«Отчет по П. З. № 4 ПЕРЕГРУЗКА ОПЕРАТОРОВ. ШАБЛОНЫ ФУНКЦИЙ» Никита \*\*\* \*\*\*. Вариант 4.

Создать класс целых чисел. Определить оператор - -, как функцию-член и ++ как дружественную функцию.

В одномерном массиве, состоящем из n элементов, вычислить сумму элементов массива, расположенных между первым и последним отрицательными элементами.

1. Постановка задачи:

№ 1:

Функции класса IntNumber:

void SetNumber(); // Функция установки начального значения

Перегруженные операторы класса IntNumber:

void operator--() // Перегрузка оператора –

Дружественная функция класса IntNumber:

friend void operator++(IntNumber tmp) // Дружественная перегрузка ++

Иные функции:

void InputInt(int\* tmp); // Функция проверки ввода целого числа

Входные данные + ограничения:

А) Поле представителя класса tmp.number. Тип int. Целое число.

Исключения: буквы, строка, дробные числа

№ 2:

Функции:

void InputINT(int\*); // Прототип функции ввода целого значения

void InputDOU(double\*); // Прототип функции ввода double значения

void InputFLO(float\*); // Прототип функции ввода float значения

void InputINTS(int\*); // Прототип функции ввода целого значения

double Proccess(T\* array, int count, double\* sum) // Функция обработки элемен-тов

Входные данные + ограничения

А) Переменная size. Тип int. Размерность матрицы (количество строк).

Исключения: буква, буква с цифрой, строка, вещественное, меньше или равно 0, большие числа (зависит от наличия свободной оперативной памяти компьютера).

Б) Переменная array1[]. Тип int. Элементы матрицы.

Исключения: буква, буква с цифрой, строка, дробное число.

В) Переменная array2[]. Тип floar. Элементы матрицы.

Исключения: буква, буква с цифрой, строка.

Г) Переменная array3[]. Тип double. Элементы матрицы.

Исключения: буква, буква с цифрой, строка.

Выходные данные + ограничения:

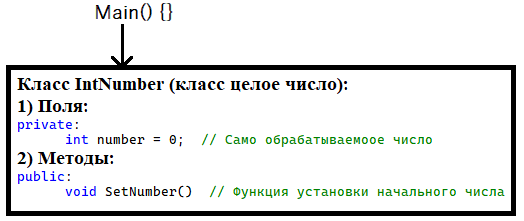
А) Перемення sum. Тип doubl. Сумма элементов от первого отрицательного до последнего отрицательного элементов.

Ограничения: буква, буква с цифрой, строка.

Среда разработки: Visual Studio 2022.

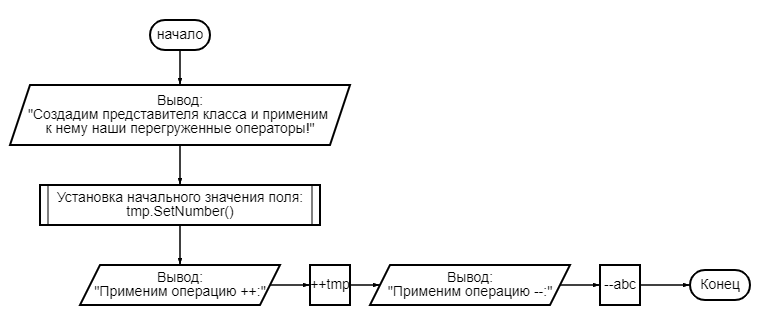
2. Разработка программы:

2.1 Описание классов, полей и методов классов.

