**Выполнил:** Комогорцева Юлия Владимировна ,ИСТБ-19-2

# Лабораторное задание №1

### Постановка задачи

**Задача №11(оранжевая)** Даны три квадратные матрицы A и B и С. Если разность между максимальным элементом и минимальным элементом матрицы B меньше разности между максимальным элементом A и минимальным элементом C то (B−C)T+2⋅A иначе A⋅3−CT⋅3+2−B

1. Находим maxB и minB
2. Находим maxA
3. Находим minC
4. ЕСЛИ (maxB – minB)<( maxA- minC) ТО
   1. F=(B−C)T+2⋅A
5. ИНАЧЕ
   1. F=A⋅3−CT⋅3+2−B

Для нахождений max min создадим метод

Так же для транспонирования, сложения ,умножения, вычитания.

**Словесное описание алгоритма**

**Метод Main**

1. Заводим переменную n под стоимость товара в копейках
2. ПОКА n не число или не в пределах 1≤n≤9999
   1. Вывод сообщения "Некорректный ввод. Введите стоимость не меньше 1 и не больше 9999"
3. Вывод результата, полученного при обращении к методу GetRezult(n)

**Метод GetRezult(n)** (Возвращает строку result , в которой записано: количество рублей + рубль/рубля/рублей + количество копеек + копейка/копеек/копейки)

1. Заводим переменные ruble и kopek под количество рублей и копеек. Переменной ruble присвоим значение целого от деления стоимости товара на 100. Переменной kopek присвоим значение остатка от деления стоимости товара на 100.
2. Заводим пустую строку result
3. ЕСЛИ рублей в стоимости товара нет
   1. То result равна числу копеек и результату обращения к методу GetVariantWordKopek(kopek)
4. ИНАЧЕ
   1. ЕСЛИ копеек в стоимости товара нет
      1. То result равна числу рублей и результату обращения к методу GetVariantWordRuble(ruble)
   2. ИНАЧЕ
      1. result равна числу рублей и результату обращения к методу GetVariantWordRuble(ruble) и числу копеек и результату обращения к методу GetVariantWordKopek(kopek)
5. Возврат result

**Метод GetVariantWordKopek** **(int a)** (Возвращает строку s, в которой записано копейка/копеек/копейки исходя из количества копеек)

1. Заводим пустую строку s
2. ЕСЛИ число копеек меньше 10 ТО
   1. ЕСЛИ число копеек
      1. равно 1
         1. Присваиваем строке s=”копейка”
      2. равно 2 или 3 или 4
         1. Присваиваем строке s=”копейки”
      3. равно 5 или 6 или 7 или 7 или 8 или 9 или 0
         1. Присваиваем строке s=”копеек”
3. ИНАЧЕ
   1. ЕСЛИ число копеек лежит в промежутке от 11 до 14 включительно ТО
      1. Присваиваем строке s=”копеек”
   2. ИНАЧЕ
      1. ЕСЛИ последняя цифра числа копеек
         1. равна 1
            1. Присваиваем строке s=”копейка”
         2. равна 2 или 3 или 4
            1. Присваиваем строке s=”копейки”
         3. равна 5 или 6 или 7 или 7 или 8 или 9 или 0
            1. Присваиваем строке s=”копеек”
4. Возврат s

**Метод GetVariantWordRuble(int a)** (Возвращает строку s, в которой записано рубль/рубля/рублей исходя из количества рублей)

