static void Main(string[] args) |
|  | { |
|  | try |
|  | { |
|  | decimalToBinary();//call the method |
|  | } |
|  | catch |
|  | { |
|  | Console.WriteLine("write a valid input"); |
|  | } |
|  | } |
|  | } |
|  | }” |

            }  
            catch  
            {  
                Console.WriteLine("Write a valid input");  
            }  
  
            return 0;  
        }

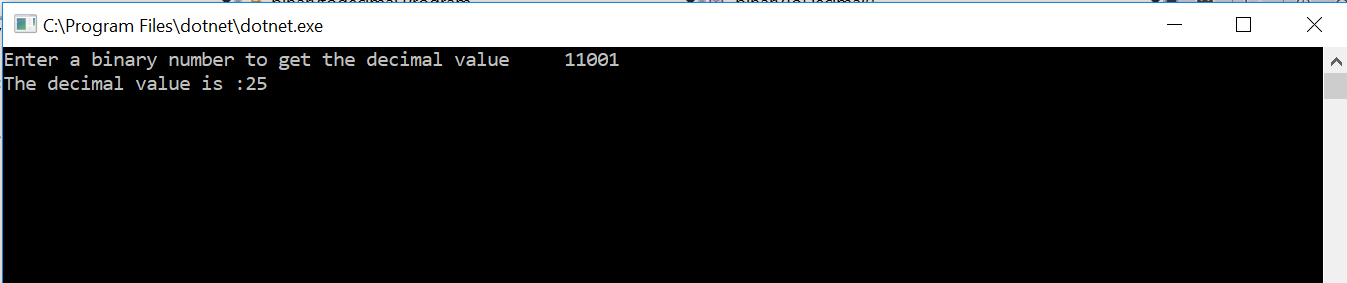


<https://github.com/saksham123/solution3/tree/master/solution3>

        public static long binaryToDecimal(long n)  
        {  
            try  
            {

|  |
| --- |
|  |
| “using System; |
|  |  |
|  | namespace binarytodecimal |
|  | { |
|  | class Program |
|  | { |
|  | public static long binaryToDecimal() |
|  | { |
|  | int x, result= 0, q = 1, r;//declare variables |
|  | Console.Write("Enter a binary number to get the decimal value ");//display message |
|  | x = int.Parse(Console.ReadLine());//input value to x |
|  | while (x > 0)//while loop if x is positive |
|  | { |
|  | r = x % 10;//get remainder |
|  | result = result + r \* q;//start by multiplying remainder with 1 |
|  | x = x / 10;//get quotient |
|  | q = q \* 2;//multiply by 2 |
|  | } |
|  | Console.Write("The decimal value is :"+ result);//print value of result |
|  | Console.ReadLine();//hold output |
|  | return 0; |
|  | } |
|  | static void Main(string[] args) |
|  | { |
|  | try |
|  | { |
|  | binaryToDecimal(); |
|  | } |
|  | catch |
|  | { |
|  | Console.WriteLine("Please enter a valid binary value"); |
|  | } |
|  | } |
|  | } |
|  | } |
|  | “ |
|  |  |

            }  
            catch  
            {  
                Console.WriteLine("Please enter a valid binary value”);  
            }  
  
            return 0;  
        }

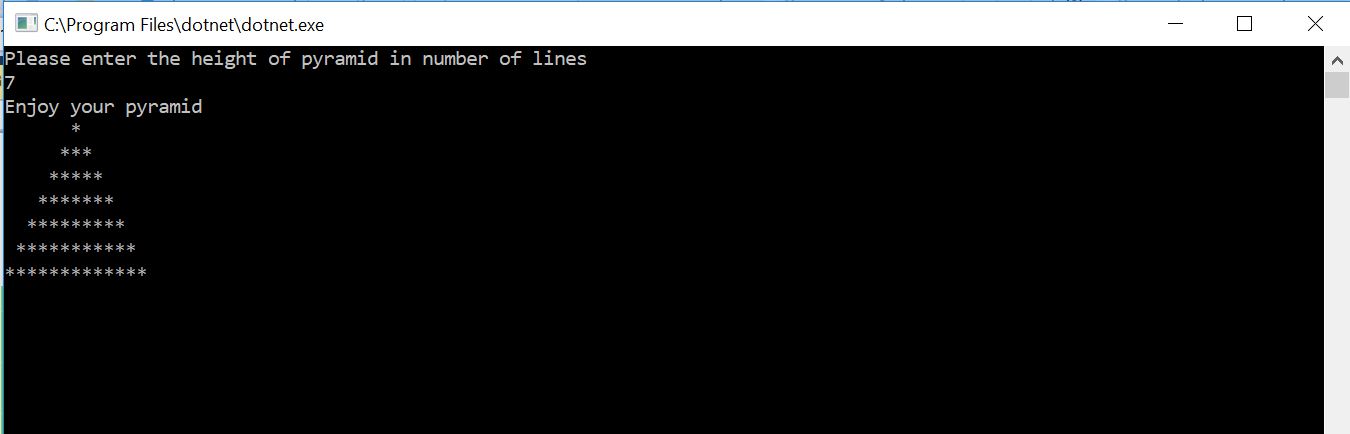


<https://github.com/saksham123/binarytodecimal/tree/master/binarytodecimal>

        public static void printTriangle(int n)  
        {  
            try  
            {

