МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ ОДЕСЬКИЙ ОДЕСЬКИЙ НАЦІОНАЛЬНИЙ ПОЛІТЕХНІЧНИЙ УНІВЕРСИТЕТ ІНСТИТУТ КОМП’ЮТЕРНИХ СИСТЕМ

КАФЕДРА ІНФОРМАЦІЙНІ ТЕХНОЛОГІЇ

КУРСОВА РОБОТА

з дисципліни «АЛГОРИТМІЗАЦІЯ ТА ПРОГРАМУВАННЯ»

на тему: “ Розробка автоматизованої інформаційної системи ”

Студента І курсу групи АД-191 Кузьмин А.Д.

Керівник:

ст.викладач Косенко О.Д.

Загальна оцінка

Одеса – 2020

ЗМІСТ

[ЗАВДАННЯ НА КУРСОВУ РАБОТУ 3](#_Toc42093423)

[АНОТАЦІЯ 5](#_Toc42093424)

[ВСТУП 6](#_Toc42093425)

[СТВОРЕННЯ СУБД 8](#_Toc42093426)

[Головна функція програми: «int main» 8](#_Toc42093427)

[Структура «AccountingForEquipmentComposition» 11](#_Toc42093428)

[Функції програми 12](#_Toc42093429)

[ВИСНОВОК 19](#_Toc42093430)

[**ДОДАТОК Б** 20](#_Toc42093431)

ОДЕСЬКИЙ НАЦІОНАЛЬНИЙ ПОЛІТЕХНІЧНИЙ УНІВЕРСИТЕТ

Інститут комп’ютерних систем Кафедра інформаційні технології

Освітньо-кваліфікаційний рівень Бакалавр

за спеціальністю 126 – Інформаційні системи та технології

ЗАВДАННЯ НА КУРСОВУ РАБОТУ

З ДИСЦИПЛІНИ АЛГОРИТМІЗАЦИЯ ТА ПРОГРАМУВАННЯ

студенту Кузьміну Антону Дмитровичу

Групи АД-191

Тема курсової роботі: «Розробка автоматизованої інформаційної системи»

Розробити автоматизовану інформаційну систему засобами мови програмування С++ для роботи с базою даних, з вказаними полями, яка б дозволяла надавати користувачеві можливість:

1. вводити дані з клавіатури і заносити їх у базу (файл);
2. читати дані з бази (файлу);
3. змінювати записи в базі даних;
4. виконувати пошук потрібної інформації за умовою (згідно варіанту);
5. упорядковувати записи в базі даних за умовою (згідно варіанту);
6. виконати розрахунок (згідно варіанту).

Необхідно написати програму, що демонструє роботу з базою даних. Необхідно виводити меню для вибору заданих операцій або для завершення програми. Проект програми повинен бути багатофайловим, тобто прототипи функцій повинні бути у заголовному файлі, а визначення функцій – у файлі-джерелі.

Дані для 9 варіанту:

Назва БД: Облік складу обладнання

Поля бази даних: Назва обладнання, Кількість обладнання,

Серійні номери, Ціна за одиницю.

Запит: Знайти за серійним номером назву обладнання

Поле та вид упорядкування: Упорядкувати за зменшенням по ціні обладнання

Розрахунок: Підрахувати вартість обладнання по кожному серійному номеру

АНОТАЦІЯ

В роботі була розроблена Система Управління Бази Даних яка приймає дані які вводить користувач і заносить їх в файл. Користувач може шукати, видаляти або редагувати дані в базі даних. Также СУБД может выполнять сортировку і виводити всі дані які були введені в таблицю.

ANNOTATION

In the course work was developed Database Management System that receives data entered by the user and enters them into a file. The user can search , delete or edit data in the database. Also, the database can perform sorting and display all the data that were entered into the table.

ВСТУП

Моя курсова робота написана на мові програмування С ++, вона широко використовується для розробки програмного забезпечення, будучи однією з найпопулярніших мов програмування. Область її застосування включає створення операційних систем, різноманітних прикладних програм, драйверів пристроїв, додатків для вбудованих систем, високопродуктивних серверів, а також ігор. Мова має багату стандартну бібліотеку, яка включає в себе поширені контейнери і алгоритми, введення-виведення, регулярні вирази, підтримку багатопоточності і інші можливості.

При розробці були використані деякі стандартні бібліотеки мови С++:

<iostream> - для роботи функцій вводу-виводу програми.

<iomanip> - для можливості маніпулювання при роботі с потоковими операціями.

