**Выполнил:** Штырев Глеб Сергеевич, группа ИСТб-23-2

# Лабораторное задание №1

## Задача на For

### Постановка задачи

**Задача №15.** Гражданин 1 марта открыл счет в банке, вложив A руб. Через каждый месяц размер вклада увеличивается на 2% от имеющейся суммы. Определить: а) за какой месяц величина ежемесячного увеличения вклада превысит B руб.; б) через сколько месяцев размер вклада превысит C руб.

### Словесное описание алгоритма

**Main:**

1. Объявляем переменные a, b и c и присваиваем им значения, введенные пользователем;

2. Выводим на экран значение, полученное при подаче переменных a, b, c на вход метода IncreaseMonthCount класса Logic;

3. Выводим на экран значение, полученное при подаче переменных a, b, c на вход метода DepositMonthCount класса Logic;

**IncreaseMonthCount:**

1. Объявляем строковую переменную aAnswer;

2. Объявляем целочисленные переменные floatA, floatB и floatC;

2. Если значения локальных аргументов a, b и c не является целыми числами:

2.1 Присваиваем переменной aAnswer значение «некорректный ввод»;

3. Иначе:

3.1 Присваиваем переменным floatA, floatB и floatC значения float.Parse(a), float.Parse(b) и float.Parse(c) соответственно;

3.2 Если значения floatA или floatB или floatC меньше 0:

3.1.1 Присваиваем переменной aAnswer значение «число за пределом ОДЗ»;

3.2 Иначе:

3.2.1 Объявляем вещественную переменную incr и присваиваем ей значение 0;

3.2.2 Объявляем целочисленную переменную aMonth и присваиваем ей значение -1;

3.2.3 Для incr от 0 до floatB:

3.2.3.1 Присваиваем переменной incr значение floatA / 50;

3.2.3.2 Увеличиваем значение переменной floatA на incr;

3.2.3.3 Если incr > floatB, а aMonth = -1, присваиваем переменной

aMonth значение переменной i;

3.2.4 Присваиваем переменной aAnswer значение переменной aMonth;

4. Возвращаем значение переменной aAnswer;

**DepositMonthCount:**

1. Объявляем строковую переменную bAnswer;

2. Объявляем целочисленные переменные floatA, floatB и floatC;

2. Если значения локальных аргументов a, b и c не является целыми числами:

2.1 Присваиваем переменной bAnswer значение «некорректный ввод»;

3. Иначе:

3.1 Присваиваем переменным floatA, floatB и floatC значения float.Parse(a), float.Parse(b) и float.Parse(c) соответственно;

3.2 Если значения floatA или floatB или floatC меньше 0:

3.1.1 Присваиваем переменной aAnswer значение «число за пределом ОДЗ»;

3.2 Иначе:

3.2.1 Объявляем вещественную переменную incr и присваиваем ей значение 0;

3.2.2 Объявляем целочисленную переменную bMonth и присваиваем ей значение -1;

3.2.3 Для floatA от floatA до floatC:

3.2.3.1 Присваиваем переменной incr значение floatA / 50;

3.2.3.2 Увеличиваем значение переменной floatA на incr;

3.2.3.3 Если floatA > floatC, а bMonth = -1, присваиваем переменной

bMonth значение переменной i;

3.2.4 Присваиваем переменной bAnswer значение переменной bMonth;

4. Возвращаем значение переменной bAnswer;

### Таблица спецификаций

**Main**

|  |  |  |
| --- | --- | --- |
| Имя | Назначение | Тип/Диапазон |
| a | Сумма, вложенная при открытии вклада | Текст |
| b | Интересующий размер увеличения вклада | Текст |
| c | Ожидаемая сумма на счету | Текст |

**IncreaseMonthCount**

|  |  |  |
| --- | --- | --- |
| Имя | Назначение | Тип/Диапазон |
| aAnswer | Выводящееся сообщение с ответом или ошибкой | Текст |
| floatA | Сумма, вложенная при открытии вклада | Вещественное положительное число |
| floatB | Интересующий размер увеличения вклада | Вещественное положительное число |
| floatC | Ожидаемая сумма на счету | Вещественное положительное число |
| incr | Размер увеличение вклада | Вещественное положительное число |
| aMonth | Месяц, когда размер увеличения вклада, больше введенного значение | Целое число |
| i | Параметр цикла | Целое число |

**DepositMonthCount**

|  |  |  |
| --- | --- | --- |
| Имя | Назначение | Тип/Диапазон |
| bAnswer | Выводящееся сообщение с ответом или ошибкой | Текст |
| floatA | Сумма, вложенная при открытии вклада | Вещественное положительное число |
| floatB | Интересующий размер увеличения вклада | Вещественное положительное число |
| floatC | Ожидаемая сумма на счету | Вещественное положительное число |
| incr | Размер увеличение вклада | Вещественное положительное число |
| bMonth | Месяц, когда размер вклада, больше введенного значение | Целое число |
| i | Параметр цикла | Целое число |

