CSC180 Assignment #07

**#1** This exercise is based on the existing code. Complete the following steps:  
 1) Test run the program.  
 2) Summarize the whole program and explain the following lines 11, 40 and 48.

The Shape program calculates the area of either a square or a circle using a user provided length or radius. Line 11, public abstract double Area(); is used as an unimplemented method because the shape, and thus, the formula for the area cannot be determined. Line 40 implements the abstract method and returns the value of 314.16 to Area() so it can be called in the Main method. Line 48, Shape[] shapes = { new Square(10), new Circle(10) };, counts the number of Shapes in an array [2] and defines the kind of shapes as Circle and Square.

3) Add a Rectangle class that inherits Shape, and test it in Program.cs.  
 4) Extra credit: add a Triangle class and test it in Program.cs.

using System;

public abstract class Shape

{

public abstract double Area();

public static double GetArea(Shape shape)

{

return shape.Area();

}

}

public class Square : Shape

{

private double size;

public Square(double length)

{

this.size = length;

}

public override double Area()

{

return Math.Pow(size, 2);

}

}

public class Circle : Shape

{

private double radius;

public Circle(double radius)

{

this.radius = radius;

}

public override double Area()

{

return Math.Round(Math.PI \* Math.Pow(radius, 2), 2);

}

}

public class Rectangle : Shape

{

private double size1, size2;

public Rectangle(double length1, double width2)

{

this.size1 = length1;

this.size2 = width2;

}

public override double Area()

{

return size1 \* size2;

}

}

public class Triangle : Shape

{

private double size1, size2;

public Triangle(double bas, double height)

{

this.size1 = bas;

this.size2 = height;

}

public override double Area()

{

return (size1 \* size2) / 2;

}

}

public class Program

{

public static void Main()

{

Shape[] shapes = { new Square(10), new Circle(10), new Rectangle(10, 8), new Triangle(3,2) };

foreach (var shape in shapes)

{

Console.WriteLine($"Area of {shape}: {shape.Area()}");

Console.WriteLine($"Area of {shape}: {Shape.GetArea(shape)}, again");

}

}

}

/\*

\* CSC180 Programming Assignment #7

\* key concepts: abstract class/method; inheritance; constructor; array of objects

\* Shape class: Shape.cs

\*/

using System;

public abstract class Shape

{

public abstract double Area();

public static double GetArea(Shape shape)

{

return shape.Area();

}

}

public class Square : Shape

{

private double size;

public Square(double length)

{

this.size = length;

}

public override double Area()

{

return Math.Pow(size, 2);

}

}

public class Circle : Shape

{

private double radius;

public Circle(double radius)

{

this.radius = radius;

}

public override double Area()

{

return Math.Round(Math.PI \* Math.Pow(radius, 2), 2);

}

}

public class Program

{

public static void Main()

{

Shape[] shapes = { new Square(10), new Circle(10) };

foreach (var shape in shapes)

{

Console.WriteLine($"Area of {shape}: {shape.Area()}");

Console.WriteLine($"Area of {shape}: {Shape.GetArea(shape)}, again");

}

}

}

**#2** This exercise is based on the existing code on the Microsoft web site below:  
 Create a simple C# console app (calculator) in Visual Studio:  
<https://docs.microsoft.com/en-us/visualstudio/get-started/csharp/tutorial-console?view=vs-2019>

Complete the following steps:  
 1) Create a new project called Calculator using the existing code: Program.cs and Calculator.cs.  
 Test run the program.

using System;

namespace Calculator

{

class Calculator

{

public static double DoOperation(double num1, double num2, string op)

{

double result = double.NaN; // Default value is "not-a-number" which we use if an operation, such as division, could result in an error.

// Use a switch statement to do the math.

switch (op)

{

case "a":

result = num1 + num2;

break;

case "s":

result = num1 - num2;

break;

case "m":

result = num1 \* num2;

break;

case "d":

// Ask the user to enter a non-zero divisor.

if (num2 != 0)

{

result = num1 / num2;

}

break;

// Return text for an incorrect option entry.

default:

break;

}

return result;

}

class Program

{

static void Main(string[] args)

{

bool endApp = false;

// Display title as the C# console calculator app.

