1. Ministry of Education and Science of the Russian Federation
2. St. Petersburg State Polytechnic University
3. —
4. **Cybersecurity and Information Security department**

**REPORT**

**FOR Laboratory work № 1**

1. **«String class»**
2. subject «OOP»
3. Completed by
4. student from gr. 4851003/00802 M.V.Vologdin

<*signature*>

1. Checked by
2. teacher’s assistant G.S.Kubrin

<*signature*>

1. Saint – Petersburg

2022

1. **MAIN TASK**

* implement a class for representing a character string in C++, not using the STL library containers and algorithms;

1. **RESULTS OF WORK**

The development was carried out in 3 files: Source.cpp with the main function for testing the implemented class, MyString.cpp containing all the constructors and methods of the class, and MyString.h containing the class body and prototypes of its methods.

The class contains 3 private fields, 7 constructors, a destructor, 18 operator overload methods, 2 friend functions, and 27 regular methods, all of it you can see on 1-3 images.

Изображение выглядит как текст

Автоматически созданное описание

Image 1 – Body of the class MyString(1)

Изображение выглядит как текст

Автоматически созданное описание

Image 2 – Body of the class MyString(2)

Изображение выглядит как текст

Автоматически созданное описание

Image 3 – Body of the class MyString(3)

In all methods, the correctness of the input data is checked, if it is incorrect, then the function ends (in the case of void functions), or the function returns nullptr (in the case of functions returning a reference to a class object), or the value -1 is returned (in the case of functions returning int) .

Constructors process the output and fill in the fields of the class. For example, in the constructor we can see on the image 4 takes an array of characters. If the array is nonzero, memory is allocated for the string of the new object, and the input character array is copied into the allocated memory, then the length of the string and the amount of allocated memory are calculated and written in.

Изображение выглядит как текст

Автоматически созданное описание

Image 4 – Constructor of MyString class

In the case of overloaded operators: copying and adding strings is implemented using memory allocation and manipulation of string data. Comparison is performed using a specially implemented helper function compare(). The input and output operators are implemented using friend functions, as you can see on image 2 and 5.

Изображение выглядит как текст

Автоматически созданное описание

Image 5 – Output operator

Table 1 - Main class methods

|  |  |
| --- | --- |
| Function | Description |
| String | The constructor for the class. Converts the entry of the user to default form. |
| ~String | The class destructor. Clears the allocated memory. |
| operator+ | Concatenation of two strings, or a string and a character. |
| operator+= | Adding another line or character to the end of the line. |
| operator= | Value assignment |
| Operator>()  Operator<()  Operator>=()  Operator<=()  Operator!=()  Operator==() | Lexicographically comparing |
| Operator[] | Index operator |
| c\_str | Return a pointer to null-terminated character array |
| data | Return a pointer to array data that not required to be null-terminated |
| size | Return the number of char elements in string |
| capacity | Return the current amount of allocated memory for array |
| length | Same as size |
| empty | Return true if string is empty |
| clear | Remove all char element in string |
| shrink\_to\_fit | Reduce the capacity to size |
| Operator<< | Output string |
| Operator>> | Input string |
| insert | Inserts a line or a fragment of a line into the source string |
| erase | Removes a sequence of characters from a string |
| append | Adds a character sequence, string, or part of a string to the source string |
| replace | Replaces part of a string with multiple characters, another string, or part of another string |
| substr | Returns the substring of the source string |
| find | Searches for the first match in a string from a specific index |

An important part of this lab is code reuse. Each overloaded method contains only one implementation, and the rest of the algorithms simply refer to the implemented one. This allows you not to waste time writing the same type of code. As an example, on image 6 you can see the implementation of the concatenation operator with the input parameter MyString, the same operator, but with the input const char \* refers to the previous one.

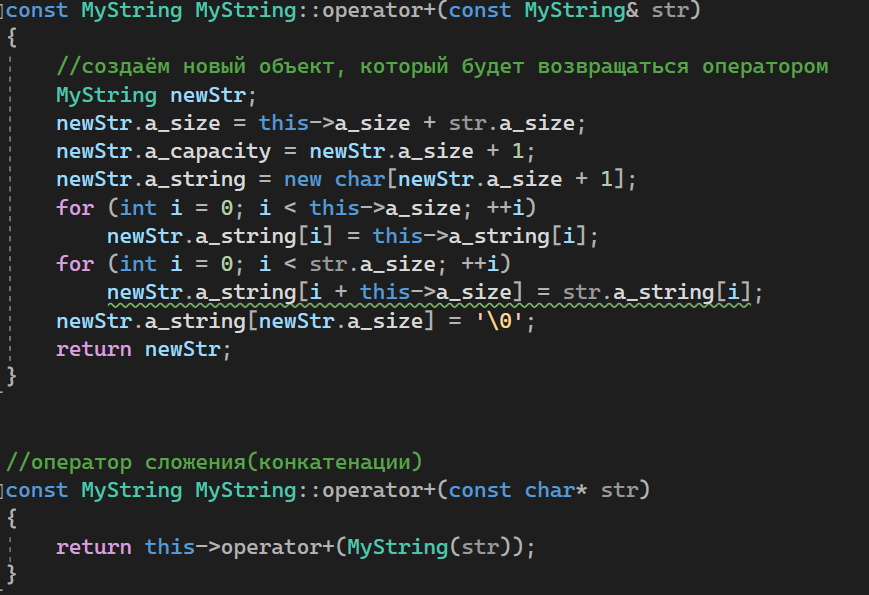


Image 6 – Reusing code

1. **TESTING**



Image 7 – Test constructor



Image 8 – Constructor test output

Изображение выглядит как текст, монитор, телевидение, экран

Автоматически созданное описание

Image 9 – Concatenation test



Image 10 – Concatenation test output

Изображение выглядит как текст

Автоматически созданное описание

Image 11 – Аssignment concatenate test



Image 12 – Аssignment concatenate test output

Изображение выглядит как текст

Автоматически созданное описание

Image 13 – Index operator test



Image 14 – Index operator test output

Изображение выглядит как текст, электроника, клавиатура

Автоматически созданное описание

Image 15 – Lexicographically comparing test



Image 16 – Lexicographically comparing test output

Изображение выглядит как текст, экран, счетчик, устройство

Автоматически созданное описание

Image 17 – Shrink\_to\_fit test



Image 18 – Shrink\_to\_fit test output

Изображение выглядит как текст

Автоматически созданное описание

Image 19 – Insert test



Image 20 – Insert test output

Изображение выглядит как текст

Автоматически созданное описание

Image 21 – Erase test



Image 22 – Erase test output

Изображение выглядит как текст

Автоматически созданное описание

Image 23 – Append test



Image 24 – Append test output

Изображение выглядит как текст

Автоматически созданное описание

Image 25 – Replace test



Image 26 – Replace test output

Изображение выглядит как текст

Автоматически созданное описание

Image 27 – Substr test



Image 28 – Substr test output

Изображение выглядит как текст

Автоматически созданное описание

Image 29 – Find test



Image 30 – Find test output

1. **CONCLUSION**

In the course of the work, basic knowledge of OOP and skills in working with the classes were obtained. The class MyString with the methods was created. In addition, I can note the complexity of implementing operator overloading, because it was not immediately possible to implement the addition operator and I did it for the first time.