**Person class updated version**

#include <iostream>

#include <string>

#include <vector>

#include <cstdlib>

#include <ctime>

using namespace std;

class person {

string name;

int age;

static int totalPeople;

public:

person();

person(string, int);

int getAge();

string getName();

void setAge(int);

void setName(string);

void print();

static int getTotal();

static void incPerson();

static void decPerson();

};

//static variable initialization

int person::totalPeople = 0;

person::person() {

setAge(1);

setName("not defined");

}

person::person(string newName, int newAge) {

setAge(newAge);

setName(newName);

}

int person::getAge() { return age; }

string person::getName() { return name; }

void person::setAge(int newAge) {

age = newAge;

}

void person::setName(string newName) {

name = newName;

}

void person::print() {

cout << "name: " << name << endl;

cout << "age: " << age << endl;

}

int person::getTotal()

{

return totalPeople;

}

void person::incPerson() {

totalPeople++;

}

void person::decPerson() {

totalPeople--;

}

void addPerson(vector<person>&);

void displayPeople(vector<person>&);

void deletePerson(vector<person>&);

int main() {

vector<person> people;

bool stop = false;

int opt;

do {

cout << "1. Add Person" << endl;

cout << "2. Delete Person" << endl;

cout << "3. Display All" << endl;

cout << "4. Display Total" << endl;

cout << "5. Exit" << endl;

cout << "enter your option ==> ";

cin >> opt;

cin.ignore();

switch (opt) {

case 1:

addPerson(people);

break;

case 2:

deletePerson(people);

break;

case 3:

displayPeople(people);

break;

case 4:

cout << "Total People: " << person::getTotal() << endl;

break;

case 5:

stop = true;

break;

default:

cout << "wrong option" << endl;

}

system("pause");

system("cls");

} while (!stop);

return 0;

}

void addPerson(vector<person>& p)

{

string pName;

int pAge;

cout << "enter name: ";

getline(cin, pName);

cout << "enter age: ";

cin >> pAge;

p.push\_back(person(pName, pAge));

person::incPerson();

}

void displayPeople(vector<person>& people) {

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

people[i].print();

}

void deletePerson(vector<person>& p)

{

string k;

bool found = false;

cout << "enter the person to be deleted: ";

getline(cin, k);

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

{

if (k == p[i].getName())

{

p.erase(p.begin() + i);

found = true;

person::decPerson();

}

}

if (!found)

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

}

**operators not members of the class**

**#include <iostream>**

**using namespace std;**

**class coord {**

**public:**

**coord();**

**coord(int, int);**

**int getX() const;**

**int getY() const;**

**void setX(int);**

**void setY(int);**

**void print();**

**private:**

**int y;**

**int x;**

**};**

**coord::coord() {**

**setX(0);**

**setY(0);**

**}**

**coord::coord(int newX, int newY) {**

**setX(newX);**

**setY(newY);**

**}**

**int coord::getX() const{**

**return x;**

**}**

**int coord::getY() const{**

**return y;**

**}**

**void coord::setX(int newX) {**

**x = newX;**

**}**

**void coord::setY(int newY) {**

**y = newY;**

**}**

**void coord::print() {**

**cout << "(" << x << "," << y << ")" << endl;**

**}**

**coord operator-(const coord& c) {**

**return coord(-c.getX(), -c.getY());**

**}**

**bool operator<(const coord& left, const coord& right) {**

**/\***

**bool less = false;**

**if (left.getX() < right.getX() && left.getY() < right.getY())**

**less = true;**

**return less;**

**\*/**

**return left.getX() < right.getX() && left.getY() < right.getY();**

**}**

**coord operator+(const coord& left, const coord& right) {**

**/\***

**coord temp;**

**temp.setX(left.getX() + right.getX());**

**temp.setY(left.getY() + right.getY());**

**return temp;**

**\*/**

**return coord(left.getX() + right.getX(), left.getY() + right.getY());**

**}**

**int main()**

**{**

**coord c1(1, 1), c2(2, 2), result;**

**c1.print();**

**c2.print();**

**if (c1 < c2)**

**cout << "less" << endl;**

**return 0;**

**}**

**operators as members of the class**

**#include <iostream>**

**using namespace std;**

**class coord {**

**public:**

**coord();**

**coord(int, int);**

**int getX() const;**

**int getY() const;**

**void setX(int);**

**void setY(int);**

**void print();**

**coord operator+(const coord&);**

**coord operator-();**

**bool operator<(const coord&);**

**private:**

**int y;**

**int x;**

**};**

**coord::coord() {**

**setX(0);**

**setY(0);**

**}**

**coord::coord(int newX, int newY) {**

**setX(newX);**

**setY(newY);**

**}**

**int coord::getX() const{**

**return x;**

**}**

**int coord::getY() const{**

**return y;**

**}**

**void coord::setX(int newX) {**

**x = newX;**

**}**

**void coord::setY(int newY) {**

**y = newY;**

**}**

**void coord::print() {**

**cout << "(" << x << "," << y << ")" << endl;**

**}**

**coord coord::operator+(const coord& right) {**

**return coord(x+right.getX(),y+right.getY());**

**}**

**coord coord::operator-() {**

**return coord(-x,-y);**

**}**

**bool coord::operator<(const coord& right) {**

**return x < right.getX() && y < right.getY();**

**}**

**int main()**

**{**

**coord c1(1, 1), c2(2, 2), result;**

**c1.print();**

**c2.print();**

**//result = c1 + c2;**

**result = c1.operator+(c2);**

**result.print();**

**result = -c1;**

**result.print();**

**if (c1 < c2)**

**cout << "less" << endl;**

**return 0;**

**}**