Console.WriteLine("Console Calculator in C#\r");

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

while (!endApp)

{

// Declare variables and set to empty.

string numInput1 = "";

string numInput2 = "";

double result = 0;

// Ask the user to type the first number.

Console.Write("Type a number, and then press Enter: ");

numInput1 = Console.ReadLine();

double cleanNum1 = 0;

while (!double.TryParse(numInput1, out cleanNum1))

{

Console.Write("This is not valid input. Please enter an integer value: ");

numInput1 = Console.ReadLine();

}

// Ask the user to type the second number.

Console.Write("Type another number, and then press Enter: ");

numInput2 = Console.ReadLine();

double cleanNum2 = 0;

while (!double.TryParse(numInput2, out cleanNum2))

{

Console.Write("This is not valid input. Please enter an integer value: ");

numInput2 = Console.ReadLine();

}

// Ask the user to choose an operator.

Console.WriteLine("Choose an operator from the following list:");

Console.WriteLine("\ta - Add");

Console.WriteLine("\ts - Subtract");

Console.WriteLine("\tm - Multiply");

Console.WriteLine("\td - Divide");

Console.Write("Your option? ");

string op = Console.ReadLine();

try

{

result = Calculator.DoOperation(cleanNum1, cleanNum2, op);

if (double.IsNaN(result))

{

Console.WriteLine("This operation will result in a mathematical error.\n");

}

else Console.WriteLine("Your result: {0:0.##}\n", result);

}

catch (Exception e)

{

Console.WriteLine("Oh no! An exception occurred trying to do the math.\n - Details: " + e.Message);

}

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

// Wait for the user to respond before closing.

Console.Write("Press 'n' and Enter to close the app, or press any other key and Enter to continue: ");

if (Console.ReadLine() == "n") endApp = true;

Console.WriteLine("\n"); // Friendly linespacing.

}

return;

}

}

}

}  
 2) Summarize how the program handles various errors (e.g., divide-by-zero, unrecognized user input numbers, menu option (other than 'a', 's', 'm' and 'd'). Program catches exceptions by running a while loop until the number is not equal to 0. Also uses TryParse and NaN as a Boolean to determine if a string can be converted to an integer. If not, you will be asked to provide another number.  
 3) Add an extra exponent calculation function to the program.

using System;

namespace Calculator

{

class Calculator

{

public static double DoOperation(double num1, double num2, string op)

{

double result = double.NaN; // Default value is "not-a-number" which we use if an operation, such as division, could result in an error.

// Use a switch statement to do the math.

switch (op)

{

case "a":

result = num1 + num2;

break;

case "s":

result = num1 - num2;

break;

case "m":

result = num1 \* num2;

break;

case "d":

// Ask the user to enter a non-zero divisor.

if (num2 != 0)

{

result = num1 / num2;

}

break;

// Return text for an incorrect option entry.

case "e":

{

result = Math.Pow(num1, num2);

}

break;

}

return result;

}

class Program

{

static void Main(string[] args)

{

bool endApp = false;

// Display title as the C# console calculator app.

Console.WriteLine("Console Calculator in C#\r");

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

while (!endApp)

{

// Declare variables and set to empty.

string numInput1 = "";

string numInput2 = "";

double result = 0;

// Ask the user to type the first number.

Console.Write("Type a number, and then press Enter: ");

numInput1 = Console.ReadLine();

double cleanNum1 = 0;

while (!double.TryParse(numInput1, out cleanNum1))

{

Console.Write("This is not valid input. Please enter an integer value: ");

numInput1 = Console.ReadLine();

}

// Ask the user to type the second number.

Console.Write("Type another number, and then press Enter: ");

numInput2 = Console.ReadLine();

double cleanNum2 = 0;

while (!double.TryParse(numInput2, out cleanNum2))

{

Console.Write("This is not valid input. Please enter an integer value: ");

numInput2 = Console.ReadLine();

}

// Ask the user to choose an operator.

