Министерство образования и науки Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего образования

**«Пермский национальный исследовательский**

**политехнический университет»**

Кафедра «Информационные технологии и автоматизированные системы»

**О Т Ч Ё Т**

**по лабораторной работе №18.2**

Дисциплина: основы алгоритмизации и программирования

Тема: “Классы и объекты. Использование конструкторов.”

Вариант 19

Выполнил:

Студент группы РИС-20-1Б

Копейкин Дмитрий Евгеньевич

Проверила:

Доцент кафедры ИТАС

Полякова О.А.

Пермь, 2021

**Постановка задачи**

Определить пользовательский класс. Определить в класс следующие конструкторы: без параметров, с параметрами, копирования. Определить в классе деструктор.

2

**Анализ задачи**

**1.** Для решения задачи необходимо:

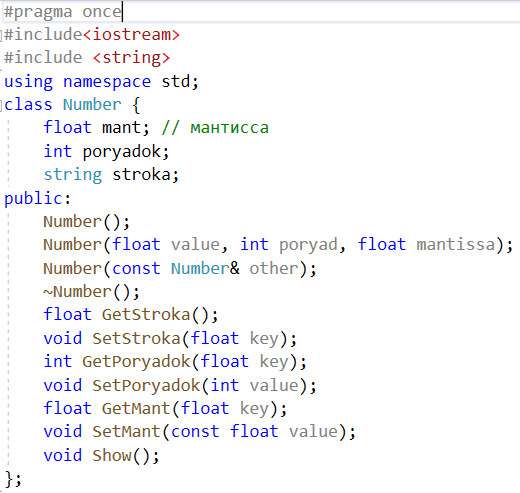
**1.1.** Организовать класс Number для хранения мантиссы в типе float, порядка типа int и самого числа в формате string.

**1.2.** Организовать необходимые методы для ввода данных в поля mant и poryadok, stroka: гетторы, сетторы, конструкторы, деструктор.

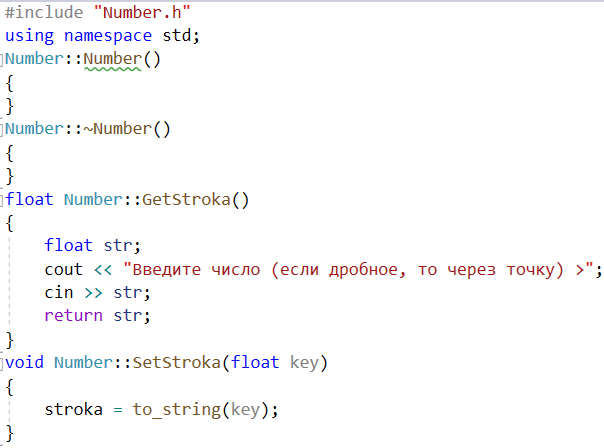
**1.3.** В функции main() использовать множественный if() для выбора пользователем, как он(-а) хочет ввести данные (через конструктор без параметров, с параметрами и конструктор копирования).

**2.** В ходе работы были использованы следующие типы данных:

**2.1.** Организовать класс Number для хранения мантиссы в типе float, порядка типа int и самого числа в формате string в заголовочном файле Number.h.



**2.2.** Организовать соответствующий файл Number.cpp, в котором будут описаны методы класса Number.



3

**2.3.** Переменные типа float: key, Por, mantissa, method, где key – введённое число, Por – переменная для хранения порядка числа, mantissa – переменная для выбора пользователем, как он хочет выполнить задачу.

float key = 0;

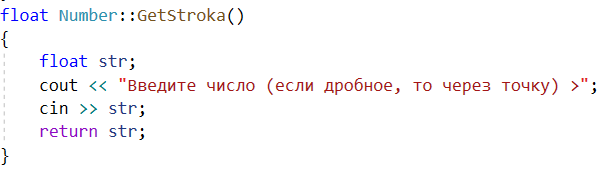
int Por = 0;

