*Федеральное государственное бюджетное образовательное учреждение высшего профессионального образования*

|  |  |  |  |
| --- | --- | --- | --- |
| Gerb-BMSTU_01 | **«Московский государственный технический университет имени Н.Э. Баумана»** | |  |
| Факультет: | | ИУ «Информатика и системы управления» |
| Кафедра: | | ИУ-6 «Компьютерные системы и сети» |

**Отчет**

**по домашнему заданию № 16**

**«Объекты. Композиция.»**

вариант №4

**Дисциплина: Основы программирования**

|  |  |  |
| --- | --- | --- |
| Студент группы ИУ6-51 | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | Милков В. В. |
|  | (Подпись, дата) | (Фамилия И.О.) |
| Преподаватель | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | Гукетлев Д. В. |
|  | (Подпись, дата) | (Фамилия И.О.) |

*Москва, 2016*

**Содержание**

Задание 3

UML-схема классов 4

[Код программы 5](#h.2et92p0)

[Скриншоты выполнения программы 9](#h.tyjcwt)

[Вывод 10](#h.3dy6vkm)

Задание

Разработать и реализовать диаграмму классов для описанных объектов предметной области, используя механизмы композиции. Проверить ее на тестовом примере, с демонстрацией всех возможностей разработанных классов на конкретных данных.

Объект – вещественное число, записанное с точкой. Параметры – его значение и запись. Методы: конструктор и процедура, определяющая количество цифр в дробной и целой части числа.

Объект – массив вещественных чисел переменной длины. Параметры: количество чисел и массив. Метод определения суммы введенных чисел и отношения количества цифр в целой и дробной частях.

**UML-схема классов**



**Код программы**

**program main;**

**uses sysutils, math;**

**type**

**RealClass = class**

**private**

**number: real;**

**integerPartString: string;**

**fractionPartString: string;**

**integerPart, fractionPart: integer;**

**public**

**constructor create(n: real);**

**function getFractionLength(): integer;**

**function getIntegerLength(): integer;**

**function ratio() : double;**

**end;**

**constructor RealClass.create(n: real);**

**var**

**numberString: string;**

**i: integer;**

**reached: boolean = false;**

**begin**

**numberString := FloatToStr(n);**

**for i := 1 to length(numberString) do begin**

**if numberString[i] = '.' then begin**

**reached := true;**

**continue;**

**end;**

**if not(reached) then begin**

**integerPartString := concat(integerPartString, numberString[i]);**

**end else begin**

**fractionPartString := concat(fractionPartString, numberString[i]);**

**end;**

**end;**

**integerPart := StrToInt(integerPartString);**

**fractionPart := StrToInt(fractionPartString);**

**number := n;**

**end;**

**function RealClass.getFractionLength() : integer;**

**begin**

**result := length(fractionPartString);**

**end;**

**function RealClass.getIntegerLength() : integer;**

**begin**

**result := length(integerPartString);**

**end;**

**type**

**RealClassList = class**

**public**

**list: array of RealClass;**

**constructor create();**

**procedure add(scObject: RealClass);**

**function sumRatio() : double;**

**function sum() : double;**

**end;**

**constructor RealClassList.create();**

**begin**

**end;**

**procedure RealClassList.add(scObject: RealClass);**

**var**

**l: integer = 0;**

**begin**

**l := length(list) + 1;**

**setlength(list, l);**

**list[l] := scObject;**

**end;**

**function RealClassList.sumRatio() : double;**

**var**

**i: integer;**

**res: double = 0;**

**begin**

**for i := 1 to length(list) do begin**

**res := res + (list[i].getIntegerLength() / list[i].getFractionLength());**

**end;**

**result := res;**

**end;**

**function RealClassList.sum() : double;**

**var**

**i: integer;**

**res: double = 0;**

**begin**

**for i := 1 to length(list) do begin**

**res := res + list[i].number;**

**end;**

**result := res;**

**end;**

**var**

**realNumberList: RealClassList;**

**begin**

**realNumberList := RealClassList.create();**

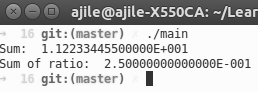
**realNumberList.add(RealClass.create(11.22334455));**

**writeln('Sum: ', realNumberList.sum());**

**writeln('Sum of ratio: ', realNumberList.sumRatio());**

**end.**

**Скриншоты выполнения программы**



**Вывод**

Агрегация (агрегирование по ссылке) — отношение «часть-целое» между двумя равноправными объектами, когда один объект (контейнер) имеет ссылку на другой объект. Оба объекта могут существовать независимо: если контейнер будет уничтожен, то его содержимое — нет.

Композиция (агрегирование по значению) — более строгий вариант агрегирования, когда включаемый объект может существовать только как часть контейнера. Если контейнер будет уничтожен, то и включённый объект тоже будет уничтожен.