|  |  |  |
| --- | --- | --- |
| American University of SharjahCollege of Engineering Department of Computer Science & Engineering  P. O. Box 26666  Sharjah, UAE |  | **Lab Instructor:** Eng. Sameer Alawnah  **Office:** EB2-101  **Phone**: 971-6-5152974  **e-mail**: salawnah@aus.edu  **Semester**: Fall 16 |

**CMP 220L – Introduction to Computer Science II**

**Lab 5**

**Note: The good programmer adds comments to his/her code. Add comments to your code.**

**Question 1:**

Given the following class definition:

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

American University of Sharjah ( AUS )

Dept of Computer Science & Eng.

P.O.Box 26666

Sharjah, UAE

Fall 2016

CMP220L

LAB5 – Q1

Revision: 04.

Date: 9th of October, 2015. 12:23 pm.

By: Eng. Sameer A. Alawnah

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

#include<iostream>

using namespace std;

class Color

{

private:

int red, green, blue;

public:

Color(){ red = 0; green = 0; blue = 0; }

void setRed(int r){ red = r; }

void setGreen(int g){ green = g; };

void setBlue(int b){ blue = b; }

int getRed(){ return red; }

int getGreen(){ return green; }

int getBlue(){ return blue; }

};

Color addition or subtracting is done by adding/subtracting each component of the first color with/from the corresponding component of the second color, red with red, green with green and blue with blue.

Since the color components ranges from 0 to 255, if you got a result greater than 255 then you have to truncate it to 255, if the result is less than 0 then you have to truncate it back to 0.

1. Add two member functions to the class, one for addition and one for subtraction and do the tasks given in the main() function.
2. Add two non-member and non-friend functions, one for addition and one for subtraction and do the tasks given in the main() function.
3. Add two non-member but friend functions, one for addition and one for subtraction and do the tasks given in the main() function.
4. Add the following operators to the code:
   * Add operator (+) as a friend function.
   * Subtract binary operator (-) as a friend function.
   * Unary (-) operator. Assume that the (-colorComponent is equal to 255 – colorComponent, ex. if the color is [R:122,G:245,B:100] then –color is [R:133,G:10,B:155].
   * == Operator. (Returns true if all colors components of both colors are equal, false otherwise.

Write a main () functions to test all the operators that you developed.

void main()

{

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

Task 1: create two objects of type Color name them col1 and col2

Read their values from the user

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

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

Task 2: add col1 and col2 and print the result

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

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

Task 3: subtract col1 from col2 and print the result

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

}

Note: you are supposed to write 4 different programs (one for each point of the above).

#include<iostream>

using namespace std;

class Color

{

private:

int red, green, blue;

public:

Color(){ red = 0; green = 0; blue = 0; }

void setRed(int r){ red = r; }

void setGreen(int g){ green = g; };

void setBlue(int b){ blue = b; }

int getRed(){ return red; }

int getGreen(){ return green; }

int getBlue(){ return blue; }

//member func---------------------------------------------

void AddColor(Color &col1)

{

col1.red = red + col1.red;

col1.green = green + col1.green;

col1.blue = blue + col1.blue;

}

void SubtColor(Color &col2)

{

col2.red = red - col2.red;

col2.green = green - col2.green;

col2.blue = blue - col2.blue;

}

//-------------friend-----------------------------------

friend Color Add(Color &col1, Color&col2)

{

Color clr;

clr.red =col1.red+col2.red;

clr.green = col1.green + col2.green;

clr.blue = col1.blue + col2.blue;

}

friend Color Sub(Color &col1, Color&col2)

{

Color clr;

clr.red = col1.red - col2.red;

clr.green = col1.green - col2.green;

clr.blue = col1.blue - col2.blue;

}

//operators--------------------------------------------------------------

friend Color operator+(Color c1,Color c2)

{

Color c3;

c3.red = c1.getRed() + c2.getRed();

c3.blue = c1.getBlue() + c2.getBlue();

c3.green = c1.getGreen() + c2.getGreen();

return(c3);

}

friend Color operator-(Color c1, Color c2)

{

Color c3;

c3.red = c1.getRed() - c2.getRed();

c3.green = c1.getGreen() - c2.getGreen();

c3.blue = c1.getBlue() - c2.getBlue();

return(c3);

}

//UNARY???

friend Color operator-(Color c1)

{

Color c3;

c3.red = 255 - c1.red;

c3.blue = 255 - c1.blue;

c3.green = 255 - c1.green;

return (c3);

}

friend bool Color::operator==(Color &col1,Color &col2)

{

if (col1.red == col2.red && col1.blue == col2.blue && col1.green == col2.green)

{

return (1);

}

return(-1);

}

};

//non-member---------------------------------------

Color ADD(Color&col1, Color &col2)

{

int red, green, blue;

red = col1.getRed() + col2.getRed();

green = col1.getGreen() + col2.getGreen();

blue = col1.getBlue + col2.getBlue;

}

Color SUB(Color &col1, Color &col2)

{

int red, green, blue;

red = col1.getRed() - col2.getRed();

green = col1.getGreen() - col2.getGreen();

blue = col1.getBlue - col2.getBlue;

}

//-------------------------------------------------

void main()

{

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

Task 1: create two objects of type Color name them col1 and col2

Read their values from the user

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

Color col1, col2;

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

Task 2: add col1 and col2 and print the result

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

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

Task 3: subtract col1 from col2 and print the result

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

}

**Question 2:**

Given the following input file:

input.txt

Ahmad Mohamad 509443432

John Jack 62320034

Orient West 524332345

Oliver Twist 54323432

Complete all required tasks in the following:

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

American University of Sharjah ( AUS )

Dept of Computer Science & Eng.