<fstream> - для можливості відкривати файли одночасно для читання і запису.

<vector> - для можливості користуватися векторами.

У програмі було використану структуру, кожна строка у таблиці – запис структурної змінної. Структура - це, якесь об'єднання різних змінних (навіть з різними типами даних), якому можна присвоїти ім'я.

Для кожного пункту меню була використана окрема функція. Функції створені для того щоб скоротити програму і виконують певні Дії. Кожну з них можна викликати ть в будь-якій частині програми

Для тимчасового збереження даних був використаний вектор. Одні з головних його достоїнств – самостійність вектора у виділенні собі необхідного розміру, а також відсутність необхідності вручну звільняти пам’ять. У програмі були використані саме вектори, тому що сам користувач має вирішувати як багато структурних змінних він хоче мати у базі даних.

Для зберігання даних було використано бінарний файл. Программа зберігає усі зміни в базі даних у файл. Це робиться для того, щоб після повторого відкриття програми користувач мав доступ до даних, які він вніс раніше. Весь процес йде по аналогії зі звичайним текстовим файлом, але є деякі нюанси. Робота йде саме з бітами, так що можливий запис або зчитування різних керуючих символів, наприклад, "\ n", який закінчує рядок і починає новий.

Система Управління Базами Даних(СУБД) розроблена під час виконання курсової роботи – програма, яка дозволяє користувачеві зручно взаємодіяти з базою даних. База даних - це найкращий спосіб зберігання даних їх. У ній можна шукати потрібні дані або відсортувати їх за певним критерієм.

# СТВОРЕННЯ СУБД

Завдання курсової роботи полягало у створенні програми, яка могла б виконувати функції СУБД. Програма має виконувати розрахунки, пошук, упорядкування таблиці за вартістю, виводити таблицю та виконувати запит наступної дії через меню дій.

## Головна функція програми: «int main»

«int main» - головна функція програми, саме з початку цієї функції починається виконання коду. На початку функції було створенно вектор типу структури для тимчасового зберігання даних, та створення змінної яка буде зберігати розмір вектора (кількість його елементів). Також на початку програми виконується функція що завантажує дані бінарного файлу у цей ж вектор. Ось так це виглядає в коді:

int main()

{

fstream file("Equipment List.txt", ios::binary | ios::out | ios::app);

streampos begin, end;

begin = file.tellg();

file.seekg(0, ios::end);

end = file.tellg();

int size = (end - begin) / sizeof(AccountingForEquipmentComposition);

file.close();

vector<AccountingForEquipmentComposition>EquipmentVector;

Меню

У другій частині головної функції виводиться меню. Перше, що бачить користувач коли запускає програму – меню, де йому пропонують ввести цифру, яка відповідає потрібному пункту.

Меню має такі пункти:

1) Enter new equipment

2) Delete equipment

3) Info about equipment

4) Dump everything from the database

5) Change info

6) Search for an item by serial number

7) Sort by price

8) Count price for all equipment

9) Exit

Для того щоб пункти меню виконувалися відповідно до введеної цифри я використовував оператор «switch». Оператор switch у даній ситуації зручніше оператора If бо займає менше місця.

switch (choice)

{

case 1:

cin.ignore();

addNew(EquipmentVector);

break;

case 2:

int num0;

cout << "Enter number to delete: ";

cin >> num0;

cin.ignore();

remove(EquipmentVector, num0 - 1);

break;

case 3:

int num1;

cout << "Enter number in base: ";

cin >> num1;

cin.ignore();

getInfo(EquipmentVector, num1 - 1);

break;

case 4:

outputAll(EquipmentVector);

break;

case 5:

int num2;

cout << "Enter number of item you want to change: ";

cin >> num2;

cin.ignore();

edit(EquipmentVector, num2 - 1);

break;

case 6:

search(EquipmentVector);

break;

case 7:

sorting(EquipmentVector);

break;

case 8:

priceForAll(EquipmentVector);

break;

case 9:

break;

default:

cout << "Wrong number" << endl;

}

Для того, щоб меню виводилося знову після кожної дії, код меню був поміщений у цикл «while». Принцип роботи даного циклу, у тому щоб виконувати тіло циклу (код який має виконувати цикл), доки користувач не вибере функцію виходу. У коді програми, цикл виконується, доки виконується умова: змінна «choice» не дорівнює дев'ять. У коді використовуються оператор «!=», що означає «не дорівнює».

