Федеральное государственное бюджетное образовательное учреждение высшего образования «Пермский национальный исследовательский политехнический университет»

Лабораторная работа №9

По дисциплине

“Объектно-ориентированное программирование”

Тема

“Классы и объекты”

Вариант 13

Выполнил работу

студент группы РИС-19-1б

Мазитов Т.Э.

Проверила

доцент кафедры ИТАС

Викентьева О.Л.

Работу выполнил:

Пермь 2020

**Постановка задачи**

Задача 1:

* Реализовать (в отдельном файле) определение нового класса (закрытые атрибуты, свойства, конструкторы, инициализация и вывод атрибутов).
* Для демонстрации работы с объектами написать главную функцию, в которой создаются объекты класса и выводится информация, которая содержится в атрибутах.
* Написать функцию, реализующую указанное в варианте действие. Рассмотреть два варианта:
  + статическую функцию;
  + метод класса.
* В основной функции продемонстрировать работу функции.
* Используя статическую компоненту класса подсчитать количество созданных в программе объектов.

|  |  |  |  |
| --- | --- | --- | --- |
| № | Название класса | Атрибуты | Методы |
| 13 | Money | int rubles, int kopeks | Вычитание копеек (int) из объекта типа Money (учесть, что рублей и копеек не может быть меньше 0). Результат должен быть типа Money. |

Задача 2:

* Добавить к реализованному классу указанные в варианте перегруженные операции.
* Написать демонстрационную программу, в которой создаются объекты пользовательских классов и выполняются указанные операции.

|  |  |  |
| --- | --- | --- |
| 13 | Money | Унарные операции:  -- вычитание копейки из объекта типа Money (учесть, что рублей и копеек не может быть меньше 0).  ++ добавление копейки к объекту типа Money (учесть, что копеек не может быть больше 99).  Операции приведения типа:  int (явная) результатом является количество рублей (копейки отбрасываются);  bool (неявная) результатом является true, если денежная сумма не равна 0.  Бинарные операции:  - Money m, целое число (лево- и право- сторонние операции). Увеличиваются копейки, необходимо учесть, что копеек не может быть больше 99.  - Money m1, Money m2 вычитание денежных сумм, учесть, что результат не может быть меньше 0. |

Задача 3:

* Реализовать класс (в отдельном файле), полем которого является одномерный массив из элементов заданного в варианте типа.
* В классе реализовать
  + конструктор без параметров,
  + конструктор с параметрами, заполняющий элементы случайными значениями,
  + конструктор с параметрами, позволяющий заполнить массив элементами, заданными пользователем с клавиатуры,
  + индексатор (для доступа к элементам массива),
  + метод для просмотра элементов массива.
* Написать демонстрационную программу, позволяющую создать массив разными способами и распечатать элементы массива. Подсчитать количество созданных объектов.
* Выполнить указанное в варианте задание (если необходимо, перегрузить нужные для выполнения задачи операции или функции).

|  |  |  |
| --- | --- | --- |
| № | Название класса | Функция |
| 13 | Money | Среднее арифметическое |