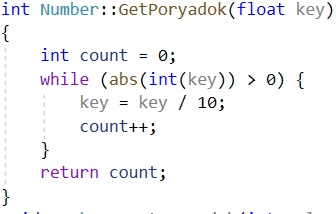
float mantissa = 0;

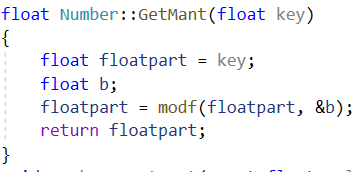
int method;

**3.** Для решения задачи данные были представлены в следующем виде:

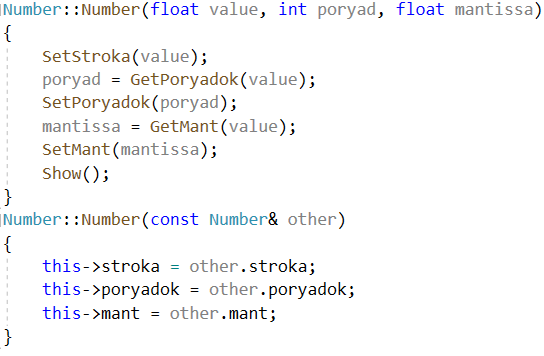
**3.1.** Данные вводятся через объекты класса Number через гетторы.







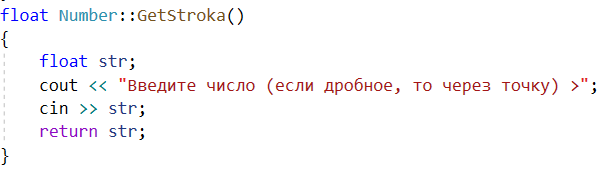
**3.2.** Данные также вводятся через объекты класса Number через конструктор с параметрами и конструктор копирования.



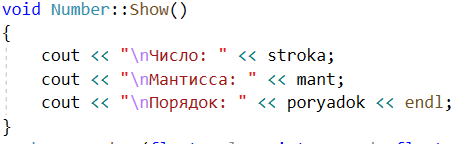
4

**4.** Для операций ввода и вывода использовались следующие операторы и функции:

**4.1.** Для ввода числа используется геттор (метод, описанный в классе для ввода), в котором используется функция cin.

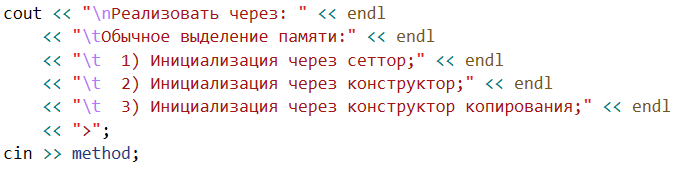


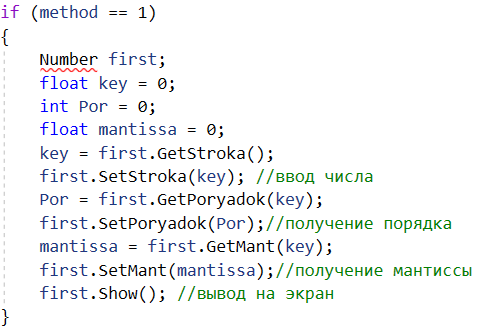
**4.2.** Для вывода данных на консоль используется метод Show().

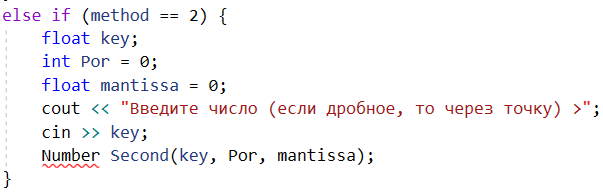


**5.** Поставленные задачи будут решены следующими действиями:

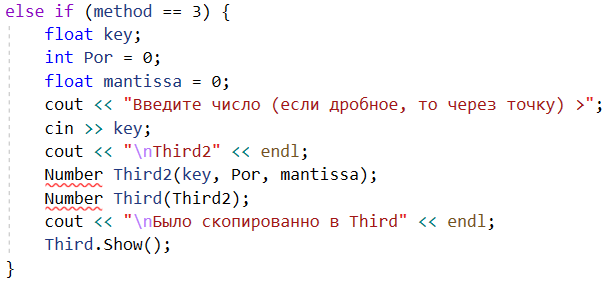
**5.1.** В функции main() через if() пользователь выбирает, какми методом выполнить поставленную задачу.