## Структура «AccountingForEquipmentComposition»

Як вказано у вступі, у програмі використовується структура. Структура була створена для того, щоб до начальних предметів була можливість вводити атрибути. Згідно з варіантом були вказані наступні атрибути структури:

Equipment - назва обладнання. NumberOfEquipment - кількість обладнання

SerialNumber - серійні номери.

Price - ціна за одиницю.

Створення структури у коді програми:

struct AccountingForEquipmentComposition

{

char Equipment[30];

int NumberOfEquipment;

int SerialNumber;

int Price;

};

Перед ім’ям кожного атрибута було вказано тип даних, який буде мати цей атрибут.

«char» - для символьних значень.

«int» - для цілочисленних значень.

У скобках «[ ]» вказується максимальний розмір значення який може зберігати даний атрибут.

Функції програми

У СУБД було використано функції для більш комфортної організації коду, та його читання. Для кожного пункту меню СУБД є відповідна функція, яка виконується коли користувач вибирає дію. Кожна функція знаходиться в окремому файлі і викликається з головного файлу.

Функція «addNew»

Ця функція дозволяє користувачеві вносити нові дані в базу даних.

Алгоритм роботи функції

Користувач по-черзі вводить значення кожного атрибуту, після чого робиться запис нової структурної змінної у бінарний файл та у вектор. Після цього бінарний файл закривається.Тепер якщо користувач зробить виведення таблиці на екран, він зможе побачити нову внесену інформацію.

void addNew(vector<AccountingForEquipmentComposition >& EquipmentVector)

{

AccountingForEquipmentComposition EquipmentElement;

char Equipment[30];

cout << "Equipment: ";

cin.getline(EquipmentElement.Equipment, sizeof(Equipment));

while (EquipmentElement.Equipment == "")

{

cout << "You have not entered anything";

cin.getline(EquipmentElement.Equipment, sizeof(Equipment));

}

cout << "Serial number: ";

cin >> EquipmentElement.SerialNumber;

cin.ignore();

cout << "Price: ";

cin >> EquipmentElement.Price;

cin.ignore();

cout << "Number of equipment: ";

cin >> EquipmentElement.NumberOfEquipment;

cin.ignore();

EquipmentVector.push\_back(EquipmentElement);

ofstream file("Equipment List.txt", ios::binary | ios::out | ios::app);

if (!file)

cout << "File does not open";

else

{

file.write((char\*)&EquipmentElement, sizeof(AccountingForEquipmentComposition));

}

file.close();

}

Функція «outputAll»

Функція виведення таблиці на екран дозволяє користувачеві побачити усі зроблені записи які зберігаються у бінарному файлі.

Алгоритм роботи функції

Функція відкриває файл і читає з нього елементи по порядку. Виводяться елементи по атрибутам, тобто за запис мається на увазі 4 атрибути: курс, назва предмету, група, викладач. Ці чотири атрибути розташовуються у строку та складають один запис.

void outputAll(vector<AccountingForEquipmentComposition >& EquipmentVector)

{

ifstream file("Equipment List.txt", ios::binary | ios::out | ios::app);

if (!file)

cout << "File does not open";

else

{

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

{

AccountingForEquipmentComposition EquipmentElement;

file.seekg(i \* sizeof(AccountingForEquipmentComposition));

file.read((char\*)&EquipmentElement, sizeof(AccountingForEquipmentComposition));

cout << "Item number " << i + 1 << endl;

cout << "Equipment: " << EquipmentVector[i].Equipment << "\t";

cout << "SerialNumber: " << EquipmentVector[i].SerialNumber << "\t";

cout << "Price: " << EquipmentVector[i].Price << "\t";

cout << "NumberOfEquipment: " << EquipmentVector[i].NumberOfEquipment << endl;

}

}

file.close();

Функція «search»

Функція пошуку предметів по викладачу дозволяє користувачу знайти предмети, які викладаються заданим викладачем. У аргументах вказано покажчик на вектор.

Алгоритм роботи функції

Функція запитує Користувач серійний номер обладнання. Коли функція отримує серійний номер, вона порівнює атрибут «SerialNumber» кожного запису у векторі із заданим, та, якщо значення атрибуту співпадає, виводить значення атрибуту «Equipment» цього запису.

void search(vector<AccountingForEquipmentComposition>& EquipmentVector)

{

int num;

bool found = false;

cout << "Enter serial number: ";

cin >> num;

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

{

if (num == EquipmentVector[i].SerialNumber)

{

found = true;

break;

}

}

if (found == true)

{

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

{

if (num == EquipmentVector[i].SerialNumber)

{

cout << endl << "Equipment: " << EquipmentVector[i].Equipment << endl;

cout << "Serial number: " << EquipmentVector[i].SerialNumber << endl;

}

}

}

else cout << "Wrong serial number" << endl;

}

Функція «getInfo»

Функція дозволяє знайти елемент по його номеру в базі даних.

