Автор: Єлєсін Артем , КІТ-119б

Дата: 01.06.2020

Лабораторна робота 3. ПОТОКИ

Мета роботи: отримати знання про основи роботи з потоковим введенням / виведенням на мові C++, роботу з файлами та рядками типу string. Загальне завдання.

Поширити попередню лабораторну роботу таким чином:

- використання функцій printf/scanf замінити на використання cin/cout;

- усі конкатенації рядків замінити на використання stringstream;

- замінити метод виводу інформації про об’єкт на метод, що повертає рядок-інформацію про об’єкт, який далі можна виводити на екран;

- замінити метод вводу інформації про об’єкт на метод, що приймає рядок з інформацією про об’єкт, обробляє його та створює об’єкт на базі цієї інформації;

- поширити клас-список, шляхом реалізації методів роботи з файлами за допомогою файлових потоків (fstream) (якщо використовувалися функції fprintf/fscanf – замінити їх на класи ifsteam/ofstream), при цьому сигнатури методів повинні виглядати таким чином:

- читання: void CList::readFromFile(string fileName); де CList – клас-список об’єктів, при цьому слід пам’ятати, що при повторному читанні з файлу, попередні дані списку повинні бути очищені;

- запис: void CList::writeToFile(string fileName);

Опис класів

Базовий клас: Ccooperator

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

Опис змінних

const char* name – ім’я.

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.

Ccooperator(); - конструктор.

Ccooperator(int a, int b, int c, const char* d); - конструктор с параметрами.

Ccooperator(const Ccooperator& a) – конструктор копіювання.

~Ccooperator() {} – деструктор.

int averageSalary() – середня заробітна плата.

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 Ccooperator
{
private:

	int id, age, salary;
	string name;
public:

	Ccooperator();
	Ccooperator(int a, int b, int c, string d);
	Ccooperator(const Ccooperator& a);
	void setId(const int id);
	void setAge(const int age);
	void setSalary(const int salary);
	void setName(string name);
	const string getName();
	int getId()const;
	int getAge()const;
	int getSalary()const;
	~Ccooperator();
};
```

Cooperator.cpp

```
#include "cooperator.h"
void Ccooperator::setName(string name) {
	this->name = name;
}
const string Ccooperator::getName() {
	return this->name;
}
void Ccooperator::setId(const int id) {
	this->id = id;
}
void Ccooperator::setAge(const int age) {
	this->age = age;
}
void Ccooperator::setSalary(const int salary) {
	this->salary = salary;
}
```

```
int Ccooperator::getId()const {
	return this->id;
}
int Ccooperator::getAge()const {
	return this->age;
}
int Ccooperator::getSalary()const {
	return this->salary;
}
Ccooperator::Ccooperator() :id(0), age(0), salary(0), name("Ivan") {
	cout << "\nБыл вызван конструктор по умолчанию в обекте с id: " << id<<"\n";

}
Ccooperator::Ccooperator(const Ccooperator& a) :id(a.id), age(a.age), salary(a.salary),
name(a.name) {
	cout << "\nБыл вызван конструктор по умолчанию в обекте с id: " << id<<"\n";
}
Ccooperator::Ccooperator(int a , int b , int c , string d ) :id(a), age(b), salary(c),
name(d) {
	cout << "\nБыл вызван конструктор по умолчанию в обекте с id: " << id<<"\n";
}
Ccooperator::~Ccooperator() {

}
```

List.h

```
#pragma once
#include "cooperator.h"
#include "heder.h"
class CList {
private:
	int amount;
	Ccooperator* fEl;
	Ccooperator* fEl1;
public:
	int averageSalary();
	void writeToFile(string fileName);
	void readFromFile(string fileName);
	void creatMass(int a);
	//Ccooperator creatEl1();
	Ccooperator creatEl2();
	void Add(Ccooperator);
	void Delete(int b);
	void AddWhithString();
	string findCooperator(const int a);
	Ccooperator getCooperator(int a);
	void showAll();
	int getAmount();
	void End();
};
```

List.cpp

```
#include "list.h"
#include <sstream>
#include <iostream>
#include <fstream>
```

```
void CList::creatMass(int a)
{
	amount = a;
	//printf("Введите количество елементов ");
	//scanf("%i", &amount);
	fEl = new Ccooperator[amount];
	//int a;
	//printf("\nВыбирите вариант создания елементов\n1. Создать 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();
	}
}
string CList::findCooperator(const int a) {
	std::stringstream ss;
	string ab;
	ss << " ";
	int b = -1, count = 0;
	for (int i = 0; i < amount; i++) {
		if (a == fEl[i].getId()) {
			count++;
			b = i;
		}
	}
	if (count >= 1) {
		cout << "Есть " << count << " похожих елементов, будет возвращен последний
элемент";
		ss << "\nId: " << fEl[b].getId() << "\nAge:" << fEl[b].getAge() <<
"\nSalary: " << fEl[b].getSalary() << "\nName " << fEl[b].getName();
		ab = ss.str();
		return ab;
	}
	if (count == 0) {
		cout << "Похожих елементов нет, возвращен пустой символ";
		return ab;
	}
}
/*Ccooperator CList::creatEl1() {
	Ccooperator 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;
}*/
```