|  |
| --- |
| “using System; |
|  |  |
|  | namespace solution5 |
|  | { |
|  | class Program |
|  | { |
|  | public static void printPyramid() |
|  | { |
|  | int n, x, blank; |
|  | Console.WriteLine("Please enter the height of pyramid in number of lines");//input value from user for number of lines |
|  | n = Int32.Parse(Console.ReadLine()); // convert to integer |
|  | Console.WriteLine("Enjoy your pyramid");//display message |
|  | for (int i = 1; i <= n; i++) //till height of pyramid |
|  | { |
|  | for (blank = 1; blank <= (n - i); blank++) // insert space |
|  | Console.Write(" "); |
|  | for (x = 1; x <= i; x++) //increment value by 1 each time |
|  | Console.Write('\*'); |
|  | for (x = (i - 1); x >= 1; x--) //decrement the value by 1 |
|  | Console.Write("\*"); |
|  | Console.WriteLine(); |
|  | } |
|  | } |
|  | static void Main(string[] args) |
|  | { |
|  | try |
|  | { |
|  | printPyramid();//call the method |
|  | } |
|  | catch (Exception) |
|  | { |
|  | Console.Write("Please enter a valid input"); |
|  | } |
|  | Console.ReadLine(); |
|  | } |
|  | } |
|  | }” |

            }  
            catch  
            {  
                Console.WriteLine("Please enter a valid input");  
            }  
        }

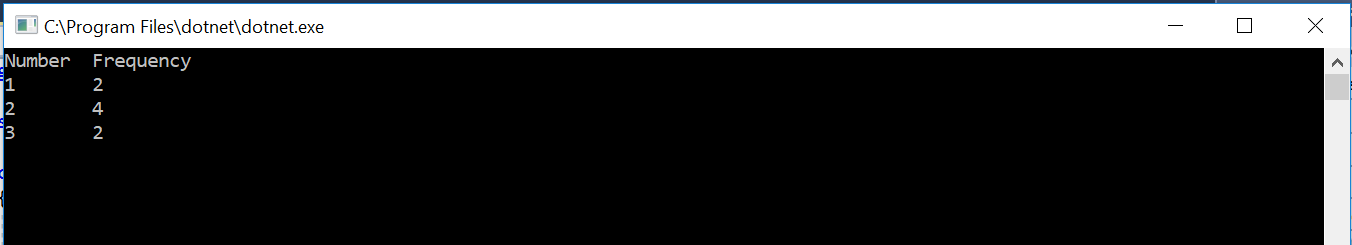


<https://github.com/saksham123/solution5/tree/master/solution5>

        public static void computeFrequency(int[] a)  
        {  
            try  
            {

|  |
| --- |
| “using System; |
|  |  |
|  | namespace solution6 |
|  | { |
|  | class Program |
|  | { |
|  | static void Main(string[] args) |
|  | { |
|  | int[] arr = new int[] { 1, 2, 3, 2, 2, 1, 3, 2 };//dataset given in example |
|  | computeFrequency(arr);//call method |
|  | Console.ReadLine();//hold the output screen |
|  | } |
|  | public static void computeFrequency(int[] a) |
|  | { |
|  | Console.WriteLine("Number Frequency"); //to differentiate between value and frequency |
|  | try |
|  | { |
|  | int x = a.Length; |
|  | Boolean[] counted = new Boolean[x]; //array to check that number is counted |
|  | for (int i = 0; i < x; i++) |
|  | { |
|  | if (counted[i] == true) |
|  | continue; //continue if number is counted previously |
|  | int freq = 1; // define freq to 1 |
|  | for (int j = i + 1; j < x; j++) //count freq of a number in an array |
|  | { |
|  | if (a[i] == a[j]) //check if value is present in j as well |
|  | { |
|  | counted[j] = true; |
|  | freq++;// increment freq by 1 |
|  | } |
|  |  |
|  | } |
|  | Console.WriteLine(a[i] + " " + freq);//display number and their frequency |
|  | } |
|  | } |
|  | catch |
|  | { |
|  | Console.WriteLine("exception try again");//error message |
|  | } |
|  | } |
|  | } |
|  | }” |

            }  
            catch  
            {  
                Console.WriteLine("Exception try again");  
            }  
        }  
    }  
}



<https://github.com/saksham123/solution6/tree/master/solution6>