Министерство науки и высшего образования Российской Федерации

федеральное государственное автономное образовательное учреждение высшего образования

«Национальный исследовательский университет ИТМО»   
(НИУ ИТМО)

Факультет программной инженерии и компьютерной техники

Дисциплина: Программирование на C++

Отчетпо Лабораторной работе № 2

Выполнил

Студент группы P4119

Суховей Я. Ю.

Преподаватель

Лаздин А. В.

Санкт-Петербург  
2024

# **Часть 1 Исходный код**

Исходный код программы:

<https://github.com/Yars2021/cpp-labs/tree/main/lab2>

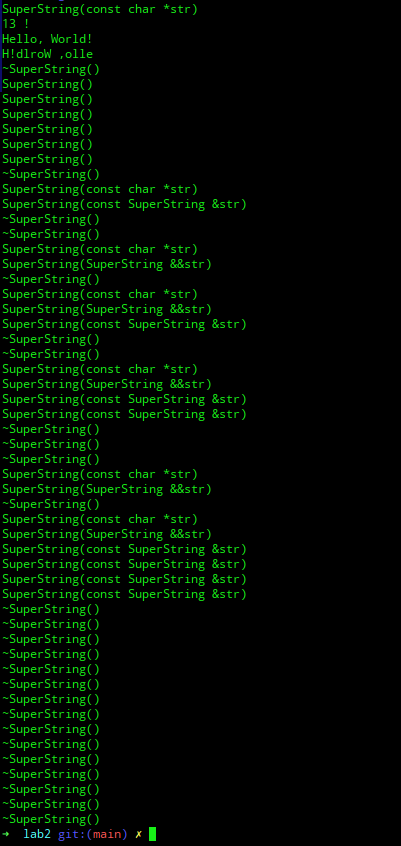
Основной класс .h:

#ifndef SUPERSTRING\_HPP  
#define SUPERSTRING\_HPP  
  
#include <string>  
  
class SuperString {  
 private:  
 unsigned int size;  
 char \*buffer = nullptr;  
 public:  
 SuperString();  
 SuperString(const char \*);  
 SuperString(const SuperString &);  
 SuperString(const std::string &);  
 SuperString(SuperString &&);  
 ~SuperString();  
 unsigned int length();  
 const char & operator[](int);  
 bool operator==(const SuperString &);  
 bool operator!=(const SuperString &);  
 bool operator>(const SuperString &);  
 bool operator<(const SuperString &);  
 bool operator>=(const SuperString &);  
 bool operator<=(const SuperString &);  
 void operator+=(const SuperString &);  
};  
  
#endif

Основной класс .cpp:

#include "superstring.hpp"  
#include <stdexcept>  
  
SuperString::SuperString() {  
 this->size = 0;  
 this->buffer = nullptr;  
}  
  
SuperString::SuperString(const char \*str) {  
 for (this->size = 0; str[this->size] != 0; this->size++);  
  
 this->buffer = new char[this->size];  
  
 for (unsigned int i = 0; i < this->size; i++) {  
 this->buffer[i] = str[i];  
 }  
}  
  
SuperString::SuperString(const SuperString &str) {  
 this->size = str.size;  
 this->buffer = new char[this->size];  
  
 for (unsigned int i = 0; i < str.size; i++) {  
 this->buffer[i] = str.buffer[i];  
 }  
}  
  
SuperString::SuperString(const std::string &str) {  
 this->size = str.size();  
 this->buffer = new char[this->size];  
  
 for (unsigned int i = 0; i < str.size(); i++) {  
 this->buffer[i] = str[i];  
 }  
}  
  
SuperString::SuperString(SuperString &&str) {  
 this->size = str.size;  
 std::swap(this->buffer, str.buffer);  
}  
  
SuperString::~SuperString() {  
 this->size = 0;  
 delete(this->buffer);  
}  
  
unsigned int SuperString::length() {  
 return this->size;  
}  
  
const char & SuperString::operator[](int index) {  
 if ((index > 0 && index >= this->size) || (index < 0 && -index >= this->size)) {  
 throw std::invalid\_argument("SuperString index out of bounds");  
 }  
 if (index >= 0) {  
 return this->buffer[index];  
 } else {  
 return this->buffer[this->size + index];  
 }  
}  
  
bool SuperString::operator==(const SuperString &other) {  
 if (this->size != other.size) return false;  
  
 for (unsigned int i = 0; i < this->size; i++) {  
 if (this->buffer[i] != other.buffer[i]) return false;  
 }  
  
 return true;  
}  
  
bool SuperString::operator!=(const SuperString &other) {  
 return !(\*this == other);  
}  
  
bool SuperString::operator>(const SuperString &other) {  
 if (\*this == other) return false;  
  
 unsigned int i = 0;  
  
 for (; i < this->size && i < other.size && this->buffer[i] == other.buffer[i]; i++);  
  
 if (this->size > other.size) return true;  
 if (this->size < other.size) return false;  
  
 return this->buffer[i] > other.buffer[i];  
}  
  
bool SuperString::operator<(const SuperString &other) {  
 return (\*this != other) && !(\*this > other);  
}  
  
bool SuperString::operator>=(const SuperString &other) {  
 return !(\*this < other);  
}  
  
bool SuperString::operator<=(const SuperString &other) {  
 return !(\*this > other);  
}  
  
void SuperString::operator+=(const SuperString &other) {  
 char \*buffer = new char[this->size + other.size];  
  
 for (unsigned int i = 0; i < this->size; i++) {  
 buffer[i] = this->buffer[i];  
 }  
  
 for (unsigned int i = 0; i < other.size; i++) {  
 buffer[this->size + i] = other.buffer[i];  
 }  
  
 this->size += other.size;  
 delete this->buffer;  
 this->buffer = buffer;  
}

# **Часть 2 Пример использования**



**Заключение**

В данной работе был реализован собственный класс на C++. Для него были переопределены различные конструкторы, операторы сравнения, оператор индекса. В тестовой программе все это было продемонстрировано.