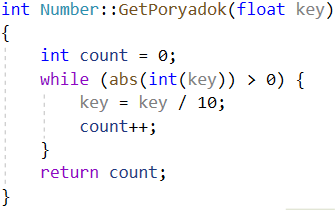




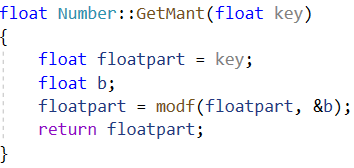
5



**5.2.** Метод GetPoryadok() класса Number cчитает порядок в цикле while(), в котором число целочисленно делится на 10 и с каждым заходом в цикл увеличивается счётчик count.

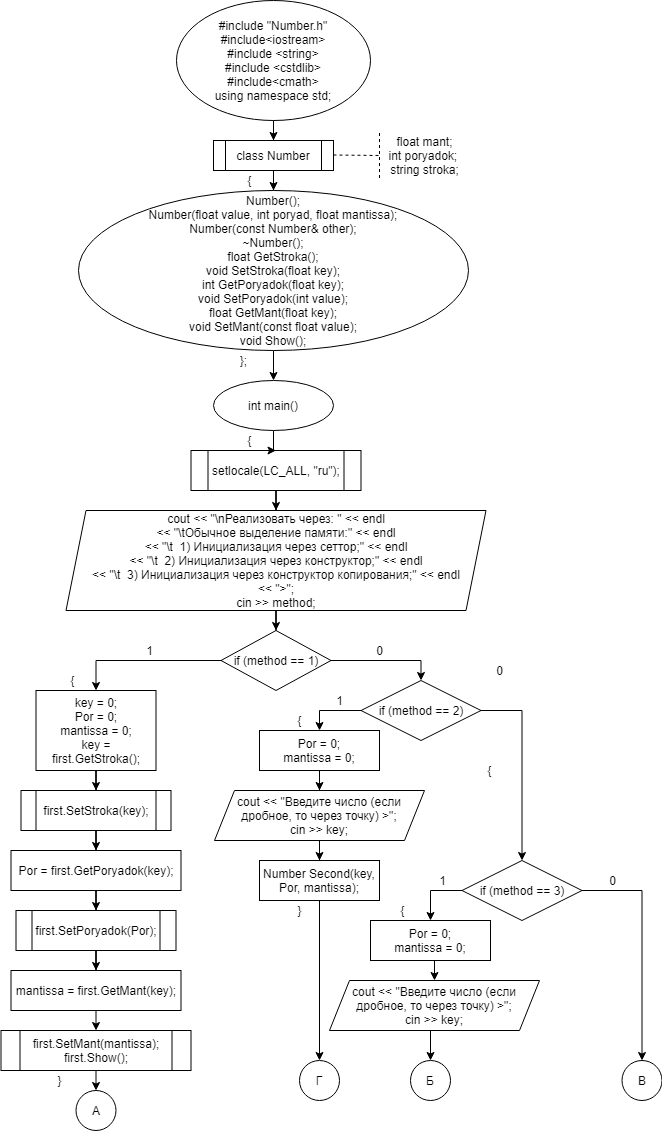


**5.3.** Метод GetMant() класса Number cчитает мантиссу через функцию modf, которая записывает в переменную floatpart дробную часть введённого числа, а в параметр b записывает целую часть числа.

