**Міністерство освіти та науки України**

**Національний технічний університет України**

**«Київський політехнічний інститут»**

**Факультет прикладної математики**

**Кафедра системного програмування і спеціалізованих**

**комп’ютерних систем**

**Лабораторна робота №1**

з дисципліни

**«Об’єктно-орієнтоване програмування»**

Виконав: Місік Дмитро Сергійович

Студент групи КВ-31

Варіант №1

Перевірив(ла):\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Київ 2015

1. Файл Multistring.h

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

name: Multistring.h

description: header file that contains prototypes of functions

that descriped in file "Multistring.cpp"

author: Dima

date of creation: 12.01.2015

written: 13.01.2015

date of last change: 13.01.2015

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#pragma once

class Multistring**{**

public**:**

//constructors destructor

Multistring**();**

Multistring**(**int**);**

Multistring**(**const Multistring **&);**

**~**Multistring**();**

//methods

Multistring **&** **operator=(**const Multistring **&**ms**);**

char **\*** **operator[]** **(**int nindex**)** const**;**

Multistring **&** **operator+=(**const Multistring **&);**

Multistring Mergemultistringexclusive**(**const Multistring **&);**

int Find**(**const char **\***pszsub**)** const**;**

int Getlength**()** const**;**

bool Isempty**()**const**{** **return** str\_nmb **==** 0**;** **};**

void Empty**();**

void Setat**(**int nindex**,** const char**\*** str**);**

void Printstr**(**int nindex**)** const**;**

void Print**()** const**;**

private**:**

//attributes

char **\*\***buf**;**//pointer to vector

int str\_nmb**;**//strings number

**};**

1. Файл Multistring.cpp

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

name: Multistring.cpp

description: file contains declarations of methods of class Multistring

it contains 3 constructors:

Multistring() - creates empty vector

Multistring(int n) - creates vector with n empty strings

Multistring(const Multistring &) - copy constructor

Empty() - cleans vector

operator[](int index) - returns copy of index string

operator+=(const Multistring &) - concatenation of all strings

of vector

operator=(const Multistring &) - copy vector

Mergemultistringexclusive(const Multistring &) - returns vector

that include only ellements that meets once

Setat(int nindex, const char\* str) - set ta nindex copy of str

Find(const char \*pszsub) - returns index of pszsub in vector

or -1, if ellement is absent

Getlength() - returns length of vector

Printstr(int nindex) - prints nindex string of vector

Print() - print all ellements of vector

author: Dima

date of creation: 12.01.2015

written: 13.01.2015

date of last change: 13.01.2015

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include "Multistring.h"

#include <string>

Multistring**::**Multistring**()** **{**

buf **=** **new** char **\*[**1**];**

str\_nmb **=** 1**;**

buf**[**0**]** **=** **new** char**[**1**];**

buf**[**0**][**0**]** **=** '\0'**;**

**}**

Multistring**::**Multistring**(**int n**)** **{**

buf **=** **new** char **\*[**n**];**

str\_nmb **=** n**;**

**for** **(**int i **=** 0**;** i **<** n**;** i**++)** **{**

buf**[**i**]** **=** **new** char**[**1**];**

buf**[**i**][**0**]** **=** '\0'**;**

**}**

**}**

Multistring**::**Multistring**(**const Multistring **&**old**)** **{**

buf **=** **new** char **\*[**old**.**str\_nmb**];**

str\_nmb **=** old**.**str\_nmb**;**

**for** **(**int i **=** 0**;** i **<** str\_nmb**;** i**++)** **{**

buf**[**i**]** **=** **new** char**[**strlen**(**old**.**buf**[**i**])** **+** 1**];**

