**Міністерство освіти і науки України**

**Запорізький національний технічний університет**

кафедра програмних засобів

**ЗВІТ**

з лабораторної роботи №2

з дисципліни «Об’єктно-орієнтоване програмування» на тему:

Динамічні класові типи

Варіант № 7

Виконав

студент групи КНТ-126 Костюк П.М.

Прийняла

викладач: Твердохліб Ю.В.

2017

**Лабораторна робота №2**

**Динамічні класові типи**

**Мета роботи:**

Навчитись будувати класи та використовувати їх при створенні програм.

**Індивідуальні завдання**

Створити клас для виконання операцій з комплексними числами. Передбачити операції: складання та віднімання; множення та ділення; обчислення модулю; консольне введення та виведення; ініціалізацію.

MyString.h

#include **<cstring>**#include **<iostream>  
  
using namespace** std;  
  
  
**class** MyString {  
**protected**:  
 **char** \*str;  
  
**public**:  
 MyString(**char** \*);  
  
 size\_t count();  
  
 **friend** ostream &**operator**<<(ostream &, **const** MyString &);  
  
 MyString **operator**+(MyString &);  
  
 **bool** hasString(**char** \*);  
  
 MyString replace(**char**\* needle, **char**\*replacement);  
  
**private**:  
 **void** str\_replace(**char** \*target, **const char** \*needle, **const char** \*replacement);  
  
};

# MyString.cpp

#include **"MyString.h"**  
MyString::MyString(**char** \*str) {  
 **this**->str = str;  
}  
ostream &**operator**<<(ostream &out, **const** MyString &myString) {  
 out << myString.str;  
 **return** out;  
}  
size\_t MyString::count() {  
 **return sizeof this**->str - 1;  
}  
MyString MyString::**operator**+(MyString &myString) {  
 **this**->str = (**char** \*) strcat((**char** \*) **this**->str, (**char** \*) myString.str);  
 **return** \***this**;  
}  
**bool** MyString::hasString(**char** \*subString) {  
 **char** \*output = **NULL**;  
 output = strstr((**char** \*) **this**->str, (**char** \*) subString);  
  
 **return** output != **nullptr**;  
}  
**void** MyString::str\_replace(**char** \*target, **const char** \*needle, **const char** \*replacement){  
 **char** buffer[1024] = {0};  
 **char** \*insert\_point = &buffer[0];  
 **const char** \*tmp = target;  
 size\_t needle\_len = strlen(needle);  
 size\_t repl\_len = strlen(replacement);  
  
 **while** (1) {  
 **const char** \*p = strstr(tmp, needle);  
  
 *// walked past last occurrence of needle; copy remaining part* **if** (p == **NULL**) {  
 strcpy(insert\_point, tmp);  
 **break**;  
 }  
  
 *// copy part before needle* memcpy(insert\_point, tmp, p - tmp);  
 insert\_point += p - tmp;  
  
 *// copy replacement string* memcpy(insert\_point, replacement, repl\_len);  
 insert\_point += repl\_len;  
  
 *// adjust pointers, move on* tmp = p + needle\_len;  
 }  
  
 *// write altered string back to target* strcpy(target, buffer);  
}  
  
  
MyString MyString::replace(**char** \*needle, **char** \*replacement) {  
 str\_replace(**this**->str, needle, replacement);  
  
 **return** \***this**;  
}

# main.cpp

#include **<iostream>**#include **<cstring>**#include **"MyString.h"  
  
using namespace** std;  
  
  
**int** main() {  
  
 **char** s[] = **"string"**;  
  
 **char** st[] = **"st"**;  
  
 **char** rp[] = **"mt"**;  
  
 MyString myString(s);  
  
  
 cout << **"Replacing:"** << myString.replace(st,rp) << endl;  
  
 string check = myString.hasString(st) ? **"true"**: **"false"**;  
  
 cout << **"Checking if has a string "** << st << **": "** << check << endl;  
  
 cout << **"Count string: "** << myString.count() << endl;  
  
 MyString myString1 = myString + myString;  
  
 cout << **"Concatenate string: "** << myString1 << endl;  
 **return** 0;  
}