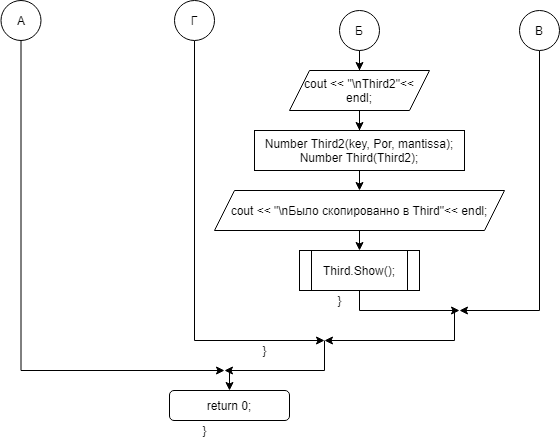


6

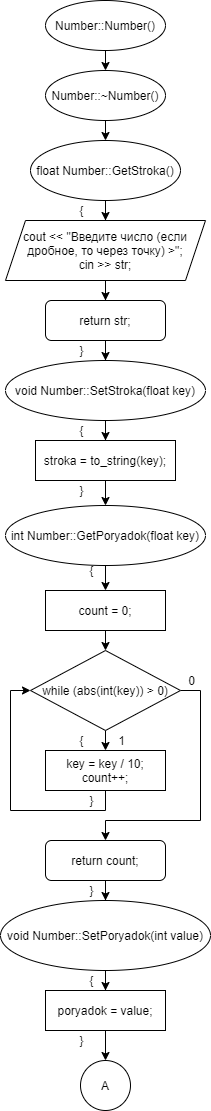
**Блок-схема**

****

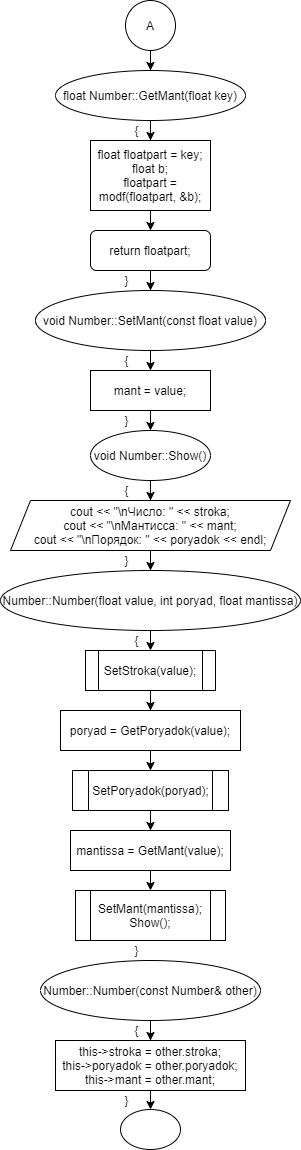
7



8



9



10

**Код**

#include "Number.h"

#include<iostream>

#include <string>

#include <cstdlib>

#include<cmath>

using namespace std;

int main()

{

setlocale(LC\_ALL, "ru");

int method;

cout << "\nРеализовать через: " << endl

<< "\tОбычное выделение памяти:" << endl

<< "\t 1) Инициализация через сеттор;" << endl

<< "\t 2) Инициализация через конструктор;" << endl

<< "\t 3) Инициализация через конструктор копирования;" << endl

<< ">";

cin >> method;

if (method == 1)

{

Number first;

float key = 0;

int Por = 0;

float mantissa = 0;

key = first.GetStroka();

first.SetStroka(key); //ввод числа

Por = first.GetPoryadok(key);

first.SetPoryadok(Por);//получение порядка

mantissa = first.GetMant(key);

first.SetMant(mantissa);//получение мантиссы

first.Show(); //вывод на экран

}

else if (method == 2) {

float key;

int Por = 0;

float mantissa = 0;

cout << "Введите число (если дробное, то через точку) >";

cin >> key;

Number Second(key, Por, mantissa);

}

else if (method == 3) {

float key;

int Por = 0;

float mantissa = 0;

cout << "Введите число (если дробное, то через точку) >";

cin >> key;

cout << "\nThird2"<< endl;

Number Third2(key, Por, mantissa);

Number Third(Third2);

cout << "\nБыло скопированно в Third"<< endl;

Third.Show();

}

}

11

#include "Number.h"

using namespace std;