strcpy**(**buf**[**i**],** old**.**buf**[**i**]);**

**}**

**}**

void Multistring**::**Empty**()** **{**

**if** **(!**Isempty**())** **{**

**for** **(**int i **=** 0**;** i **<** str\_nmb**;** i**++)** **{**

**delete[]** buf**[**i**];**

**}**

**delete[]** buf**;**

str\_nmb **=** 0**;**

**}**

**}**

char **\*** Multistring**::operator[]** **(**int index**)** const **{**

**if** **(**index **<** str\_nmb**)** **{**

char **\***s **=** **new** char**[**strlen**(**buf**[**index**])** **+** 1**];**

strcpy**(**s**,** buf**[**index**]);**

**return** s**;**

**}**

**return** **NULL;**

**}**

Multistring **&** Multistring**::operator+=(**const Multistring **&**src**)** **{**

int m**;**

**if** **(**src**.**str\_nmb **<** str\_nmb**)** m **=** src**.**str\_nmb**;**

**else** m **=** str\_nmb**;**

**for** **(**int i **=** 0**;** i **<** m**;** i**++)** **{**

char **\***s **=** **new** char**[**strlen**(**buf**[**i**])** **+** strlen**(**buf**[**i**])** **+** 1**];**

s**[**0**]** **=** '\0'**;**

strcat**(**s**,** buf**[**i**]);**

strcat**(**s**,** src**.**buf**[**i**]);**

**delete[]** buf**[**i**];**

buf**[**i**]** **=** s**;**

**}**

**return** **\*this;**

**}**

Multistring **&** Multistring**::operator=(**const Multistring **&**ms**)** **{**

int m **=** ms**.**str\_nmb **<** str\_nmb **?** ms**.**str\_nmb **:** str\_nmb**;**

Multistring **\***res **=** **new** Multistring**(**m**);**

**for** **(**int i **=** 0**;** i **<** m**;** i**++)** **{**

char **\***s **=** **new** char**[**strlen**(**ms**.**buf**[**i**])** **+** 1**];**

strcpy**(**s**,** ms**.**buf**[**i**]);**

**delete[]** buf**[**i**];**

buf**[**i**]** **=** s**;**

**}**

**return** **\***res**;**

**}**

Multistring Multistring**::**Mergemultistringexclusive**(**const Multistring **&**src**)** **{**

int num **=** str\_nmb **+** src**.**str\_nmb**;**

**if** **(**num **!=** 0**)** **{**

Multistring **\***temp **=** **new** Multistring**(**num**);**

temp**->**str\_nmb **=** num**;**

**\***temp **+=** **\*this;**

**for** **(**int i **=** 0**;** i **<** src**.**str\_nmb**;** i**++)** temp**->**Setat**(**i **+** str\_nmb**,** src**.**buf**[**i**]);**

**for** **(**int i **=** 0**;** i **<** temp**->**str\_nmb **-** 1**;** i**++)** **{**

**if** **(**temp**->**buf**[**i**][**0**]** **!=** '\0'**)** **{**

bool flag **=** **false;**

**for** **(**int j **=** i **+** 1**;** j **<** temp**->**str\_nmb**;** j**++)** **if** **(**strcmp**(**temp**->**buf**[**i**],** temp**->**buf**[**j**])** **==** 0**)** **{**

