Автор: Єлєсін Артем , КІТ-119б

Дата: 01.06.2020

Лабораторна робота №1

Класи та специфікатори доступу. Інкапсуляція. Константи.

***Мета роботи***: отримати базові знання про класи. Дослідити механізм інкапсуляції.

Завдання до роботи

Для предметної галузі з табл. 1.2 розробити два класи:

- клас, що відображає сутність «базового класу». При цьому, в даному класі повинно бути мінімум три числових поля (бажано, щоб одне з цих полів було унікальним ідентифікатором об’єкта);

- клас, що має у собі динамічний масив об’єктів базового класу та має в собі методи додавання, видалення елемента, отримання елемента по індексу (або ідентифікатору), вивід усіх елементів на екран. Рекомендовані сигнатури методів:

- додавання: void CList::addPhone(Phone& phone);

- видалення: void CList::removePhone(int index);

- отримання по індексу: CPhone& CList::getPhone(int index);

- виведення усіх елементів: void CList::showAll(); при цьому цей метод повинен викликати метод getPhone(index), щоб не було дублювання коду.

Опис класів

Базовий клас: Ccooperator

Клас, що має в собі масив базового класу та методи для роботи з ним: CList

Опис змінних

int amount - кількість елементів

Ccooperator\* fEl - 1 масив

Ccooperator\* fEl1 - 2 масив

int id - Id персоны

int age - вік

int salary -заробітна плата

Опис методів

void setId(const int id); - встановлює id.

void setAge(const int age);- встановлює вік.

void setSalary(const int salary); - встановлює заробітну плату.

int getId()const; - повертає id.

int getAge()const; - повертає вік.

int getSalary()const; - повертає заробітну плату.

void creatMass(int a); - створює масив.

cooperator creatEl1(); - створює елемент.

cooperator creatEl2(); - створює елемент.

void Add(cooperator); - додає елемент.

void Delete(int ); - видаляє елемент.

cooperator getCooperator(int a); - повертає елемент.

void showAll(); - показує всі елементи.

cooperator findCooperator(const int a); - знаходить елемент.

int getAmount(); - повертає кількість елементів.

void End(); - видаляє всі масиви.

Текст програми

Cooperator.h

#pragma once

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <iostream>

#include <clocale>

using namespace std;

class cooperator

{

private:

int id, age, salary;

public:

void setId(const int id);

void setAge(const int age);

void setSalary(const int salary);

int getId()const;

int getAge()const;

int getSalary()const;

}

Cooperator.cpp

#include "cooperator.h"

void cooperator::setId(const int id) {

this->id = id;

}

void cooperator::setAge(const int age) {

this->age = age;

}

void cooperator::setSalary(const int salary) {

this->salary = salary;

}

int cooperator::getId()const {

return this->id;

}

int cooperator::getAge()const {

return this->age;

}

int cooperator::getSalary()const {

return this->salary;

}

List.h

#pragma once

#include "cooperator.h"

class list {

private:

int amount;

cooperator\* fEl;

cooperator\* fEl1;

public:

void creatMass(int a);

cooperator creatEl1();

cooperator creatEl2();

void Add(cooperator);

void Delete(int );

cooperator getCooperator(int a);

void showAll();

cooperator findCooperator(const int a);

int getAmount();

void End();

};

List.cpp

#include "list.h"

void list::creatMass(int a)

{

amount = a;

//printf("Введите количество елементов ");

//scanf("%i", &amount);

fEl = new cooperator[amount];

//int a;

//printf("\nВыбирите вариант создания елементов\n1. Cоздать елемент вручную\n2. Готовый елемент\nВаш выбор: ");

//scanf("%i", &a);

//if (a == 1)

//for (int i = 0; i < amount; i++) {

//fEl[i] = creatEl1();

//}

//if (a == 2)

for (int i = 0; i < amount; i++) {

fEl[i] = creatEl2();

}

}

cooperator list::creatEl1() {

cooperator El;

int a;

printf("Введите id сотрудника: ");

scanf("%i", &a);

El.setId(a);

printf("Введите зарплату сотрудника: ");

scanf("%i", &a);

El.setSalary(a);

printf("Введите возраст сотрудника: ");

scanf("%i", &a);

El.setAge(a);

return El;

}

cooperator list::creatEl2() {

cooperator El;

El.setId(0);

El.setSalary(0);

El.setAge(0);

return El;

