КАФЕДРА № 43

ОТЧЕТ
ЗАЩИЩЕН С ОЦЕНКОЙ
ПРЕПОДАВАТЕЛЬ

должность, уч. степень, звание	подпись, дата	инициалы, фамилия

ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ №1

Классы, определение методов класса, права доступа

по курсу: Объектно-ориентированное программирование

РАБОТУ ВЫПОЛНИЛ

СТУДЕНТ ГР. №

подпись, дата	инициалы, фамилия

Санкт-Петербург 2022

Условие

6 вариант (16 mod 10)

6. Поле *left* - вещественное число, левая граница диапазона. Поле *right* - вещественное число, правая граница диапазона. Пара этих числе представляет полуоткрытый интервал [left,right). Реализовать класс, в котором предусмотреть метод *rangecheck()* - проверку заданного числа на принадлежность диапазону.

Листинг программы

main.cpp

```
#include <iostream>
using namespace std;

#include "libs/lib.h"
#include <cmath>
#include <time.h>

// проверка ввода
#include "libs/simple_char.h"
#include "libs/input_validation.h"

#include "range.h"

int main() {


  Range range;

  // вводи диапазон
  double left = read_value("Левая граница: ", true, true, false);
  double right = read_value("Правая граница: ", true, true, false);

  // переворачиваем диапазон если пользователь его не правльно ввёл
  if (left > right) {
    double buf = left;
    left = right;
    right = buf;
  }

  // устанавливаем диапазон
  range.rangeset(left, right);
```

```
  // вывод введённых чисел
  draw_line(20);
  cout << "Введёный диапазон" << endl;
  range.rangedraw();
  draw_line(20);

  bool range_status = range.rangecheck(
    read_value("Число для проверки диапазона: ", true, true, false)
  );

  // вывод результата
  if (range_status)
    cout << "Число входит в диапазон." << endl;

  else
    cout << "Число не входит в диапазон." << endl;

        return 0;
}
```

range.h

```
#include <iostream>
using namespace std;

class Range{
  public:
    void rangedraw();
    void rangeset(double set_left, double set_right);
    bool rangecheck(double num);

  private:
    double left = 0;
    double right = 0;

};

void Range::rangedraw() {
  cout << "Левая граница: " << left << endl;
  cout << "Правая граница: " << right << endl;
}

void Range::rangeset(double set_left, double set_right) {
  left = set_left;
  right = set_right;
```

```
}

bool Range::rangecheck(double num) {
  return (left <= num) && (num < right);
}
```

input_validation.h

```
#include <iostream>

#define SYMB_LEN_DOUBLE 11

using namespace std;

double read_value(
        const char *promt = "",
        bool check_dot = true,
        bool check_minus = true,
        bool check_space = true
    ) {

    // строка введённая пользователем
    char *char_str;
    int len;

    // очищенная строка (без пробелов)
    char *clear_char_str;
    int len_clear;

    // счётсчик для динамического массива
    int capacity;

    // словарь
    char symb[SYMB_LEN_DOUBLE] = "1234567890";

    int i, j;

    // флаг состояния
    bool ok = false;

    // флаги ошибок
    bool error_dot = false;
    bool error_minus = false;

    // флаг на существование точки
```

```
bool dot = false;

bool minus = false;

while (true) {
    // возвращаем все флаги в исходное состояние
    ok = false;
    dot = false;
    minus = false;

    error_dot = false;
    error_minus = false;

    // читаем строку
    cout << promt;
    char_str = get_string(&len);

    capacity = 1;
    clear_char_str = (char*) malloc(sizeof(char));

    // очистка строки от пробелов
    if (check_space){
        len_clear = 0;
        for (i = 0; i < len; i++) {
            if (char_str[i] != ' ') {
                // замента запятой на точку
                if (char_str[i] == ',')
                clear_char_str[len_clear++] = '.';
                else
                clear_char_str[len_clear++] = char_str[i];

                capacity *= 2;
                clear_char_str = (char*) realloc(clear_char_str, capacity * sizeof(char));
            }
        }
    } else {
        len_clear = len;
        clear_char_str = char_str;
    }


    // проверка на знак минуса
    if (char_str[0] == '-' && check_minus) {
        minus = true;
    }
```

```
// проходимся по каждому символу строки
for (i = ((minus)? 1: 0); i < len_clear; i++) {
   switch (clear_char_str[i]) {
      // проверка на точку
      case('.'):
         if (check_dot)
            if (dot) {
               ok = false;
               error_dot = true;
            } else {
               ok = true;
               dot = true;
            }
         else {
            ok = false;
            error_dot = true;
         }

         break;

      // проверка на знак минуса
      case('-'):
         ok = false;
         error_minus = true;
         break;

      // проверка на остальные символы
      default:
         ok = false;
         // проходимся по каждому символу словаря
         for (j = 0; j < SYMB_LEN_DOUBLE; j++) {

            // стравниваем символ со словарём
            if (clear_char_str[i] == symb[j]){
               // если нашли символ в словаре,
               // то останавливаем цикл со словарём
               ok = true;
               break;
            }
         }

         break;
   }
```

```
            // если мы не нашли символ, пишем ошибку
            if (!ok) {
                cout << " [ Ошибка ввода ]: ";

                // ошибки связанные с точками
                if (error_dot && !check_dot) {
                    cout << "Число не должно содержать точки.";
                } else if (error_dot) {
                    cout << "Слишком много точек.";
                }

                // ошибки связанные со знаком минус
                if (error_minus && !check_minus) {
                    cout << "Число не может быть отрицательным.";
                } else if (error_minus) {
                    cout << "Число не может содержать несколько знаков минус.";
                }

                // остальные ошибки
                if (!error_dot && !error_minus) {
                    cout << "Число не может содержать в себе посторонних символов.";
                }

                cout << endl;
                break;
            }
        }

        // если небыло ошибок, то останавливаем бесконечный цикл
        if (ok)
            break;

    }

    // переводим и возвращаем значение
    return atof(clear_char_str);
}
```

Скриншоты

```
Левая граница: ыав
 [ Ошибка ввода ]: Число не может содержать в себе посторонних символов.
Левая граница: 12ds
 [ Ошибка ввода ]: Число не может содержать в себе посторонних символов.
Левая граница: 123
Правая граница: -13
--------------------
Введёный диапазон
Левая граница: -13
Правая граница: 123
--------------------
Число для проверки диапазона: 2323
Число не входит в диапазон.
Program ended with exit code: 0
```

```
Левая граница: -111
Правая граница: 11111
--------------------
Введёный диапазон
Левая граница: -111
Правая граница: 11111
--------------------
Число для проверки диапазона: 11
Число входит в диапазон.
Program ended with exit code: 0
```

Вывод

Я смог изучить принцип создания класса, ограничения прав доступа к полям и методам класса.