Application Development Orienting

Assignments week 4

Practical assignments.

**Version:** 0.2 **Last updated:** 23 August 2019

Contents

1 Quiz 4

1.1 Question 1 4

1.2 Question 2 4

1.3 Question 3 4

1.4 Question 4 5

1.5 Question 5 5

1.6 Question 6 5

2 Assignment – Calculations 6

2.1 Case Description 6

2.2 Tasks 6

3 Assignment – More algorithms 10

3.1 Case Description 10

4 Quiz answers 11

# Quiz

Difficulty: C:\Users\874156\Desktop\flatastic-icons-part-1-by-custom-icon-design\png\16x16\star-3_5.png. Estimated time: 30 minutes.

Answers to the quiz-questions can be found in the last section of this chapter.

## Question 1

What is the value of the variable sum after running this piece of code:

int a = 0 ;

int sum = 0;

while (a<5)

{

sum = sum + 2\*a;

a = a + 1;

}

## Question 2

What is the value of the variable sum after running this piece of code:

int a = 5 ;

int sum = 0;

while (a<23)

{

sum = sum + a;

a = a + 5;

}

## Question 3

What is the value of the variable sum after running this piece of code:

int a = 0 ;

int sum = 0;

int blip = 10;

while (a<7)

{

if (a % 2 == 0)

{ sum = sum + blip; }

else

{ sum = sum + a; }

a = a + 1;

}

## Question 4

What is the last text that is displayed by **MessageBox.Show(s)**?

int counter = 27 ;

String s;

while (counter<77)

{

s = "number is "+ counter.ToString();

MessageBox.Show(s);

counter = counter + 5;

}

## Question 5

What is the last text that is displayed by **MessageBox.Show(s)**?

int counter = 27 ;

String s;

while (counter<77)

{

counter = counter + 5;

s = "number is "+ counter.ToString();

MessageBox.Show(s);

}

## Question 6

What is the last text that is displayed by **MessageBox.Show(s)**?

int counter = 27 ;

String s;

do

{

counter = counter + 10;

s = "number is "+ counter.ToString();

MessageBox.Show(s);

}

while (counter<80)

# Assignment – Calculations

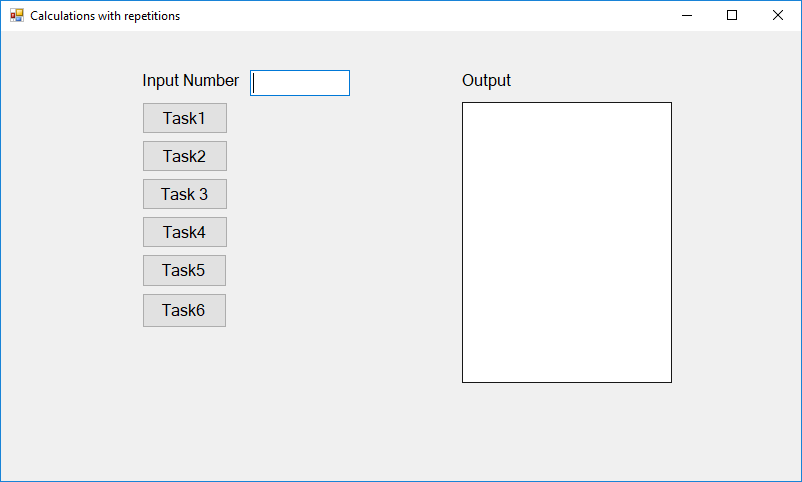
Difficulty: C:\Users\874156\Desktop\flatastic-icons-part-1-by-custom-icon-design\png\16x16\star-2_5.png . Estimated time: 120 minutes.

The assignment covers the following learning goals:

* Use loops to implement repetitive behavior in an application.

## Case Description

Create the skeleton of the following user interface as given below. Within this skeleton, you are expected to implement several calculation algorithms which use loops to repeat one task many times.



## Tasks

**Task 1:** Input a positive number “n” inside the input field. After pressing the “Task 1” button, the output box should display the result of the following calculation: 0 + 1 + 2 + 3 + 4 + … + n.

Tips:

You can use the function Convert.ToInt32(textBox1.Text) to convert String to Integer.

Input and output examples:

|  |  |
| --- | --- |
| Input | Output |
| 2 | The result is: 3 |
| 3 | The result is: 6 |
| 4 | The result is: 10 |

**Task 2:** This tackles the same problem as in the former one, but now, after every adding a number to the sum, you should display what was the sum so far and what was the last added number.