Алгоритм роботи функції

Функція запитує у користувача номер елемента який він хоче знайти. Коли функція отримує значення , то виводиться значення всіх атрибутів цього запису.

void getInfo(vector<AccountingForEquipmentComposition>& EquipmentVector, int num)

{

ifstream file("Equipment List.txt", ios::binary | ios::out | ios::app);

if (!file)

cout << "File does not open";

else

{

if (num >= 0 && num < EquipmentVector.size())

{

AccountingForEquipmentComposition EquipmentElement;

file.seekg(num \* sizeof(AccountingForEquipmentComposition));

file.read((char\*)&EquipmentElement, sizeof(AccountingForEquipmentComposition));

cout << "Equipment: " << EquipmentVector[num].Equipment << endl;

cout << "SerialNumber: " << EquipmentVector[num].SerialNumber << endl;

cout << "Price: " << EquipmentVector[num].Price << endl;

cout << "NumberOfEquipment: " << EquipmentVector[num].NumberOfEquipment << endl;

}

else cout << "Wrong number\n";

}

Функція «edit»

Функція редагування рядка дозволяє користувачу вносити зміни до існуючих записів у базі даних.

Алгоритм роботи функції

У користувача запитується номер рядка, який він хоче редагувати. Коли користувач вводить номер рядка, функція робить заклик нових даних та перезаписує вибраний рядок. Після завершення редагування користувачу виводиться змінений рядок.

void edit(vector<AccountingForEquipmentComposition >& EquipmentVector, int num)

{

if (num >= 0 && num < EquipmentVector.size())

{

cout << "Before change: \n";

getInfo(EquipmentVector, num);

AccountingForEquipmentComposition EquipmentElement;

char Equipment[30];

cout << endl << "Equipment: ";

cin.getline(EquipmentElement.Equipment, sizeof(Equipment));

while (EquipmentElement.Equipment == "")

{

cout << "You have not entered anything";

cin.getline(EquipmentElement.Equipment, sizeof(Equipment));

}

cout << "Serial number: ";

cin >> EquipmentElement.SerialNumber;

cin.ignore();

cout << "Price: ";

cin >> EquipmentElement.Price;

cin.ignore();

cout << "Number of equipment: ";

cin >> EquipmentElement.NumberOfEquipment;

cin.ignore();

auto iterator = EquipmentVector.begin();

\*(iterator + num) = EquipmentElement;

ofstream file("Equipment List.txt", ios::binary | ios::out | ios::app);

if (!file) //Проверка открыт ли файл

cout << "File does not open";

else {

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

{

file.write((char\*)&EquipmentVector[i], sizeof(AccountingForEquipmentComposition));

}

}

file.close();

cout << endl << "After change: \n";

getInfo(EquipmentVector, num);

}

else cout << "Wrong number\n";

}

Функція «remove»

Функція дозволяє користувачеві видаляти дані з бази даних.

Алгоритм роботи функції

У користувача запитується номер рядка, який він хоче видалити. Коли користувач вводить номер рядка, функція видаляє елемент з вектора і перезаписує вектор в файл.

void remove(vector<AccountingForEquipmentComposition >& EquipmentVector, int num)

{

if (num >= 0 && num < EquipmentVector.size())

{

auto iterator = EquipmentVector.begin();

EquipmentVector.erase(iterator + num);

ofstream file("Equipment List.txt", ios::binary | ios::out | ios::app);

if (!file)

cout << "File does not open";

else

{

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

{

file.write((char\*)&EquipmentVector[i], sizeof(AccountingForEquipmentComposition));

}

}

file.close();

cout << "Item successfully deleted\n";

}

else cout << "Dont have so text\n";

}

Функція «priceForAll»

Функція виводить загальну ціну кожного товару.