temp**->**buf**[**j**][**0**]** **=** '\0'**;**

num**--;**

flag **=** **true;**

**}**

**if** **(**flag**)** **{**

temp**->**buf**[**i**][**0**]** **=** '\0'**;**

num**--;**

**}**

**}**

**}**

**if** **(**num **!=** 0**)** **{**

Multistring **\***res **=** **new** Multistring**(**num**);**

res**->**str\_nmb **=** num**;**

int in\_res **=** 0**,** in\_temp **=** 0**;**

**while** **(**in\_temp **<** temp**->**str\_nmb**)** **{**

**if** **(**temp**->**buf**[**in\_temp**][**0**]** **!=** '\0'**)** **{**

res**->**Setat**(**in\_res**,** temp**->**buf**[**in\_temp**]);**

in\_res**++;**

**}**

in\_temp**++;**

**}**

**return** **\***res**;**

**}**

**}**

**return** Multistring**();**

**}**

void Multistring**::**Setat**(**int nindex**,** const char**\*** str**)** **{**

**if** **(**nindex **<** str\_nmb**)** **{**

char **\***s **=** **new** char**[**strlen**(**str**)** **+** 1**];**

strcpy**(**s**,** str**);**

buf**[**nindex**]** **=** s**;**

**}**

**}**

Multistring**::~**Multistring**()** **{**

Empty**();**

**}**

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

**for** **(**int i **=** 0**;** i **<** str\_nmb**;** i**++)** **{**

**if** **(**strcmp**(**buf**[**i**],** pszsub**)** **==** 0**)** **return** i**;**

**}**

**return** **-**1**;**

**}**

int Multistring**::**Getlength**()** const **{**

int n **=** 0**;**

**for** **(**int i **=** 0**;** i **<** str\_nmb**;** i**++)** **{**

n **+=** strlen**(**buf**[**i**]);**

**}**

**return** n**;**

**}**

void Multistring**::**Printstr**(**int nindex**)** const **{**

printf**(**"%s\n"**,** buf**[**nindex**]);**

**}**

void Multistring**::**Print**()** const **{**

**for** **(**int i **=** 0**;** i **<** str\_nmb**;** i**++)** Printstr**(**i**);**

**}**

1. Файл test.cpp

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

name: test.cpp

description: test file

author: Dima

date of creation: 12.01.2015

written: 13.01.2015

date of last change: 13.01.2015

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include "Multistring.h"

#include <stdio.h>

int main**()** **{**

Multistring **\***M1 **=** **new** Multistring**(**5**);**

Multistring **\***M2 **=** **new** Multistring**(**5**);**

Multistring **\***M3 **=** **new** Multistring**(**6**);**

M2**->**Setat**(**0**,** "start"**);**

M2**->**Setat**(**1**,** "begin"**);**

M2**->**Setat**(**2**,** "middle"**);**

M2**->**Setat**(**3**,** "finish"**);**

M2**->**Setat**(**4**,** "end"**);**

M3**->**Setat**(**0**,** "1"**);**

M3**->**Setat**(**1**,** "2"**);**

M3**->**Setat**(**2**,** "3"**);**

M3**->**Setat**(**3**,** "4"**);**

M3**->**Setat**(**4**,** "5"**);**

M3**->**Setat**(**5**,** "6"**);**

**\***M1 **=** **\***M2**;**

**\***M2 **+=** **\***M3**;**

Multistring **\***M4 **=** **new** Multistring**(\***M2**);**

Multistring **\***M5 **=** **new** Multistring**(**M4**->**Mergemultistringexclusive**(\***M3**));**

**delete** M5**;**

M5 **=** **new** Multistring**(**M4**->**Mergemultistringexclusive**(\***M2**));**

M2**->**Setat**(**0**,** "start1"**);**

M2**->**Setat**(**1**,** "begin"**);**

M2**->**Setat**(**2**,** "middle3"**);**

M2**->**Setat**(**3**,** "finish"**);**

M2**->**Setat**(**4**,** "end5"**);**

**delete** M5**;**

M5 **=** **new** Multistring**(**M4**->**Mergemultistringexclusive**(\***M2**));**

printf**(**"M1\n"**);**

M1**->**Print**();**

printf**(**"\nM2\n"**);**

M2**->**Print**();**

printf**(**"\nM3\n"**);**

M3**->**Print**();**

printf**(**"\nM4\n"**);**

M4**->**Print**();**

printf**(**"\nM5\n"**);**

M5**->**Print**();**

printf**(**"4 in M3 has index %d\n"**,** M3**->**Find**(**"4"**));**

printf**(**"length of M3 is %d\n"**,** M3**->**Getlength**());**

printf**(**"length of M5 is %d\n"**,** M5**->**Getlength**());**

**delete** M1**;**

**delete** M2**;**

**delete** M3**;**

**delete** M4**;**

**return** 0**;**

**}**