### Таблица тестов

| Номер теста | Что проверяем | Входные данные | Выходные данные |
| --- | --- | --- | --- |
| 1 | Вводим текст | a = "AAAAA"  b = "BBBBB"  c = "CCCCC" | aAnswer = "некорректный ввод"  bAnswer = "некорректный ввод" |
| 2 | Вводим числа за пределами ОДЗ | a = "-100"  b = "5"  c = "200" | aAnswer = "число за пределом ОДЗ"  bAnswer = "число за пределом ОДЗ" |
| 3 | Вводим корректный формат чисел | a = "100"  b = "5"  c = "200" | aAnswer = 48  bAnswer = 36 |

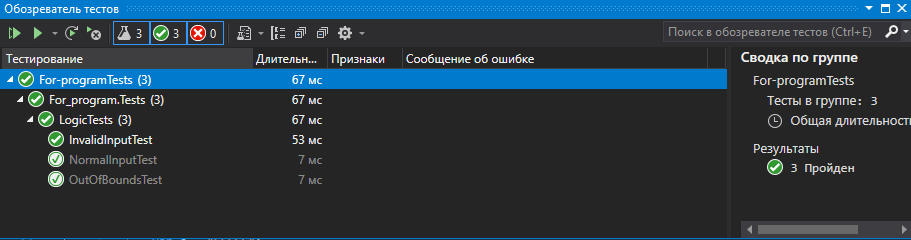
### Код программы

|  |
| --- |
| using System;  namespace For\_program  {  public class Logic  {  public static string IncreaseMonthCount(string a, string b, string c)  {  string aAnswer;  if (!float.TryParse(a, out float floatA) || !float.TryParse(b, out float floatB) || !float.TryParse(c, out float floatC))  {  aAnswer = "некорректный ввод";  }  else  {  floatA = float.Parse(a);  floatB = float.Parse(b);  floatC = float.Parse(c);  if (!(floatA >= 0 && floatB >= 0 && floatC >= 0))  {  aAnswer = "числа за пределами ОДЗ";  }  else  {  float incr = 0;  int aMonth = -1;  for (int i = 1; incr <= floatB; i++)  {  incr = floatA / 50;  floatA += incr;  if (incr > floatB && aMonth == -1)  {  aMonth = i;  }  }  aAnswer = aMonth.ToString();  }  }  return aAnswer;  }  public static string DepositMonthCount(string a, string b, string c)  {  string bAnswer;  if (!float.TryParse(a, out float floatA) || !float.TryParse(b, out float floatB) || !float.TryParse(c, out float floatC))  {  bAnswer = "некорректный ввод";  }  else  {  floatA = float.Parse(a);  floatB = float.Parse(b);  floatC = float.Parse(c);  if (!(floatA >= 0 && floatB >= 0 && floatC >= 0))  {  bAnswer = "числа за пределами ОДЗ";  }  else  {  float incr;  int bMonth = -1;  for (int i = 1; floatA <= floatC; i++)  {  incr = floatA / 50;  floatA += incr;  if (floatA > floatC && bMonth == -1)  {  bMonth = i;  }  }  bAnswer = bMonth.ToString();  }  }  return bAnswer;  }  }  class Program  {  static void Main(string[] args)  {  Console.WriteLine("Введите сумму, вложенную при открытии вклада");  var a = Console.ReadLine();  Console.WriteLine("Введите интересующий размер увеличения вклада");  var b = Console.ReadLine();  Console.WriteLine("Введите ожидаемую сумму на счету");  var c = Console.ReadLine();  Console.WriteLine("a) " + Logic.IncreaseMonthCount(a,b,c));  Console.WriteLine("b) " + Logic.DepositMonthCount(a,b,c));  Console.ReadKey();  }  }  } |

### Код тестов

|  |
| --- |
| using Microsoft.VisualStudio.TestTools.UnitTesting;  namespace For\_program.Tests  {  [TestClass()]  public class LogicTests  {  [TestMethod()]  public void InvalidInputTest()  {  var a = "AAAAA";  var b = "BBBBB";  var c = "CCCCC";  Assert.AreEqual(Logic.IncreaseMonthCount(a, b, c), "некорректный ввод");  Assert.AreEqual(Logic.DepositMonthCount(a, b, c), "некорректный ввод");  }  [TestMethod()]  public void OutOfBoundsTest()  {  var a = "-100";  var b = "5";  var c = "200";  Assert.AreEqual(Logic.IncreaseMonthCount(a, b, c), "числа за пределами ОДЗ");  Assert.AreEqual(Logic.DepositMonthCount(a, b, c), "числа за пределами ОДЗ");  }  [TestMethod()]  public void NormalInputTest()  {  var a = "100";  var b = "5";  var c = "200";  Assert.AreEqual(Logic.IncreaseMonthCount(a, b, c), "48");  Assert.AreEqual(Logic.DepositMonthCount(a, b, c), "36");  }  }  } |

### Результаты тестирования



### Ссылка на ГИТ

<https://github.com/Glebwen/TP-Lab1>