For instance, suppose you inputted the value 6 for n. Then in the output box you should have:

sum is 0   
sum is 1, last added is 1   
sum is 3, last added is 2   
sum is 6, last added is 3   
sum is 10, last added is 4   
sum is 15, last added is 5   
sum is 21, last added is 6

Implement this functionality for the “Task 2” button.

**Task 3:** Input a positive number “n” inside the input field. After pressing the “Task 3” button, the output box should display the result of the following calculation: 02 + 12 + 22 + ... + n2.

For instance, suppose the user inputted the value 4.   
The required answer is 30, since 02 + 12 + 22 +32 + 42 = 0 + 1 + 4 + 9 + 16 = 30.

Implement this functionality for pressing the “Task 3” button.

Input and output examples:

|  |  |
| --- | --- |
| Input | Output |
| 4 | 30 |
| 6 | 91 |
| 8 | 204 |

## 

**Task 4:** Input a positive number **“n”** inside the input field. After pressing the “Task 1” button, the output box should display n number of lines with n number of stars on each line. For example:

Input and output examples:

|  |  |
| --- | --- |
| Input | Output |
| n = 3 | \* \*\* \*\*\* |
| n = 5 | \* \*\* \*\*\* \*\*\*\* \*\*\*\*\* |

Implement the code for this functionality. It should be executed when the “Task 4” button is pressed.

**Task 5:** Input a positive number – “n” inside the input field. After pressing the “Task 5” button, the output box should display the following content:

Say the number of lines is **2\*n - 1**. and **“i”** is the number of the line which is currently being printed Your task is to display **i** number of stars on each line until **i** becomes **equal to n** and then **2\*n – i** stars on each line until the end of the sequence. The output should look like this:

Input and output examples:

|  |  |
| --- | --- |
| Input | Output |
| n = 3 | \* \*\* \*\*\*  \*\*  \* |
| n = 5 | \* \*\* \*\*\* \*\*\*\* \*\*\*\*\*  \*\*\*\*  \*\*\*  \*\*  \* |

Implement the “Task 5” functionality.

**Task 6:** Input a positive number “n” inside the input field. After pressing the “Task 6” button, the output box should display the following content:

Say the number of lines is **n** and **”i”** is the index of the line which is being printed now.  
Your task is to display **2\*n – 1** number of characters and **i** number of stars on each line. The stars **must** be separated by one **“-“**. We will define the length of the sequence of stars plus the dashes which are between them as **“p”**. On either side of each sequence of stars must be **((2\*n – 1) – p)/2** dashes. The results should look something like this:

Input and output examples:

|  |  |
| --- | --- |
| Input | Output |
| n = 3 | --\*-- -\*-\*- \*-\*-\* |
| n = 5 | ----\*---- ---\*-\*--- --\*-\*-\*--  -\*-\*-\*-\*- \*-\*-\*-\*-\* |

Implement the “Task 6” functionality.

# Assignment – More algorithms

Difficulty: Estimated time: 60 minutes.  
The assignment covers the following learning goals:

* Use loops to implement repetitive behavior in an application.

## Case Description

You a similar user interface for this question as in Assignment 1. Students are now expected to research some of the topics for the tasks **on their own**. This assignment has various popular problems which require loops to be solved.

**Task 1:** Write a program Windows Forms application to display the first **n** terms of the Lucas numbers. If you are not familiar with the Lucas numbers, research the web.

Input and output examples:

|  |  |
| --- | --- |
| Input | Output |
| 10 | 2 1 3 4 7 11 18 29 47 76 |

**Task 2:** Write a program Windows Forms application to display the Lucas numbers smaller than a given input number n.

Input and output examples:

|  |  |
| --- | --- |
| Input | Output |
| 200 | 2 1 3 4 7 11 18 29 47 76 123 199 |

**Task 3:** Write a program Windows Forms application to find the number and sum of all integers between an input number n and “n + 100” which are divisible by 9.

Input and output examples:

|  |  |
| --- | --- |
| Input | Output |
| 100 | 108  117  126  135  144  153  162  171  180  189  198  Sum is: 1683 |

# Quiz answers

|  |  |
| --- | --- |
| Question | Answer |
| 1 | 20 ( = 0 + 2 + 4 + 6 + 8 ) |
| 2 | 50 ( = 5 + 10 + 15 + 20 ) |
| 3 | 49 ( = 10 + 1 + 10 + 3 + 10 + 5 + 10 ) |
| 4 | number is 72 |
| 5 | number is 77 |
| 6 | number is 87 |