1. Заводим пустую строку s
2. ЕСЛИ число рублей меньше 10 ТО
   1. ЕСЛИ число рублей
      1. равно 1
         1. Присваиваем строке s=” рубль”
      2. равно 2 или 3 или 4
         1. Присваиваем строке s=” рубля”
      3. равно 5 или 6 или 7 или 7 или 8 или 9 или 0
         1. Присваиваем строке s=” рублей”
3. ИНАЧЕ
   1. ЕСЛИ число рублей лежит в промежутке от 11 до 14 включительно ТО
      1. Присваиваем строке s=” рублей”
   2. ИНАЧЕ
      1. ЕСЛИ последняя цифра числа рублей
         1. равна 1
            1. Присваиваем строке s=” рубль”
         2. равна 2 или 3 или 4
            1. Присваиваем строке s=” рубля”
         3. равна 5 или 6 или 7 или 7 или 8 или 9 или 0
            1. Присваиваем строке s=” рублей”
4. Возврат s

### 

### Таблица спецификаций

|  |  |  |
| --- | --- | --- |
| Имя | Назначение | Тип/Диапазон |
| Входные величины | | |
| n | Стоимость товара в копейках | Целое число в диапазоне  1≤n≤9999 |
| Выходные величины | | |
| result | Строка с количеством рублей и копеек | Текст |

### 

### Таблица тестов

| Номер теста | Что проверяем | Входные данные | Выходные данные |
| --- | --- | --- | --- |
| **Проверяется автоматизированным тестированием** | | | |
| 1 | Нет рублей ,различные вариации употребления слова копейка | n=1 | result= “1 копейка” |
| 2 | n=2 | result= “2 копейки” |
| 3 | n=9 | result= “9 копеек” |
| 4 | n=12 | result= “12 копеек” |
| 5 | n=21 | result= “21 копейка” |
| 6 | n=100 | result= “1 рубль ровно” |
| 7 | Нет копеек ,различные вариации употребления слова рубль | n=200 | result= “2 рубля ровно” |
| 8 | n=500 | result= “5 рублей ровно” |
| 9 | n=1200 | result= “12 рублей ровно” |
| 10 | n=2100 | result= “21 рубль ровно” |
| 11 | Различные вариации употребления слов рубль и копейка | n=101 | result= “1 рубль 1 копейка” |
| 12 | n=1255 | result= “12 рублей 55 копеек” |
| **Не проверяется автоматизированным тестированием** | | | |
| 13 | Проверка корректности ввода-диапазон | n=0 | "Некорректный ввод. Введите стоимость не меньше 1 и не больше 9999" |
| 14 | n=10000 |
| 15 | Проверка корректности ввода-тип | n=abc |

### Все автотесты логичнее проверять в методе GetRezult , так как он автоматически обращается к остальным методам.

### 

### Код программы

|  |
| --- |
| using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  namespace ConsoleApp4  {  public class Logic  {  //метод возвращаемый подходящее слово к числу рублей  public static string GetVariantWordRuble(int a)  {  string s = null;  if (a < 9)  {  switch (a)  {  case 1:  s = " рубль ";  break;  case 2:  case 3:  case 4:  s = " рубля ";  break;  case 5:  case 6:  case 7:  case 8:  case 9:  case 0:  s = " рублей ";  break;  }  }  else if ((15 > a) & (a > 10))  {  s = " рублей ";  }  else  {  switch (a % 10)  {  case 1:  s = " рубль ";  break;  case 2:  case 3:  case 4:  s = " рубля ";  break;  case 5:  case 6:  case 7:  case 8:  case 9:  case 0:  s = " рублей ";  break;  }  }  return s;  }  //метод возвращаемый подходящее слово к числу копеек  public static string GetVariantWordKopek(int a)  {  string s = null;  if (a < 9)  {  switch (a)  {  case 1:  s = " копейка ";  break;  case 2:  case 3:  case 4:  s = " копейки ";  break;  case 5:  case 6:  case 7:  case 8:  case 9:  case 0:  s = " копеек ";  break;  }  }  else if ((15 > a) & (a > 10))  {  s = " копеек ";  }  else  {  switch (a % 10)  {  case 1:  s = " копейка ";  break;  case 2:  case 3:  case 4:  s = " копейки ";  break;  case 5:  case 6:  case 7:  case 8:  case 9:  case 0:  s = " копеек ";  break;  }  }  return s;  }  //метод возвращения результата  public static string GetRezult(int n)  {  int ruble = n / 100;  int kopek = n % 100;  string result;  //если рублей нет  if (rub le == 0)  {  //то с помощью метода GetVariantWordKopek(kopek) находим подходящее слово к числу копеек  result = kopek + GetVariantWordKopek(kopek);  }  else  //если копеек нет  if (kopek == 0)  {  //то с помощью метода GetVariantWordRuble(ruble) находимподходящее слово к числу рублей  result = ruble + GetVariantWordRuble(ruble) + "ровно";  }  //есть и копейки и рубли  else  {  //то с помощью метода GetVariantWordKopek(kopek) и GetVariantWordRuble(ruble) находим подходящее слова  result = ruble + GetVariantWordRuble(ruble) + kopek + GetVariantWordKopek(kopek);  }  return result;  }  }  class Program  {  static void Main(string[] args)  {  Console.WriteLine("Введите стоимость товара в копейках:");  int n;  // цикл на проверку корректности ввода,подходит если число и 1≤n≤9999  while (!(Int32.TryParse(Console.ReadLine(), out n)) || (n < 1) || (n > 9999))  {  Console.WriteLine("Некорректный ввод.Введите стоимость не меньше 1 и не больше 9999");  }  Console.WriteLine("Стоимость товара в рублях и копейках:" + Logic.GetRezult(n));  Console.ReadLine();  }  }  } |