**Проектирование**

|  |  |  |  |
| --- | --- | --- | --- |
| **№** | **Имя** | **Входные данные** | **Выходные данные** |
| **Money** | | | |
| **Свойства** | | | |
| 1 | Rubles – устанавливает и возвращает сумму рублей | Int – целое число рублей | Int – целое число рублей |
| 2 | Kopecks - устанавливает и возвращает сумму копеек | Int – целое число копеек | Int – целое число копеек |
| 3 | ObjectCount – счетчик количества объектов класса | Int – целое число | Int – целое число |
| **Конструкторы** | | | |
| 4 | Money () |  | Создает объект класса |
| 5 | Money (int, int) | int rubles – количество рублей,  int kopecks – количество копеек | Создает объект класса |
| 6 | Money(int) | int kopecks – количество копеек | Создает объект класса |
| **Перегруженные функции** | | | |
| 7 | Equals (object) – сравнивает текущий объект с переданным в аргументе | object obj – объект, с которым сравнивается текущим объектом | bool:  true – объекты равны,  false – объекты не равны |
| **Методы** | | | |
| 8 | ConsoleInput (string, int, int) – принимает и обрабатывает ввод пользователя | string msg – сообщению пользователю,  int beg – начало диапазона допустимых значений,  int end – конец диапазона | int elem – число, введенное пользователем |
| 9 | MoneyInput (int, int) – создает объект на основе ввода пользователя | int beg – начало диапазона допустимых значений,  int end – конец диапазона | Money – новый объект класса |
| 10 | StaticSubtractKopecks (Money, int) – вычитает копейки из объекта | Money m – объект класса,  Int kopecks – число копеек, которое нужно вычесть | Money – новый объект класса |
| 11 | ShowObjCount () – показывает в консоли сообщение о количестве созданных объектов |  | Вывод в консоль |
| 12 | Show () – вывод количества рублей и копеек в консоль |  | Вывод в консоль |
| 13 | Sum () – возвращает сумму рублей и копеек в копейках |  | int - сумма копеек и рублей |
| 14 | SubtractKopecks (int) - вычитает копейки | Int kopecks – число копеек, которое нужно вычесть | Money – новый объект класса |
| 15 | SetToZero () – устанавливает нуля в полях |  | Присваивает полям Rubles и Kopecks нули |
| **Перегруженный операторы** | | | |
| 16 | Money operator ++ (Money) – увеличивает количество копеек на 1 | Money m - объект класса | Money - объект класса |
| 17 | Money operator -- (Money) – уменьшает количество копеек на 1 | Money m - объект класса | Money - объект класса |
| 18 | operator int (Money) – преобразует объект к типу int | Money m - объект класса | Возвращает количество рублей объекта |
| 19 | operator bool (Money) | Money m - объект класса | true – сумма на равна 0,  false – сумма равна 0 |
| 20 | operator – (Money, Money) – вычитает соответствующие поля объектов | Money m1 - уменьшаемое, Money m2 - вычитаемое | Money m - объект класса, разность |
| 21 | operator + (Money, Money) – складывает соответствующие поля объектов | Money m1 - слагаемое, Money m2 - слагаемое | Money m - объект класса, сумма |
| 22 | operator – (Money, int) – вычитает копейки | Money m – уменьшаемое, int kopecks - вычитаемое | Money m - объект класса, разность |
| 23 | operator + (Money, int) – прибавляет копейки | Money m – слагаемое, int kopecks - слагаемое | Money m - объект класса, сумма |
| 24 | operator – (int, Money) – вычитает сумму в копейках | Money m – вычитаемое, int kopecks - уменьшаемое | int – целое число |
| 25 | operator + (int, Money) – прибавляет сумму в копейках | Money m – слагаемое, int kopecks - слагаемое | int – целое число |
| **MoneyArray** | | | |
| **Свойства** | | | |
| 1 | Size – размер массива | Int – целое число | Int – целое число |
| 2 | AverageRubels – среднее арифметическое рублей |  | Double – дробное число |
| 3 | AverageKopecks – среднее арифметическое копеек |  | Double – дробное число |
| **Конструкторы** | | | |
| 4 | MoneyArray () – конструктор по умолчанию |  | MoneyArray – объект класса |
| 5 | MoneyArray (int) – создает массив с заданным размером | Int size – размер массива | MoneyArray – объект класса |
| 6 | MoneyArray (int, int) – создает массива и заполняет его случайными значениями в пределах от 0 до end | Int size – размер массива,  Int end – конец диапазона значений | MoneyArray – объект класса |
| 7 | MoneyArray (Money []) – создает массив на основе переданного | Money [] arr – массив объектов Money | MoneyArray – объект класса |
| **Индексатор** | | | |
| 8 | Money this[int] – доступ к элементам по индексу | Int i – индекс элемента | Money – объект класса |
| **Методы** | | | |
| 9 | Show () – выводит элементы массива в консоль |  | Вывод в консоль |
| 10 | MoneyArrayInput () – создает объект на основе ввода пользователя |  | MoneyArray – объект класса |
| 11 | Average () – считает среднее арифметическое всех элементов |  | Money – объект класса |