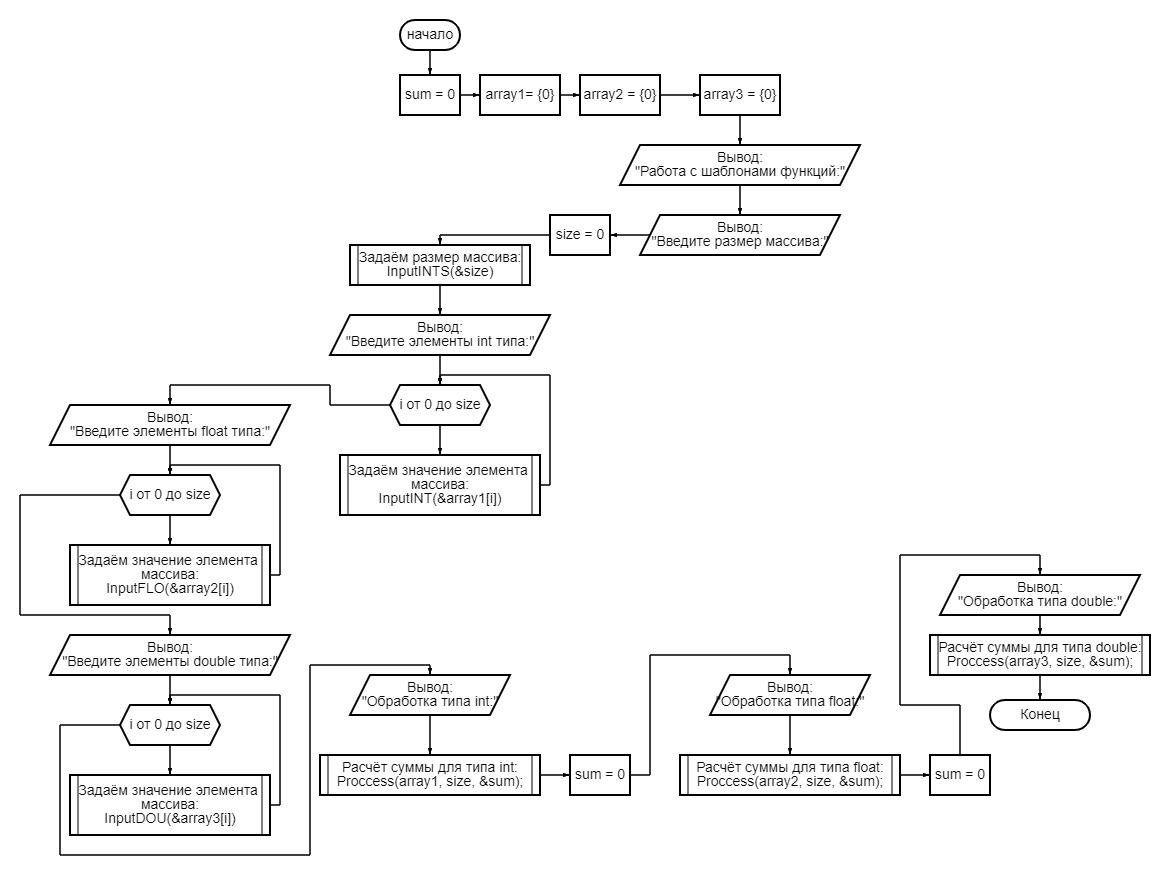


2.2Разработка схемы алгоритма.

№ 1:



№ 2:



2.3 Описание пользовательского интерфейса.

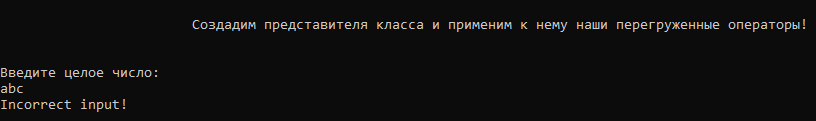
«Взаимодействие пользователя с программой осуществляется с помощью диалога».

Примеры диалогов:

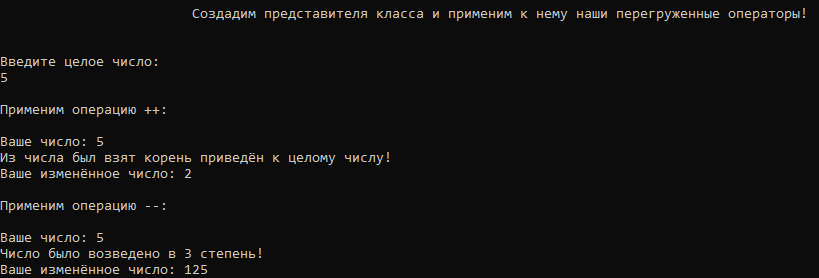
№ 1:

1. Ввод целого значения:

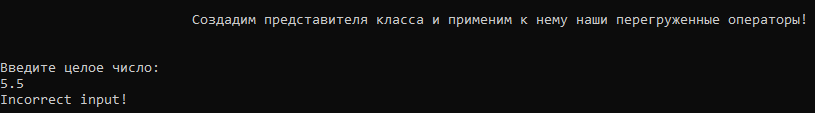
Вводимое значение – “abc”



Вводимое значение – 5



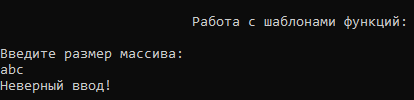
Вводимое значение – 5.5



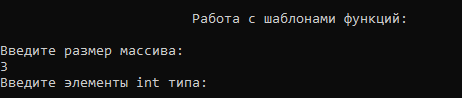
№ 2:

1. Ввод размера массива:

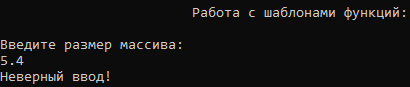
Вводимое значение – “abc”



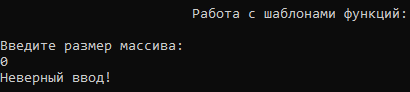
Вводимое значение – 3



Вводимое значение – 5.4



Вводимое значение – 0

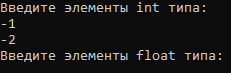


1. Ввод значения массива int;