**Код автоматизированного тестирования**

using Microsoft.VisualStudio.TestTools.UnitTesting;

using Лаба1;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Лаба1.Tests

{

[TestClass()]

public class LogicTests

{

[TestMethod()]

public void GetRezultTest1()

{

var n = 1;

var rezult = Logic.GetRezult(n);

Assert.AreEqual("1 копейка ", rezult);

}

[TestMethod()]

public void GetRezultTest2()

{

var n = 2;

var rezult = Logic.GetRezult(n);

Assert.AreEqual("2 копейки ", rezult);

}

[TestMethod()]

public void GetRezultTest3()

{

var n = 9;

var rezult = Logic.GetRezult(n);

Assert.AreEqual("9 копеек ", rezult);

}

[TestMethod()]

public void GetRezultTest4()

{

var n = 12;

var rezult = Logic.GetRezult(n);

Assert.AreEqual("12 копеек ", rezult);

}

[TestMethod()]

public void GetRezultTest5()

{

var n = 21;

var rezult = Logic.GetRezult(n);

Assert.AreEqual("21 копейка ", rezult);

}

[TestMethod()]

public void GetRezultTest6()

{

var n =100;

var rezult = Logic.GetRezult(n);

Assert.AreEqual("1 рубль ровно", rezult);

}

[TestMethod()]

public void GetRezultTest7()

{

var n = 200;

var rezult = Logic.GetRezult(n);

Assert.AreEqual("2 рубля ровно", rezult);

}

[TestMethod()]

public void GetRezultTest8()

{

var n =500;

var rezult = Logic.GetRezult(n);

Assert.AreEqual("5 рублей ровно", rezult);

}

[TestMethod()]

public void GetRezultTest9()

{

var n =1200;

var rezult = Logic.GetRezult(n);

Assert.AreEqual("12 рублей ровно", rezult);

}

[TestMethod()]

public void GetRezultTest10()

{

var n =2100;

var rezult = Logic.GetRezult(n);

Assert.AreEqual("21 рубль ровно", rezult);

}

[TestMethod()]

public void GetRezultTest11()

{

var n =101;

var rezult = Logic.GetRezult(n);

Assert.AreEqual("1 рубль 1 копейка ", rezult);

}

[TestMethod()]

public void GetRezultTest12()

{

var n = 1255;

var rezult = Logic.GetRezult(n);

Assert.AreEqual("12 рублей 55 копеек ", rezult);

}

}

}