Number::Number() {

}

Number::~Number() {

}

float Number::GetStroka()

{

float str;

cout << "Введите число (если дробное, то через точку) >";

cin >> str;

return str;

}

void Number::SetStroka(float key)

{

stroka = to\_string(key);

}

int Number::GetPoryadok(float key)

{

int count = 0;

while (abs(int(key)) > 0) {

key=key / 10;

count++;

}

return count;

}

void Number::SetPoryadok(int value)

{

poryadok = value;

}

float Number::GetMant(float key)

{

float floatpart = key;

float b;

floatpart = modf(floatpart, &b);

return floatpart;

}

void Number::SetMant(const float value)

{

mant = value;

}

void Number::Show()

{

cout << "\nЧисло: " << stroka;

cout << "\nМантисса: " << mant;

cout << "\nПорядок: " << poryadok << endl;

}

Number::Number(float value, int poryad, float mantissa)

{

SetStroka(value);

poryad=GetPoryadok(value);

SetPoryadok(poryad);

mantissa = GetMant(value);

SetMant(mantissa);

Show();

}

Number::Number(const Number& other)

{

this->stroka = other.stroka;

this->poryadok = other.poryadok;

this->mant = other.mant;

}

12

#pragma once

#include<iostream>

#include <string>

using namespace std;

class Number {

float mant; // мантисса

int poryadok;

string stroka;

public:

Number();

Number(float value, int poryad, float mantissa);

Number(const Number& other);

~Number();

float GetStroka();

void SetStroka(float key);

int GetPoryadok(float key);

void SetPoryadok(int value);

float GetMant(float key);

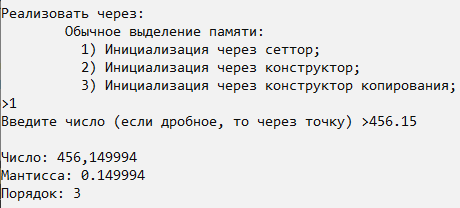
void SetMant(const float value);

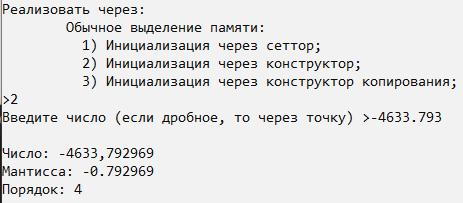
void Show();

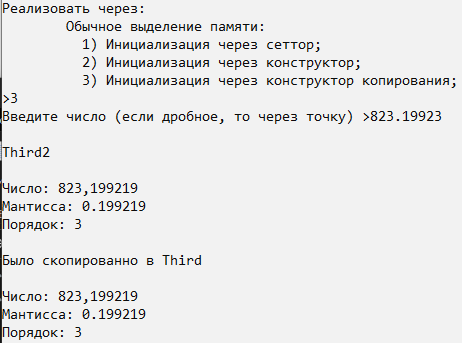
};

13

**Скриншоты**







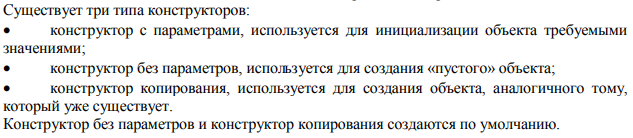
14

**Контрольные вопросы**



1. Конструктор необходим для инициализации объекта.



2. 

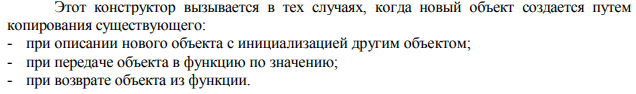


3. Деструктор – это особый вид метода, применяющийся для освобождения ресурсов, выделенных конструктором объекту. Описывать деструктор явным образом требуется в случае, когда объект содержит указатели на память, выделяемую динамически.



4. Конструктор с параметрами используется для инициализации объекта требуемыми значениями. Конструктор без параметров используется для создания «пустого» объекта. Конструктор копирования – это специальный вид конструктора, получающий в качестве единственного параметра указатель на объект этого же класса.