```
int CList::averageSalary() {
	int averageSalary = 0;
	for (int i = 0; i < amount; i++)
	{
		averageSalary = averageSalary + fEl[i].getSalary();
	}
	return averageSalary = averageSalary / amount;

};
Ccooperator CList::creatEl2() {
	Ccooperator El;
	El.setId(0);
	El.setSalary(0);
	El.setAge(0);
	return El;
}
void CList::readFromFile(string fileName) {
	End();
	creatMass(0);
	int a, b, c;
	string d;
	Ccooperator temp;
	ifstream file;
	file.open(fileName);
	if (!file.is_open())
	{
		cout << " Файл не открыт, давай по новой Миша\n";
		return;
	}
	while (!file.eof()) {
		file >> a >> b >> c >> d;
		temp.setId(a);
		temp.setAge(b);
		temp.setSalary(c);
		temp.setName(d);
		Add(temp);
	}
	file.close();
}
void CList::writeToFile(string fileName) {
	ofstream file;
	string str1;
	std::stringstream ss;
	file.open(fileName);
	if (!file.is_open())
	{
		cout << " Файл не открыт, давай по новой Миша\n";
		return;
	}
	for (int i = 0; i < amount; i++) {
		file << fEl[i].getId() << " " << fEl[i].getAge() << " " <<
fEl[i].getSalary() << " " << fEl[i].getName() << "\n";
	}

	file.close();
}
void CList::AddWhithString() {
	Ccooperator temp;
```

```
	std::stringstream ss1, ss2, ss3, ss4;
	fEl1 = new Ccooperator[amount + 1];
	for (int i = 0; i < amount; i++) {
		fEl1[i] = fEl[i];
	}

	std::cout << "\nВведите данные с клавиатуры в таком порядке: id, age, salary,
name\n";
	string tid = " ", tage = " ", tsalary = " ", tname = " ";
	int tid1, tage1, tsalary1;
	string tname1 = " ";
	cin >> tid >> tage >> tsalary >> tname;
	ss1 << tid;
	ss1 >> tid1;
	ss2 << tage;
	ss2 >> tage1;
	ss3 << tsalary;
	ss3 >> tsalary1;
	ss4 << tname;
	ss4 >> tname1;
	temp.setId(tid1);
	temp.setSalary(tsalary1);
	temp.setAge(tage1);
	temp.setName(tname1);

	fEl1[amount] = temp;
	delete[] fEl;
	amount++;
	fEl = new Ccooperator[amount];
	for (int i = 0; i < amount; i++) {
		fEl[i] = fEl1[i];
	}
	delete[] fEl1;
}
void CList::Add(Ccooperator El1) {
	fEl1 = new Ccooperator[amount + 1];
	for (int i = 0; i < amount; i++) {
		fEl1[i] = fEl[i];
	}
	fEl1[amount] = El1;
	delete[] fEl;
	amount++;
	fEl = new Ccooperator[amount];
	for (int i = 0; i < amount; i++) {
		fEl[i] = fEl1[i];
	}
	delete[] fEl1;
}
int CList::getAmount() {
	return amount;
}
void CList::Delete(int a) {
	Ccooperator* fEl1 = new Ccooperator[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 Ccooperator[amount];
	for (int i = 0; i < amount; i++) {
		fEl[i] = fEl1[i];
	}
	delete[] fEl1;
}
Ccooperator CList::getCooperator(const int a) {
	return fEl[a];
}
void CList::showAll() {
	for (int i = 0; i < amount; i++) {
		cout << "ID:" << getCooperator(i).getId() << "\n Age: " <<
getCooperator(i).getAge() << "\n Salary: " << getCooperator(i).getSalary() << "\n Name: "
<< getCooperator(i).getName();
	}
}
void CList::End() {
	delete[] fEl;
}
```

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");
	Ccooperator a;
	a.setAge(0);
	a.setAge(0);
	a.setSalary(0);
	CList 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);
		}
	}
	Ccooperator Obtest1, Obtest2;
	Obtest1.setSalary(-200);
	Obtest2.setSalary(300);
	CList TestList;
	TestList.creatMass(0);
	TestList.Add(Obtest1);
	TestList.Add(Obtest2);
	if (TestList.averageSalary() == 50) {
		printf("Тест 4 пройден\n");
	}
	else {
		printf("Тест 4 не пройден\n");
	}
	TestList.readFromFile("File.txt");
	if (TestList.getAmount() == 3)
		printf("Тест 5 пройден\n");
	else
		printf("Тест 5 не пройден\n");
}
```

Висновок

При виконанні даної лабораторної роботи було набуто практичного досвіду роботи з читанням з файлу, stringstream, cin/cout.

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