**Тестирование**

**Счетчик созданных объектов**

|  |  |  |  |
| --- | --- | --- | --- |
| № | Ввод | ER | AR |
| 1 | MoneyArray(0) | 0 | 0 |
| 2 | MoneyArray(1) | 1 | 1 |
| 3 | MoneyArray(2) | 3 | 3 |
| 4 | MoneyArray(3) | 6 | 6 |
| 5 | MoneyArray(4) | 10 | 10 |

**Вычитание копеек**

|  |  |  |  |
| --- | --- | --- | --- |
| № | Ввод | ER | AR |
| 1 | Money (100,100)  100 | Money (100,0) | Money (100,0) |
| 2 | Money (100,100)  101 | Money (100, 100) | Money (99,99) |
| 3 | Money (1,0)  100 | Money (0, 0) | Money (0, 0) |
| 4 | Money (100,100)  -100 | Money (102, 0) | Money (102, 0) |
| 5 | Money (100,100)  0 | Money (101, 0) | Money (101, 0) |
| 6 | Money (1,0)  101 | Money (1,0) | Money (1,0) |

**Унарное сложение**

|  |  |  |  |
| --- | --- | --- | --- |
| № | Ввод | ER | AR |
| 1 | 0,0 | 0,1 | 0,1 |
| 2 | 0,99 | 1,0 | 1,0 |
| 3 | 0,100 | 1,1 | 1,1 |
| 4 | 1,0 | 1,1 | 1,1 |
| 5 | 1,99 | 2,0 | 2,0 |

**Унарное вычитание**

|  |  |  |  |
| --- | --- | --- | --- |
| № | Ввод | ER | AR |
| 1 | 0,0 | 0,0 | 0,0 |
| 2 | 0,1 | 0,0 | 0,0 |
| 3 | 0,100 | 0,99 | 0,99 |
| 4 | 1,0 | 0,99 | 0,99 |
| 5 | 1,1 | 1,0 | 1,0 |

**Приведение к типу int**

|  |  |  |  |
| --- | --- | --- | --- |
| № | Ввод | ER | AR |
| 1 | 0,0 | 0 | 0 |
| 2 | 0,1 | 0 | 0 |
| 3 | 0,100 | 1 | 1 |
| 4 | 1,0 | 1 | 1 |
| 5 | 1,1 | 1 | 1 |
| 6 | 1,100 | 2 | 2 |

**Приведение к типу bool**

|  |  |  |  |
| --- | --- | --- | --- |
| № | Ввод | ER | AR |
| 1 | 0,0 | false | False |
| 2 | 0,1 | true | true |
| 3 | 0,100 | true | true |
| 4 | 1,0 | true | true |
| 5 | 1,1 | true | true |
| 6 | 1,100 | true | true |

**Вычитание (Money - int)**

|  |  |  |  |
| --- | --- | --- | --- |
| № | ER | Ввод | AR |
| 1 | 0,0  0 | 0,0 | 0,0 |
| 2 | 0,1  1 | 0,0 | 0,0 |
| 3 | 1,0  1 | 0,99 | 0,99 |
| 4 | 1,0  100 | 0,0 | 0,0 |
| 5 | 1,1  1 | 1,0 | 1,0 |
| 6 | 1,0  101 | 1,0 | 1,0 |

**Вычитание (int - Money)**

|  |  |  |  |
| --- | --- | --- | --- |
| № | ER | Ввод | AR |
| 1 | 0  0,0 | 0 | 0 |
| 2 | 1  0,1 | 0 | 0 |
| 3 | 100  0,99 | 1 | 1 |
| 4 | 100  1,0 | 0 | 0 |
| 5 | 1  1,1 | -100 | -100 |
| 6 | 101  1,0 | 1 | 1 |

