Федеральное государственное образовательное бюджетное учреждение

высшего образования «Финансовый университет при Правительстве

Российской Федерации»

Отчёт по дисциплине «Учебная Практика»

Практическая работа №10

Выполнил студент

группы 2ПКС-116

Зайцев Н. В.

Москва 2018

**Вариант 9**

1. Создать класс с именем NOTE, содержащий следующие закрытые поля:

\* фамилия;

\* номер телефона;

Написать программу, выполняющую следующие действия:

\* ввод с клавиатуры данных в массив, состоящий 4 объектов типа NOTE;

\* вывод на экран информации обо всех людях.

\* вывод на экран информации о людях, чьи телефоны начинаются на 499;

\* если таких людей нет, выдать на дисплей соответствующее сообщение.

\* Вывести адреса объектов и размер;

Задание 2. Создание односвязного списка (2 часа) На условиях задания 1 создать список из 4 узлов.

Предусмотреть следующие действия:

\* Инициализация списка

\* Добавление элемента в начало списка или добавление элемента в конец списка

\* Показ всех элементов списка с адресами и размерами узлов

\* Удаление всех элементов списка

\* Определение количества элементов списка

\* Проверка списка на пустоту

\* Удаление первого элемента или удаление последнего элемента

\* Поиск данного значения в списке

\* Удаление элемента списка с данным значением.

**Код к заданию 1**

#include <iostream>  
#include <string>  
  
using namespace std;  
  
int msize = 4;  
int size = 0;  
  
void wait() {  
 char a;  
 cout << "Type 'c' to continue" << endl;  
 cin >> a;  
}  
  
class Note {  
private:  
 string family, number;  
public:  
 Note(string \_fam, string \_num) {  
 family = \_fam;  
 number = \_num;  
 }  
  
 string getNumber() {  
 return number;  
 }  
  
 string getFamiliy() {  
 return family;  
 }  
  
 void show() {  
 cout << "Объект: " << this << endl;  
 cout << "Размер объекта: " << sizeof(\*this) << endl;  
 cout << "Фамилия: " << family << endl;  
 cout << "Телефон: " << number << endl;  
 }  
};  
  
Note \*mas[4];  
  
void vvod() {  
 string fam, num;  
 if (size == 3) {  
 cout << "Массив заполнен!" << endl;  
 return;  
 }  
 for (int i = 0; i < msize; i++) {  
 cout << "Введите " << i << " элемент:" << endl;  
 cout << "Введите телефон:" << endl;  
 cin >> num;  
 cout << "Введите фамилию:" << endl;  
 cin >> fam;  
 mas[i] = new Note(fam, num);  
 size++;  
 }  
}  
  
void showAll() {  
 if(size == 0){  
 cout << "Массив пуст!" << endl;  
 return;  
 }  
 for (int i = 0; i < msize; i++) {  
 cout << "- - - - -" << endl;  
 mas[i]->show();  
 cout << "- - - - -" << endl;  
 }  
}  
  
void show499() {  
 bool showed = false;  
 for (int i = 0; i < msize; i++) {  
 if ((mas[i]->getNumber().substr(0, 4) == "8499") || (mas[i]->getNumber().substr(0, 4) == "7499")) {  
 cout << "- - - - -" << endl;  
 mas[i]->show();  
 cout << "- - - - -" << endl;  
 showed = true;  
 }  
 }  
 if (!showed) {  
 cout << "Таких элементов нет!" << endl;  
 }  
}  
  
  
int main() {  
 int sw = 0;  
 cout << "1)Ввести элементы" << endl;  
 cout << "2)Вывести все элементы" << endl;  
 cout << "3)Вывести все элементы номера которых начинаются на 499" << endl;  
 cout << "4)Выход" << endl;  
 cin >> sw;  
 switch (sw) {  
 case 1:{  
 cout << "----------------" << endl;  
 vvod();  
 break;  
 }  
 case 2: {  
 cout << "----------------" << endl;  
 showAll();  
 break;  
 }  
 case 3: {  
 cout << "----------------" << endl;  
 show499();  
 break;  
 }  
 case 4:{  
 exit(0);  
 }  
 default:  
 break;  
 }  
 wait();  
 cout << "----------------" << endl;  
 main();  
 return 0;  
}