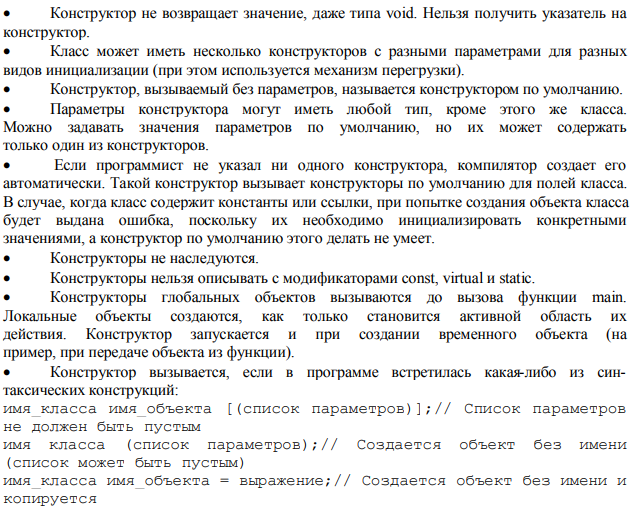


5. 

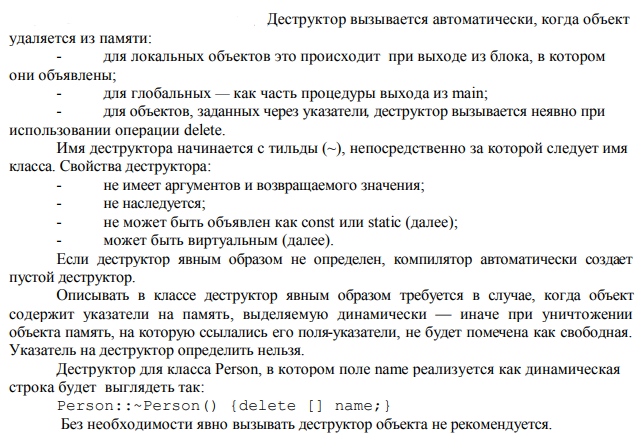


15

6.





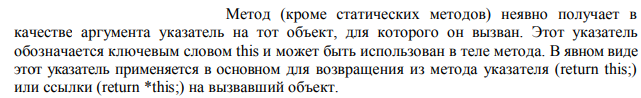
7. 

16

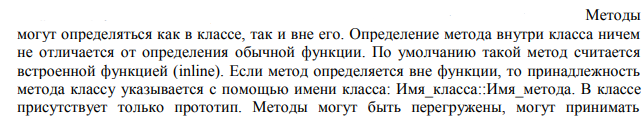


8.



9.



10.



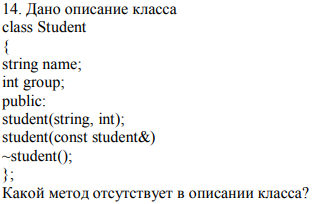
11. Конструктор не возвращает значения.



12. Конструкторы и деструкторы без параметров, если таковые не были прописаны явно.



13. Деструктор не возвращает значения.



14. Отсутствует конструктор без параметров в описании класса.

17



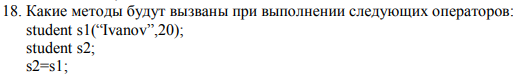
15. Будет вызван конструктор без параметров.



16. Будет вызван конструктор с параметрами.



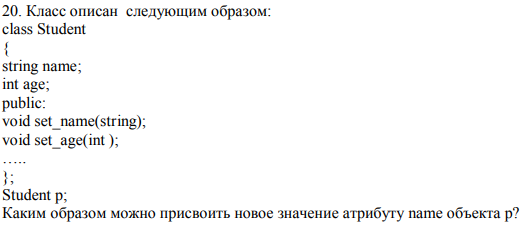
17. Будут вызваны конструктор с параметрами, а затем конструктор копирования.



18. Будут вызваны конструктор с параметрами, затем конструктор без параметров, а затем конструктор копирования.



19. Конструктор без параметров.



20. C помощью метода set\_name().

18