P.O.Box 26666

Sharjah, UAE

Fall 2016

CMP220L

LAB5 - Q2.

Revision: 02.

Date: 9th of October, 2015. 12:58 am.

By: Eng. Sameer A. Alawnah

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

#include<iostream>

#include<string>

#include<vector>

using namespace std;

class Contact{

private:

string fname,lname;

int phone;

public:

Contact(); // Default constructor, initialize the contact to None None 0

Contact(string fn,string ln,int ph); // you have to implement the contstructor using the initalizer list

string getFfirstName(); // get the first name of the contact and return it.

string getLastName(); // get the last name of the contact and return it.

int getPhone();// get the phone # of the contact and return it.

string getFullName();// get the full name of the contact and return it.

void print();// print the full name and the phone # for the contact

};

class ContactList{

private:

vector<Contact> list; // vector of contacts

public:

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

Constructor with parameter ( fileName).

This constructor should read the file

given in fileName and save the content

into the list.

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

ContactList(string fileName);

// a function to add contact con to list.

// return the contact at index, exit the program with error if index is out of range

Contact getContactAt(int index);

//return the index of contact in which firstName is equal to target, -1 otherwise.

int searchByFristName(string target);

};

int main()

{

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

Task 1: Implement all the functions in the

two classes above.

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

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

Task 2: create an object of type contact

list with the data loaded from the file

inputs.txt using the constructor

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

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

Task 3: ask the user for the target he want to search for (search by first name ) and print

the found contact if exist, error message otherwise.

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

return 0;

}

#include<iostream>

#include <string>

#include<vector>

#include<fstream>

using namespace std;

class Contact

{

private:

string fname, lname;

int phone;

string fullname;

public:

Contact() // Default constructor, initialize the contact to None None 0

{

fname = "None";

lname = "None";

phone = 0;

}

Contact(string fn, string ln, int ph) // you have to implement the contstructor using the initalizer list

{

fname = fn;

lname = ln;

phone = ph;

}

string getFirstName()

{

return fname;

} // get the first name of the contact and return it.

string getLastName()

{

return lname;

}// get the last name of the contact and return it.

int getPhone()

{

return phone;

}// get the phone # of the contact and return it.

string getFullName()

{

fullname = fname +" " +lname;

return fullname;

}// get the full name of the contact and return it.

void print()

{

cout << "Full Name: " << getFullName() << endl << "Phone #: " << phone << endl;

}// print the full name and the phone # for the contact

};

class ContactList{

private:

vector<Contact> list; // vector of contacts

int index;

public:

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

Constructor with parameter ( fileName).

This constructor should read the file

given in fileName and save the content

into the list.

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

ContactList();

ContactList(string fileName)

{

ifstream mycin(fileName);

string fn,ln;

int ph;

if (mycin.fail())

{

cout << "ERROR, file not found" << endl;

exit(1);

}

while (!mycin.eof())

{

mycin >> fn >> ln >> ph;

Contact clist(fn, ln, ph);

list.push\_back(clist);

}

}

// a function to add contact con to list.

// return the contact at index, exit the program with error if index is out of range

Contact getContactAt(int index)

{

if ((index>list.size()) || (index < 0))

{

cout << "ERROR, program is out of range" << endl;

exit(1);

}

return list[index];

}

//return the index of contact in which firstName is equal to target, -1 otherwise.

int searchByFirstName(string target)

{

for (int i = 0; i < list.size();i++)

{

if (list[i].getFirstName()==target)

{

return i;

}

}

return -1;

}

};

int main()

{

string target;

int i;

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

Task 1: Implement all the functions in the

two classes above.

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

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

Task 2: create an object of type contact

list with the data loaded from the file

inputs.txt using the constructor

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

ContactList clist("input.txt");

Contact lst;

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

Task 3: ask the user for the target he want to search for (search by first name ) and print

the found contact if exist, error message otherwise.

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

cout << "Please enter first name of the targeted person: " << endl;

cin >> target;

int index = clist.searchByFirstName(target);

if (index == -1)

{

cout << "ERROR, person not found" << endl;

}

if (index != -1)

{

lst=clist.getContactAt(index);

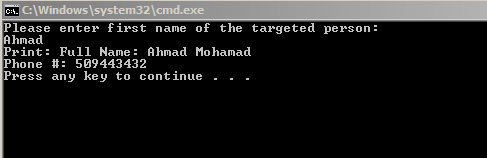
cout << "Print: ";

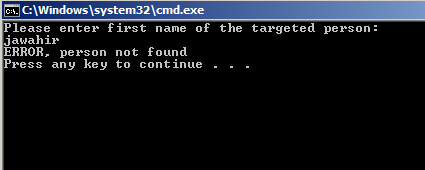
lst.print();

}

return 0;

}





Good Luck ☺