Вводимое значение – “abc”



Вводимое значение – -2



Вводимое значение – 5.5



3) Ввод значения массива float;

Вводимое значение – “abc”



Вводимое значение – -2.5

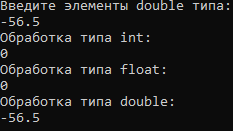


4) Ввод значения массива double;

Вводимое значение – “abc”



Вводимое значение – -56.5



3. Реализация и тестирование программы

3.1. Описание разработанной программы

№ 1:

Класс имеет одно приватное поле целого типа (number) – само целочисленное значение класса и один публичный метод (SetNumber) – установка этого начального значения, перегруженный оператор --, который берёт корень из числа и отбрасывает дробную часть, а также дружественный метод ++, который возводит число в 3 степень.

В функции main мы создаём представителя класса tmp и задаём значение его полю при помощи функции SetNumber, дальше создаём представителя класса abc и копируем значение поля из представителя tmp. После чего для каждого представителя вызываем перегруженный оператор.

№ 2:

Заранее определён шаблон для функции Process, которая выполняет основную логику программы.

После в main функции мы создаём три статических массива разных типов: int, float, double. И вызываем трижды метод Process, в которые в качестве аргументов суём массивы разных типов.

3.2. Тестирование программы

№ 1:

|  |  |  |
| --- | --- | --- |
| Входные данные | Результат программы | Смысл |
| tmp.SetNumber();  (abc) | Неверный ввод! | Проверка на строку |
| tmp.SetNumber();  (5.4) | Неверный ввод! | Проверка на дробное число |
| tmp.SetNumber();  (5) | 2  125 | Проверка на корректность работы программы |

№ 2:

|  |  |  |
| --- | --- | --- |
| Входные данные | Результат программы | Смысл |
| InputINTS(&size)  (abc) | Неверный ввод! | Проверка на строку |
| InputINTS(&size)  (2.4) | Неверный ввод! | Проверка на дробное число |
| InputINTS(&size)  (2) | 2 | Проверка на корректность работы программы |
| InputINTS(&size)  (-2) | Неверный ввод! | Проверка на отрицательную размерность |
| InputINT(&array1[i])  (abc) | Неверный ввод! | Проверка на строку |
| InputINT(&array1[i])  (5.5) | Неверный ввод! | Проверка на дробное число |
| InputINT(&array1[i])  (4) | 4 | Проверка на корректность работы программы |
| InputFLO(&array2[i])  (abc) | Неверный ввод! | Проверка на строку |
| InputFLO(&array2[i])  (5.5) | 5.5 | Проверка на корректность работы программы |
| InputDOU(&array3[i])  (abc) | Неверный ввод! | Проверка на строку |
| InputDOU(&array3[i])  (5.5) | 5.5 | Проверка на корректность работы программы |
| InputINTS(&size)  (2)  InputINT(&array1[i])  (0)  InputINT(&array1[i+1])  (0)  InputFLO(&array2[i])  (5.5)  InputFLO(&array2[i+1])  (5.5)  InputDOU(&array3[i])  (-6.4)  InputDOU(&array3[i+1])  (-4) | Обработка типа int:  0  Обработка типа float:  0  Обработка типа double:  -10.4 | Проверка на корректность работы программы |

Возможный диалог:

№ 1:

Ввод начального значения поля:

5

№ 2:

Ввод размера массива – ввод элементов целого типа – ввод элементов float типа – ввод элементов double типа:

3 \_ -1 4 -3 \_ 2.5 -9.4 4 \_ -64.1 -4 5

Код:

№ 1:

#include <iostream> // Стандартная библиотека ввода-вывода

#include <cmath> // Библиотека математических операций

using namespace std; // Стандартное пространство имён

void InputInt(int\* tmp) // Функция проверки ввода целого числа

{

while (!(cin >> \*tmp) || (cin.peek() != '\n'))

{

cin.clear();

while (cin.get() != '\n');

cout << "Incorrect input!" << endl;

}

}

class IntNumber // Класс целого числа

{

private:

int number = 0; // Само обрабатываемоое число

public:

void SetNumber() // Функция установки начального числа

{

cout << "Введите целое число: " << endl;

InputInt(&number); // Вызов функции проверки ввода

}

void operator--() // Перегрузка оператора --

{

cout << "Ваше число: " << number << endl;

number = sqrt(number);

cout << "Из числа был взят корень приведён к целому числу!" << endl;

cout << "Ваше изменённое число: " << number << endl;

}

friend void operator++(IntNumber tmp) // Дружественная перегрузка ++

{

cout << "Ваше число: " << tmp.number << endl;

tmp.number = pow(tmp.number, 3);

cout << "Число было возведено в 3 степень!" << endl;

cout << "Ваше изменённое число: " << tmp.number << endl;

