Государственный Университет Молдовы

Факультет Математики и Информатики

Департамент Информатики

**Лабораторная работа №#3**

по курсу “Объектно ориентированное программирование”

тема: Перегрузка операций

Выполнил студент группы I2302:

Ciobanu Stanislav.,

Проверил преподаватель:

G.Latul

Кишинэу, 2023

**Условия работы**

Вариант 1 : Создайте класс Строка.

1. С обязательными член–данными: *длина строки, указатель на строку* (строка хранится в динамически выделенной памяти)
2. Создайте конструктор по заданной длине строки, конструктор по заданной строке, конструктор по другому объекту класса Строка, конструктор копирования.
3. Создайте методы: очистка строки, вывод строки.
4. Перегрузить (переопределить) операции: + (конкатенация ), -(разность: удаление символов входящих в строку второго операнда), < (меньше), > (больше).

**Задние 1**

Создаём класс Line

// Line.h

class Line

{

private:

int \_size;

char\* \_str;

public:

. . .

};

**Задние 2**

Создаём внутри класса Rectangle конструкторы

// Line.h

class Line

{

. . .

public:

Line(const char\* str);

Line(int size);

Line(const Line& line);

Line(Line&& line);

. . .

// Line.cpp

Line::Line(const Line& line)

{

\_size = line.\_size;

\_str = new char[\_size + 1];

int i = 0;

while (line.\_str[i] != '\0')

{

\_str[i] = line.\_str[i];

i++;

}

\_str[\_size] = '\0';

}

Line::Line(Line&& line)

{

\_size = line.\_size;

\_str = line.\_str;

line.\_str = nullptr;

}

Line::Line(int size) {

\_size = size;

\_str = new char[size];

}

Line::Line(const char\* str)

{

\_size = CountLengthOfString(str);

\_str = new char[\_size + 1];

for (int i = 0; i < \_size; ++i) {

\_str[i] = str[i];

}

\_str[\_size] = '\0';

}

**Задние 3**

Создаём методы очистки строки и вывода на экран

// line.h

class Line

{

. . .

public:

. . .

void ClearLine();

friend ostream& operator << (ostream& os, Line& line);

};

// main.cpp

ostream& operator << (ostream& os, Line& line)

{

int i = 0;

while (line.\_str[i] != '\0')

{

os << line.\_str[i];

i++;

}

return os;

}

// Line.cpp

void Line::ClearLine()

{

delete \_str;

\_size = 0;

\_str = nullptr;

}

**Задние 3**

Перегружаем операции +,-,<,>

// Line.h

class Line

{

. . .

public:

. . .

Line operator+(const char\* str);

Line operator-(const char\* str);

bool operator>(Line line);

bool operator<(Line line);

bool operator>(char\* str);

bool operator>(const char\* str);

bool operator<(char\* str);

. . .

};

// Line.cpp

Line Line::operator+(const char\* str)

{

. . .

}

Line Line::operator-(const char\* str)

{

. . .

}

bool Line::operator>(Line line)

{

. . .

}

bool Line::operator<(Line line)

{

. . .

}

**Используемые библиотеки**

<iostream> // Console

<fstream> // Потоки ввода и вывода

"Line.h" // Headers