**Вычитание (Money - Money)**

|  |  |  |  |
| --- | --- | --- | --- |
| № | ER | Ввод | AR |
| 1 | 0,0  0,0 | 0,0 | 0,0 |
| 2 | 0,1  0,0 | 0,1 | 0,1 |
| 3 | 1,0  0,99 | 0,1 | 0,1 |
| 4 | 1,0  1,0 | 0,0 | 0,0 |
| 5 | 1,1  1,0 | 0,1 | 0,1 |
| 6 | 0,50  0,90 | 0,50 | 0,50 |

**Среднее арифметическое**

|  |  |  |  |
| --- | --- | --- | --- |
| № | ER | Ввод | AR |
| 1 | 1,0  2,0  3,0 | 2,0 | 2,0 |
| 2 | 3,1  3,1  3,1 | 3,1 | 3,1 |
| 3 | 0,1  0,2  0,3 | 0,2 | 0,2 |
| 4 | 1,1  2,2  3,3 | 2,2 | 2,2 |
| 5 | 1,50  2,50  3,50 | 2,50 | 2,50 |
| 6 | 0,0  0,0  0,0 | 0,0 | 0,0 |
| 7 | 0,3  0,3  0,4 | 0,3 | 0,3 |

**Среднее арифметическое копеек**

|  |  |  |  |
| --- | --- | --- | --- |
| № | ER | Ввод | AR |
| 1 | 1,10  2,30  3,40 | 26.67 | 26.67 |
| 2 | 3,0  3,0  3,0 | 0 | 0 |
| 3 | 0,1  0,2  0,3 | 2 | 2 |

**Среднее арифметическое рублей**

|  |  |  |  |
| --- | --- | --- | --- |
| № | ER | Ввод | AR |
| 1 | 10,1  30,2  40,3 | 26.67 | 26.67 |
| 2 | 0,0  0,0  0,0 | 0 | 0 |
| 3 | 1,1  2,2  3,3 | 2 | 2 |

**Листинг программы**

**Money.cs**

using System;

using System.Collections.Generic;

using System.Net.Mail;

using System.Runtime.CompilerServices;

using System.Text;

using System.Windows.Markup;

namespace LR9

{

public class Money

{

//-------------- ПОЛЯ КЛАССА ----------------//

private int rubles = 0;

private int kopecks = 0;

//------------- СВОЙСТВА КЛАССА -------------//

public int Rubles

{

get { return rubles; }

set

{

if (value < 0)

Console.WriteLine("\nОшибка: нельзя присвоить отрицательное значение");

else rubles = value;

}

}

public int Kopecks

{

get { return kopecks; }

set

{

if (value < 0)

Console.WriteLine("\nОшибка: нельзя присвоить отрицательное значение");

else

{

Rubles += value / 100;

kopecks = value % 100;

}

}

}

public static int ObjectCount { get; set; } = 0;

//----------- КОНСТРУКТОРЫ КЛАССА -----------//

//static Money() => ObjectCount = 0;

public Money()

{

Rubles = 0;

Kopecks = 0;

ObjectCount++;

}

public Money(int rubles, int kopecks): this()

{

Rubles = rubles;

Kopecks = kopecks;

}

public Money(int in\_kopecks): this() => Kopecks = in\_kopecks;

//---------- ПЕРЕГРУЖЕННЫЕ МЕТОДЫ КЛАССА ---------//

public override bool Equals(object obj)

{

return Rubles == ((Money)obj).Rubles && Kopecks == ((Money)obj).Kopecks;

}

//---------- СТАТИЧЕСКИЕ МЕТОДЫ КЛАССА ---------//

// Возвращает пользовательский ввод с консоли

public static int ConsoleInput(string msg, int beg, int end)

{

bool flag;

int elem = 0;

do

{

flag = false;

Console.WriteLine();

Console.Write(msg);

try

{

elem = Convert.ToInt32(Console.ReadLine());

if (elem < beg || elem > end) throw new IndexOutOfRangeException();

}

catch

{

flag = true;

Console.WriteLine("\nВы ввели некорректные данные. Повторите ввод.");

}

} while (flag);

return elem;