}

};

void main(void)

{

setlocale(LC\_ALL, "Russian"); // Установка русского языка в консоли

cout << "\n\t\t\tСоздадим представителя класса и применим к нему наши перегруженные операторы!\n\n" << endl;

IntNumber tmp; // Создание представителя класса

tmp.SetNumber(); // Установка начального значения полю представителя класса

IntNumber abc = tmp; // Создание нового представителя для демонстрации работы второй перегрузки оператора

cout << "\nПрименим операцию ++:" << endl << endl;

--tmp;

cout << "\nПрименим операцию --:\n" << endl;

++abc;

}

№ 2:

#define SIZE 15 // Размер статических массивов

#include <iostream> // Стандартная библиотека ввода-вывода

using namespace std; // Использование стандартного пространства имён

void InputINT(int\*); // Прототип функции ввода целого значения

void InputDOU(double\*); // Прототип функции ввода double значения

void InputFLO(float\*); // Прототип функции ввода float значения

void InputINTS(int\*); // Прототип функции ввода целого значения

template <typename T> // Шаблон

double Proccess(T\* array, int count, double\* sum, bool\* key) // Функция обработки элемен-тов

{

int I = 0; // Индекс первого отрицательного числа

int J = 0; // Индекс последнего отрицательного числа

for (int i = 0; i < count; i++) // Поиск индекса

{

if ((array[i]) < 0)

{

I = i;

break;

}

}

for (int i = 0; i < count; i++) // Поиск индекса

{

if ((array[i]) < 0)

{

J = i;

}

}

if (I == J)

{

cout << "Недостаточно отрицательных элементов!" << endl;

\*key = false;

return 0;

}

if (array[I] < 0) // Поиск суммы элементов между первым и последним отри-цательными числами

{

for (I; I <= J; ++I)

{

\*sum += array[I];

}

return \*sum;

}

}

void main(void)

{

setlocale(LC\_ALL, "Russian"); // Установка русского языка в консоли

double sum = 0; // Сумма элементов

bool key = true;

int array1[SIZE] = { 0 }; // Массив целых чисел

float array2[SIZE] = { 0 }; // Массив float чисел

double array3[SIZE] = { 0 }; // Массив double чисел

cout << "\n\t\t\tРабота с шаблонами функций:\n" << endl;

cout << "Введите размер массива:" << endl;

int size = 0; // Размер массива вводимого с клавиатуры

InputINTS(&size); // Установка размера массива

cout << "Введите элементы int типа:" << endl;

for (int i = 0; i < size; ++i) // Ввод элементов массива

{

InputINT(&array1[i]); // Вызов функции проверки введённого значения

}

cout << "Введите элементы float типа:" << endl;

for (int i = 0; i < size; ++i) // Ввод элементов массива

{

InputFLO(&array2[i]); // Вызов функции проверки введённого значения

}

cout << "Введите элементы double типа:" << endl;

for (int i = 0; i < size; ++i) // Ввод элементов массива

{

InputDOU(&array3[i]); // Вызов функции проверки введённого значения

}

double tmpl = 0;

cout << "Обработка типа int:" << endl;

tmpl = Proccess(array1, size, &sum, &key);

if (key == true)

{

cout << tmpl << endl;

}

sum = 0;

key = true;

cout << "Обработка типа float:" << endl;

tmpl = Proccess(array2, size, &sum, &key);

if (key == true)

{

cout << tmpl << endl;

}

sum = 0;

key = true;

cout << "Обработка типа double:" << endl;

tmpl = Proccess(array3, size, &sum, &key);

if (key == true)

{

cout << tmpl << endl;

}

}

void InputINTS(int\* tmp) // Функция проверки размера массива

{

while (!(cin >> \*tmp) || (cin.peek() != '\n') || (\*tmp <= 0))

{

cin.clear();

while (cin.get() != '\n');

cout << "Неверный ввод!" << endl;

}

}

void InputINT(int\* tmp) // Функция проверки элемента типа int

{

while (!(cin >> \*tmp) || (cin.peek() != '\n'))

{

cin.clear();

while (cin.get() != '\n');

cout << "Неверный ввод!" << endl;

}

}

void InputDOU(double\* tmp) // Функция проверки элемента типа double

{

while (!(cin >> \*tmp) || (cin.peek() != '\n'))

{

cin.clear();

while (cin.get() != '\n');

cout << "Неверный ввод!" << endl;

}

}

void InputFLO(float\* tmp) // Функция проверки элемента типа float

{

while (!(cin >> \*tmp) || (cin.peek() != '\n'))

{

cin.clear();

while (cin.get() != '\n');

cout << "Неверный ввод!" << endl;

}

}