Console.WriteLine("Choose an operator from the following list:");

Console.WriteLine("\ta - Add");

Console.WriteLine("\ts - Subtract");

Console.WriteLine("\tm - Multiply");

Console.WriteLine("\td - Divide");

Console.WriteLine("\te - Exponent");

Console.Write("Your option? ");

string op = Console.ReadLine();

try

{

result = Calculator.DoOperation(cleanNum1, cleanNum2, op);

if (double.IsNaN(result))

{

Console.WriteLine("This operation will result in a mathematical error.\n");

}

else Console.WriteLine("Your result: {0:0.##}\n", result);

}

catch (Exception e)

{

Console.WriteLine("Oh no! An exception occurred trying to do the math.\n - Details: " + e.Message);

}

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

// Wait for the user to respond before closing.

Console.Write("Press 'n' and Enter to close the app, or press any other key and Enter to continue: ");

if (Console.ReadLine() == "n") endApp = true;

Console.WriteLine("\n"); // Friendly linespacing.

}

return;

}

}

}

}  
 4) The Calculator class handles the bulk of the calculation work, and the Program class handles  
 the user interface and error-capturing work. Would it be a good idea to move the user interface and error-capturing work from the Program class to the Calculator class? Please briefly discuss. The user interface is where all the errors are coming from and need to be caught separately. The calculator is performing the methods for the calculations. It would be a bad idea to keep the input of possible errors and the methods used for calculation in the same class for an error to affect everything at once.

The code is copied from the Microsoft site and pasted as following:

/\*

\* CSC180 Programming Assignment #7

\* Calculator class: Calculator.cs

\* Source: https://docs.microsoft.com/en-us/visualstudio/get-started/csharp/tutorial-console?view=vs-2019

\*/

using System;

namespace Calculator

{

class Calculator

{

public static double DoOperation(double num1, double num2, string op)

{

double result = double.NaN; // Default value is "not-a-number" which we use if an operation, such as division, could result in an error.

// Use a switch statement to do the math.

switch (op)

{

case "a":

result = num1 + num2;

break;

case "s":

result = num1 - num2;

break;

case "m":

result = num1 \* num2;

break;

case "d":

// Ask the user to enter a non-zero divisor.

if (num2 != 0)

{

result = num1 / num2;

}

break;

// Return text for an incorrect option entry.

default:

break;

}

return result;

}

}

}

/\*

\* CSC180 Programming Assignment #7

\* Tester: Program.cs

\* Source: https://docs.microsoft.com/en-us/visualstudio/get-started/csharp/tutorial-console?view=vs-2019

\*/

using System;

namespace Calculator

{

class Program

{

static void Main(string[] args)

{

bool endApp = false;

// Display title as the C# console calculator app.

Console.WriteLine("Console Calculator in C#\r");

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

while (!endApp)

{

// Declare variables and set to empty.

string numInput1 = "";

string numInput2 = "";

double result = 0;

// Ask the user to type the first number.

Console.Write("Type a number, and then press Enter: ");

numInput1 = Console.ReadLine();

double cleanNum1 = 0;

while (!double.TryParse(numInput1, out cleanNum1))

{

Console.Write("This is not valid input. Please enter an integer value: ");

numInput1 = Console.ReadLine();

}

// Ask the user to type the second number.

Console.Write("Type another number, and then press Enter: ");

numInput2 = Console.ReadLine();

double cleanNum2 = 0;

while (!double.TryParse(numInput2, out cleanNum2))

{

Console.Write("This is not valid input. Please enter an integer value: ");

numInput2 = Console.ReadLine();

}

// Ask the user to choose an operator.

Console.WriteLine("Choose an operator from the following list:");

Console.WriteLine("\ta - Add");

Console.WriteLine("\ts - Subtract");

Console.WriteLine("\tm - Multiply");

Console.WriteLine("\td - Divide");

Console.Write("Your option? ");

string op = Console.ReadLine();

try

{

result = Calculator.DoOperation(cleanNum1, cleanNum2, op);

if (double.IsNaN(result))

{

Console.WriteLine("This operation will result in a mathematical error.\n");

}

else Console.WriteLine("Your result: {0:0.##}\n", result);

}

catch (Exception e)

{

Console.WriteLine("Oh no! An exception occurred trying to do the math.\n - Details: " + e.Message);

}

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

// Wait for the user to respond before closing.

Console.Write("Press 'n' and Enter to close the app, or press any other key and Enter to continue: ");

if (Console.ReadLine() == "n") endApp = true;

Console.WriteLine("\n"); // Friendly linespacing.

}

return;

}

}

}