}

// Создает объект на основе пользовательского ввода

public static Money MoneyInput(int beg, int end)

{

int r = ConsoleInput($"Введите рубли (от {beg} до {end}): ", beg, end);

int k = ConsoleInput($"Введите копейки (от {beg} до {end}): ", beg, end);

return new Money(r,k);

}

// Вычитает копейки (статический)

public static Money StaticSubtractKopecks(Money m, int in\_kopecks)

{

int k = m.Sum() - in\_kopecks;

if (k < 0)

{

Console.WriteLine("\nОшибка: нельзя вычесть такое количество копеек.");

return new Money(m.Rubles, m.Kopecks);

}

else

return new Money(k);

}

//-------------- МЕТОДЫ КЛАССА --------------//

// Вывод количества рублей и копеек в консоль.

public void Show()

{

Console.WriteLine($"\nСумма: {Rubles} {(Rubles % 10 > 0 && Rubles % 10 < 5 ? Rubles % 10 == 1 ? "рубль" : "рубля" : "рублей")} и " +

$"{Kopecks} {(Kopecks % 10 > 0 && Kopecks % 10 < 5 ? Kopecks % 10 == 1 ? "копейка" : "копейки" : "копеек")}.");

}

// Возвращает сумму в копейках

public int Sum()

{

return Rubles \* 100 + Kopecks;

}

// Вычитает копейки

// Если вычитаемое количество больше суммы, то вычитание не происходит

public Money SubtractKopecks(int in\_kopecks)

{

int k = Sum() - in\_kopecks;

if (k < 0)

Console.WriteLine("\nОшибка: нельзя вычесть такое количество копеек.");

else

{

Rubles = 0;

Kopecks = k;

}

return this;

}

// Установливает нули в полях

public void SetToZero()

{

Kopecks = 0;

Rubles = 0;

}

// Выводит в консоль информацию о количестве созданных объектов

public static void ShowObjCount()

{

Console.WriteLine($"\nЗа время работы программы {(ObjectCount == 1 ? "был создан" : "было создано")} {ObjectCount} {(ObjectCount % 10 > 0 && ObjectCount % 10 < 5 ? ObjectCount % 10 == 1 ? "объект" : "объекта" : "объектов")} класса Money.");

}

//------------- УНАРНЫЕ ОПЕРАЦИИ ------------//

public static Money operator ++(Money m)

{

return new Money {Rubles = m.Rubles, Kopecks = m.Kopecks + 1 };

}

public static Money operator --(Money m)

{

return m.SubtractKopecks(1);

}

//----------- ПРЕОБРАЗОВАНИЯ ТИПОВ ----------//

public static explicit operator int(Money m)

{

return m.Rubles;

}

public static implicit operator bool(Money m)

{

return m.Rubles > 0 || m.Kopecks > 0;

}

//------------ БИНАРНЫЕ ОПЕРАЦИИ ------------//

//------------- Money +- Money --------------//

public static Money operator -(Money m1, Money m2)

{

return Money.StaticSubtractKopecks(m1, m2.Sum());

}

public static Money operator +(Money m1, Money m2)

{

return new Money { Kopecks = m1.Sum() + m2.Sum() };

}

//-------------- Money +- int ---------------//

public static Money operator -(Money m, int in\_kopecks)

{

return m.SubtractKopecks(in\_kopecks);

}

public static Money operator +(Money m, int in\_kopecks)

{

if (in\_kopecks < 0)

return m -= -in\_kopecks;

else

return new Money { Rubles = m.Rubles, Kopecks = m.Kopecks + in\_kopecks };

}

//-------------- int +- Money ---------------//

public static int operator +(int num, Money m)

{

return num + m.Sum();

}

public static int operator -(int num, Money m)

{

return num - m.Sum();

}

}

}

**MoneyArray.cs**

using System;

using System.Collections.Generic;

using System.Security.Cryptography;

using System.Text;

namespace LR9

{

public class MoneyArray

{

//-------------- ПОЛЯ КЛАССА ----------------//

private Money[] arr;

private int size;