}

void list::Add(cooperator El1) {

fEl1 = new cooperator[amount + 1];

for (int i = 0; i < amount; i++) {

fEl1[i] = fEl[i];

}

fEl1[amount] = El1;

delete[] fEl;

amount++;

fEl = new cooperator[amount];

for (int i = 0; i < amount; i++) {

fEl[i] = fEl1[i];

}

delete[] fEl1;

}

int list::getAmount() {

return amount;

}

void list::Delete(int a) {

cooperator\* fEl1 = new cooperator[amount - 1];

for (int i = 0; i < a - 1; i++) {

fEl1[i] = fEl[i];

}

for (int i = a - 1, j = a; j < amount; i++, j++) {

fEl1[i] = fEl[j];

}

delete[] fEl;

amount--;

fEl = new cooperator[amount];

for (int i = 0; i < amount; i++) {

fEl[i] = fEl1[i];

}

delete[] fEl1;

}

cooperator list::findCooperator(const int a) {

int b=-1, count=0;

for (int i=0; i < amount; i++) {

if (a == fEl[i].getId()) {

printf("ID: %i\n Age: %i\n Salary: %i\n", getCooperator(i).getId(), getCooperator(i).getAge(), getCooperator(i).getSalary());

count++;

b = i;

}

}

if (count > 1) {

printf("Есть %i похожих елементов", count);

}

if (count == 0) {

printf("Похожих елементов нет, возвращен 1 елемент");

return fEl[0];

}

return fEl[b];

}

cooperator list::getCooperator(const int a) {

return fEl[a];

}

void list::showAll() {

for (int i = 0; i < amount; i++) {

printf("ID: %i\n Age: %i\n Salary: %i\n", getCooperator(i).getId(), getCooperator(i).getAge(), getCooperator(i).getSalary());

}

}

void list::End() {

delete[] fEl;

}

Source.cpp

#include "cooperator.h"

#include "list.h"

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <iostream>

#include <clocale>

using namespace std;

int main() {

setlocale(LC\_ALL, "rus");

list a1;

cooperator a2;

a2.setId(111);

a2.setAge(19);

a2.setSalary(15000);

a1.creatMass(2);

//a1.showAll();

a1.Add(a2);

//a1.showAll();

a1.Delete(2);

//a1.showAll();

a1.findCooperator(111);

a1.End();

if (\_CrtDumpMemoryLeaks())

printf("\nMemory leack deteckted\n");

else

printf("\nMemory is not leack deteckted\n");

}

Test.cpp

#include "cooperator.h"

#include "list.h"

#define \_CRT\_SECURE\_NO\_WARNINGS

#define N 5

#include <iostream>

#include <clocale>

using namespace std;

int main() {

setlocale(LC\_ALL, "rus");

cooperator a;

a.setAge(0);

a.setAge(0);

a.setSalary(0);

list a1[N];

int test[N];

int rezult1[N];

int rezult2[N];

int rezult3[N];

test[0] = 1;

test[1] = 5;

test[2] = 10;

test[3] = 25;

test[4] = 50;

rezult1[0] = 1;

rezult1[1] = 5;

rezult1[2] = 10;

rezult1[3] = 25;

rezult1[4] = 50;

rezult2[0] = 2;

rezult2[1] = 6;

rezult2[2] = 11;

rezult2[3] = 26;

rezult2[4] = 51;

rezult3[0] = 1;

rezult3[1] = 5;

rezult3[2] = 10;

rezult3[3] = 25;

rezult3[4] = 50;

for (int i = 0; i < N; i++) {

a1[i].creatMass(test[i]);

if (a1[i].getAmount() == rezult1[i]) {

printf("Тест 1.%i пройден\n", i);

}

else {

printf("Тест 1.%i не пройден\n", i);

}

}

for (int i = 0; i < N; i++) {

a1[i].Add(a);

if (a1[i].getAmount() == rezult2[i]) {

printf("Тест 2.%i пройден\n", i);

}

else {

printf("Тест 2.%i не пройден\n", i);

}

}

for (int i = 0; i < N; i++) {

a1[i].Delete(test[i]);

if (a1[i].getAmount() == rezult3[i]) {

printf("Тест 3.%i пройден\n", i);

}

else {

printf("Тест 3.%i не пройден\n", i);

}

}

}

Висновок

При виконанні даної лабораторної роботи було набуто практичного досвіду роботи з потоками.

Програма протестована, витоків пам’яті немає, виконується без помилок