**МІНІСТЕРСТВО ОСВІТИ І НАУКИ**

**НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАІНИ**

**«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ»**

**ФАКУЛЬТЕТ ПРИКЛАДНОЇ МАТЕМАТИКИ**

**КАФЕДРА СИСТЕМНОГО ПРОГРАМУВАННЯ І СПЕЦІАЛІЗОВАНИХ КОМП`ЮТЕНИХ СИСТЕМ**

**ЛАБОРАТОРНАЯ РОБОТА №1**

з дисципліни «**Об’єктно-орієнтоване програмування 2**»

Тема: «**Класи**»

|  |  |
| --- | --- |
|  | Виконав  студент 2 курсу  гр. КВ-43  Вітовщик Михайло Васильович  Перевірив(ла):  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |

Київ – 2016

***Постановка задачі***

Сконструювати клас cString.

***Текст програми***

**StringList.h**

#pragma once

#include <string.h>

#include <stdio.h>

class cString{

public:

//constructors/destructor

cString();

cString( const char \*psz );

cString( const cString& stringsrc );

~cString();

//methods

int Getlength( ) const;

bool Isempty()const;

void Empty();

void SetAt( int nindex, char ch );

int Compare( const cString& s ) const;

int Find( char ch ) const;

int Find( char \*pszsub ) const;

cString Mid( int nfirst, int ncount) const;

cString Left( int ncount ) const;

cString Right( int ncount ) const;

//operators

cString& operator =( const cString& stringsrc );

const cString& operator =( const unsigned char\* psz );

char operator [](int indx);

cString operator +( const cString& string );

cString& operator +=( const cString& string );

//

void Print();

private:

char \*string;

int size;

};

**StringList.cpp**

#include "StringList.h"

cString::cString(){

string = NULL;

size = 0;

}

cString::cString(const char \*psz){

if (!psz){

string = NULL;

size = 0;

return;

}

size = strlen(psz);

string = new char[size + 1];

strcpy(string, psz);

string[size] = '\0';

}

cString::cString(const cString& stringsrc){

if (!stringsrc.string){

string = NULL;

size = 0;

return;

}

size = strlen(stringsrc.string);

string = new char[size + 1];

strcpy(string, stringsrc.string);

string[size] = '\0';

}

cString::~cString(){

delete[] string;

}

int cString::Getlength() const{

return size;

}

bool cString::Isempty() const{

if (size) return true;

return false;

}

void cString::Empty(){

delete[] string;

string = NULL;

size = 0;

}

void cString::SetAt(int nindex, char ch){

if (nindex < 0 || nindex >= size || ch == '\0') return;

string[nindex] = ch;

}

int cString::Compare(const cString& s) const{

return strcmp(string, s.string);

}

int cString::Find(char ch) const{

if (ch == '\0') return -1;

char \*str = strchr(string, ch);

if (str) return str - string;

return -1;

}

int cString::Find(char \*pszsub) const{

if(!pszsub) return -1;

char \*str = strstr(string, pszsub);

if(str) return str - string;

return -1;

}

cString cString::Mid(int nfirst, int ncount) const{

cString str;

int i;

if (ncount > size)

ncount = size - nfirst;

str.size = ncount;

str.string = new char[ncount + 1];

for(i = nfirst; i < ncount; i++)

str.string[i - nfirst] = string[i];

str.string[ncount] = '\0';

return str;

}

cString cString::Left(int ncount) const{

cString str;

int i;

if (ncount > size) ncount = size;

str.size = ncount;

str.string = new char[ncount + 1];

for(i = 0; i < ncount; i++)

str.string[i] = string[i];

str.string[ncount] = '\0';

return str;

}

cString cString::Right(int ncount) const{

cString str;

int i;

if(ncount > size) ncount = size;

str.size = ncount;

str.string = new char[ncount + 1];

for(i = 0; i < ncount; i++)

str.string[i] = string[size - i - 1];

str.string[ncount] = '\0';

return str;

}

cString& cString::operator =(const cString& stringsrc){

if (this == &stringsrc) return \*this;

delete[] string;

size = stringsrc.size;

string = new char[size+1];

strcpy(string, stringsrc.string);

string[size] = '\0';

return \*this;

}

const cString& cString::operator =(const unsigned char\* psz){

delete[] string;

if (!psz){

string = NULL;

return \*this;

}

size = strlen(string);

string = new char[size+1];

strcpy(string, (char\*)psz);

string[size] = '\0';

return \*this;

}

char cString::operator [](int indx){

return string[indx];

}

cString cString::operator +(const cString& stringsrc){

char \* str = new char[size + stringsrc.size + 1];

if (size) strcpy(str, string);

delete[] string;

string = str;

str += size;

if(stringsrc.size) strcpy(str, stringsrc.string);

str[stringsrc.size + 1] = '\0';

return \*this;

}

cString& cString::operator +=(const cString& stringsrc){

char \* str = new char[size + stringsrc.size + 1];

if (size) strcpy(str, string);

delete[] string;

string = str;

str += size;

if(stringsrc.size) strcpy(str, stringsrc.string);

str[stringsrc.size + 1] = '\0';

return \*this;

}

void cString::Print(){

if (string)

printf("%s", string);

}

**main.cpp**

#include "StringList.h"

int main(){

cString a;

cString b("abcd");

cString c(b);

char \*str = new char[2];

strcpy(str,"qd");

printf("----------------------------------------------------------\n");

printf("cString: ");

a.Print();

printf("\nlength: %d\n", a.Getlength());

printf("is empty: %d\n", a.Isempty());

printf("\n");

printf("cString: ");

b.Print();

printf("\nlength: %d\n", b.Getlength());

printf("is empty: %d\n", b.Isempty());

printf("\n");

printf("cString: ");

c.Print();

printf("\nlength: %d\n", c.Getlength());

printf("is empty: %d\n", c.Isempty());

c.Empty();

printf("\nempty c: string = ");

c.Print();

printf("\n\tsize = %d\n\n", c.Getlength());

printf("string = ");

b.Print();

printf("\nSetAt : ch = 'q' nindex = 2\n");

b.SetAt(2, 'q');

printf("string = ");

b.Print();

printf("\n\nCompare: ");

b.Print();

printf(" abcd = %d\n\n", b.Compare("abcd"));

printf("Find ch = 'd', str = \"qd\":\n");

printf("\tstring = ");

b.Print();

printf("\n\t'd' - %d, \"qd\" - %d\n\n", b.Find('d'), b.Find(str));

printf("string = ");

b.Print();

printf("\n\tMid(2,100): ");

a = b.Mid(2, 100);

a.Print();

printf("\n\tLeft(2): ");

a = b.Left(2);

a.Print();

printf("\n\tRight(3): ");

a = b.Right(3);

a.Print();

printf("\nb[1] = %c\n", b[1]);

printf("\na = ");

a.Print();

printf("\nb = ");

b.Print();

c = a + b;

printf("\nc = a + b = ");

c.Print();

a += b;

printf("\na += b = ");

a.Print();

printf("\n");

printf("----------------------------------------------------------\n");

return 0;

}