//------------- СВОЙСТВА КЛАССА -------------//

public int Size

{

get { return size; }

set { size = value; }

}

/// <summary>

/// Возвращает среднее арифметическое рублей

/// </summary>

/// <returns>Число float</returns>

public double AverageRubles

{

get

{

double average = 0;

foreach (var m in arr)

average += m.Rubles;

average /= Size;

return Math.Round(average, 2);

}

}

/// <summary>

/// Возвращает среднее арифметическое копеек

/// </summary>

/// <returns>Число float</returns>

public double AverageKopecks

{

get

{

double average = 0;

foreach (var m in arr)

average += m.Kopecks;

average /= Size;

return Math.Round(average, 2);

}

}

//----------- КОНСТРУКТОРЫ КЛАССА -----------//

/// <summary>

/// Сводка:

/// Создает объект по умолчанию (пустой массив).

/// </summary>

/// <returns>Объект класса MoneyArray.</returns>

public MoneyArray()

{

Size = 0;

arr = new Money[Size];

}

/// <summary>

/// Сводка:

/// Создает объект и заполняет массив значениями по умолчанию.

/// </summary>

/// <param name="Size">Количество элементов в массиве.</param>

/// <returns>Объект класса MoneyArray.</returns>

public MoneyArray(int s)

{

Size = s;

arr = new Money[Size];

for (int i = 0; i < Size; i++)

arr[i] = new Money();

}

/// <summary>

/// Сводка:

/// Создает объект и заполняет его массив случайными числами от 0 до end.

/// </summary>

/// <param name="Size">Количество элементов в массиве.</param>

/// <param name="end">Верхний предел диапазона значений.</param>

/// <returns>Объект класса MoneyArray.</returns>

public MoneyArray(int s, int end)

{

Random rnd = new Random();

Size = s;

arr = new Money[Size];

for (int i = 0; i < Size; i++)

{

arr[i] = new Money(rnd.Next(0,end), rnd.Next(0, end));

}

}

/// <summary>

/// Создает объект на основе массива in\_arr.

/// </summary>

/// <param name="in\_arr">Массив объектов класса Money.</param>

/// <returns>Объект класса MoneyArray.</returns>

public MoneyArray(Money [] in\_arr)

{

Size = in\_arr.Length;

arr = new Money[Size];

in\_arr.CopyTo(arr,0);

}

//------------ ИНДЕКСАТОР КЛАССА ------------//

/// <summary>

/// Доступ к элементам массива по индексу.

/// </summary>

/// <param name="i">Номер элемента в массиве.</param>

/// <returns>Объект класса Money.</returns>

public Money this[int i]

{

get { return arr[i]; }

set { arr[i] = value; }

}

//------------- МЕТОДЫ КЛАССА ---------------//

/// <summary>

/// Вывод элементов массива.

/// </summary>

public void Show()

{

Console.WriteLine("\n\nВывод элементов массива:");

if (Size == 0) Console.WriteLine("В массиве нет элементов.");

for (int i = 0; i < Size; i++)

arr[i].Show();

}

/// <summary>

/// Создание объекта на основе данных пользователя.

/// </summary>

/// <returns>Новый объект класса MoneyArray</returns>

public static MoneyArray MoneyArrayInput()

{

Console.WriteLine("\n\nСоздание массива: ");

int Size = Money.ConsoleInput("Введите количество элементов в массиве (от 1 до 10): ", 1, 10);

Money[] array = new Money[Size];

for (int i = 0; i < Size; i++)

{

array[i] = Money.MoneyInput(0,100);

}

return new MoneyArray(array);

}

/// <summary>

/// Возвращает среднее арифметическое значение для элементов массива.

/// </summary>

/// <returns>Новый объект класса Money</returns>

public Money Average()

{

Money average = new Money();

int kop = 0;

for (int i = 0; i < Size; i++)

{

kop += arr[i].Kopecks + arr[i].Rubles \* 100;

}

if (Size != 0)

average.Kopecks = (kop /= Size);

return average;

}

}

}