**Код к заданию 2**

#include <iostream>  
#include <string>  
  
using namespace std;  
  
  
void wait() {  
 char a;  
 cout << "Type 'c' to continue" << endl;  
 cin >> a;  
}  
  
class Note {  
private:  
 string family, number;  
public:  
 Note(string \_fam, string \_num) {  
 family = \_fam;  
 number = \_num;  
 }  
  
 string getNumber() {  
 return number;  
 }  
  
 string getFamiliy() {  
 return family;  
 }  
  
 void show() {  
 cout << "Объект: " << this << endl;  
 cout << "Размер объекта: " << sizeof(\*this) << endl;  
 cout << "Фамилия: " << family << endl;  
 cout << "Телефон: " << number << endl;  
 }  
};  
  
struct Item {  
 Note \*item = nullptr; // имя переменной  
 Item \*next;// ссылка на следующий элемент  
};  
  
Item \*first = nullptr, \*p;  
int m;  
int sw;  
  
  
void add() {  
 string fam, num;  
 cout << "----------ADD-----------" << endl;  
 if (first == nullptr) { // создание начала стека (если не создан)  
 first = new (Item);  
 p = first;  
 }  
 while (true) {  
 // Вводить числа, пока не введем 0  
 cout << "Input 0 for stop" << endl;  
 cout << "Input number:" << endl;  
 cin >> num;  
 if (num == "0")  
 break;  
 cout << "Input family:" << endl;  
 cin >> fam;  
 p->item = new Note(fam, num);  
 p->next = new (Item);  
 p = p->next;  
 }  
}  
  
  
void delitem() {  
 Item \*iter = first;  
 first = iter->next;  
 cout << "---------DELETE---------" << endl;  
 iter->item->show();  
 cout << "Was deleted!" << endl;  
 delete iter;  
}  
  
void delspis() {  
 Item \*iter;  
 cout << "-------DELETE-ALL-------" << endl;  
 while (first->item != nullptr) {  
 iter = first;  
 first = iter->next;  
 delete iter;  
 iter->item->show();  
 }  
}  
  
void show() {  
 Item \*iter = first;  
 cout << "----------SHOW----------" << endl;  
 if (iter->item == nullptr) {  
 cout << "Nothing to show!" << endl;  
 return;  
 }  
 while (iter->item != nullptr) {  
 iter->item->show();  
 iter = iter->next;  
 }  
}  
  
void SearchAndDelete(string &fam) {  
 cout << "----SEARCH-N-DELETE-----" << endl;  
 Item \*iter = first;  
 Item \*s\_iter = nullptr;  
 if (iter->item == nullptr) {  
 cout << "Nothing to search!" << endl;  
 return;  
 }  
 while (iter->item != nullptr) {  
 if (iter->item->getFamiliy() == fam) {  
 cout << "Element was founded and removed!" << endl;  
 if (s\_iter != nullptr) {  
 s\_iter->next = iter->next;  
 delete iter;  
 } else {  
 first = iter->next;  
 delete iter;  
 }  
 break;  
 }  
 s\_iter = iter;  
 iter = iter->next;  
 }  
}  
  
void Search(string &fam) {  
 cout << "---------SEARCH---------" << endl;  
 Item \*iter = first;  
 if (iter->item == nullptr) {  
 cout << "Nothing to search!" << endl;  
 return;  
 }  
 while (iter != nullptr) {  
 if (iter->item->getFamiliy() == fam) {  
 cout << "Element was founded!" << endl;  
 iter->item->show();  
 break;  
 }  
 iter = iter->next;  
 }  
  
}  
  
  
int main() {  
 sw = 0;  
 cout << "List of Notes menu: " << endl;  
 cout << "1) Add (create) list of Notes" << endl;  
 cout << "2) Show list" << endl;  
 cout << "3) Delete first item" << endl;  
 cout << "4) Delete all list" << endl;  
 cout << "5) Search element" << endl;  
 cout << "6) Search and delete" << endl;  
 cout << "7) Exit" << endl;  
 cin >> sw;  
 switch (sw) {  
 case 1: {  
 add();  
 break;  
 }  
 case 2: {  
 show();  
 break;  
 }  
 case 3: {  
 delitem();  
 break;  
 }  
 case 4: {  
 delspis();  
 break;  
 }  
 case 5: {  
 string \*fam = new string();  
 cout << "Enter the search value:" << endl;  
 cin >> \*fam;  
 Search(\*fam);  
 delete fam;  
 break;  
  
 }  
 case 6: {  
 string \*fam = new string();  
 cout << "Enter the search value:" << endl;  
 cin >> \*fam;  
 SearchAndDelete(\*fam);  
 delete fam;  
 break;  
 }  
 case 7: {  
 exit(0);  
 }  
 default:  
 break;  
 }  
 wait();  
 main();  
 return 0;  
}

**Скриншоты к Заданию**