Алгоритм роботи функції

Користувачеві виводяться серійний номер і загальна ціна кожного товару.

void priceForAll(vector<AccountingForEquipmentComposition>& EquipmentVector) //Функция считает цену всего оборудования

{

int price = 0;

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

{

price = EquipmentVector[i].Price \* EquipmentVector[i].NumberOfEquipment;

cout << "SerialNumber: " << EquipmentVector[i].SerialNumber << "\tPrice for all equipment: " << price << endl;

}

}

# ВИСНОВОК

Було розроблено СУБД, яка дозволяє заносити дані, змінювати та зберігати їх, маніпулювати над ними. Під час розробки курсового проекту було отримано навички у використанні можливостей коду на мові С++. Було отримано досвід у комфортному оформленні коду та його опису, а також написанні документації до нього. Було вивчено та опрацьовано багато теоретичного матеріалу стосовно кодування на С++. Створена програма демонструє роботу типової СУБД які використовуються повсюдно.

# **ДОДАТОК Б**

**Повний код програми курсової роботи**

#include <iostream>

#include <iomanip>

#include <fstream>

#include <vector>

#include <windows.h>

using namespace std;

const int NotUsed = system("color F0");

struct Menu

{

int number;

const char\* text;

};

struct AccountingForEquipmentComposition

{

char Equipment[30];

int NumberOfEquipment;

int SerialNumber;

int Price;

};

Menu menu[] =

{

{ 1, "Enter new equipment" },

{ 2, "Delete equipment" },

{ 3, "Info about equipment" },

{ 4, "Dump everything from the database" },

{ 5, "Change info" },

{ 6, "Search for an item by serial number" },

{ 7, "Sort by price" },

{ 8, "Count price for all equipment" },

{ 9, "Exit"}

};

int showMenu(const Menu\* menu1)

{

for (int i = 0; i < 9; i++)

{

cout << menu1[i].number << ") " << menu1[i].text << endl;

}

int choice;

cout << "Make your choose: ";

cin >> choice;

return choice;

}

void addNew(vector<AccountingForEquipmentComposition >& EquipmentVector)

{

AccountingForEquipmentComposition EquipmentElement;

char Equipment[30];

int NumberOfEquipment;

int SerialNumber;

int Price;

cout << "Equipment: ";

cin.getline(EquipmentElement.Equipment, sizeof(Equipment));

while (EquipmentElement.Equipment == "")

{

cout << "You have not entered anything";

cin.getline(EquipmentElement.Equipment, sizeof(Equipment));

}

cout << "Serial number: ";

cin >> EquipmentElement.SerialNumber;

cin.ignore();

cout << "Price: ";

cin >> EquipmentElement.Price;

cin.ignore();

cout << "Number of equipment: ";

cin >> EquipmentElement.NumberOfEquipment;

cin.ignore();

EquipmentVector.push\_back(EquipmentElement);

ofstream file("Equipment List.txt", ios::binary | ios::out | ios::app);

if (!file)

cout << "File does not open";

else

{

file.write((char\*)&EquipmentElement, sizeof(AccountingForEquipmentComposition));

}

file.close();

}

void getInfo(vector<AccountingForEquipmentComposition>&EquipmentVector, int num)

{

ifstream file("Equipment List.txt", ios::binary | ios::out | ios::app);

if (!file)

cout << "File does not open";

else

{

if (num >= 0 && num < EquipmentVector.size())

{

AccountingForEquipmentComposition EquipmentElement;

file.seekg(num \* sizeof(AccountingForEquipmentComposition) );

file.read((char\*)&EquipmentElement, sizeof(AccountingForEquipmentComposition));

cout << "Equipment: " << EquipmentVector[num].Equipment << endl;

cout << "SerialNumber: " << EquipmentVector[num].SerialNumber << endl;

cout << "Price: " << EquipmentVector[num].Price << endl;

cout << "NumberOfEquipment: " << EquipmentVector[num].NumberOfEquipment << endl;

}

else cout << "Wrong number\n";

}

file.close();

}

void outputAll(vector<AccountingForEquipmentComposition >& EquipmentVector)

{

ifstream file("Equipment List.txt", ios::binary | ios::out | ios::app);

if (!file)

cout << "File does not open";

else

{

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

{

AccountingForEquipmentComposition EquipmentElement;

file.seekg(i \* sizeof(AccountingForEquipmentComposition));

file.read((char\*)&EquipmentElement, sizeof(AccountingForEquipmentComposition));

cout << "Item number " << i+1 << endl;

cout << "Equipment: " << EquipmentVector[i].Equipment << "\t";

cout << "SerialNumber: " << EquipmentVector[i].SerialNumber << "\t";

cout << "Price: " << EquipmentVector[i].Price << "\t";

cout << "NumberOfEquipment: " << EquipmentVector[i].NumberOfEquipment << endl;

}

}

file.close();

}

void remove(vector<AccountingForEquipmentComposition >& EquipmentVector, int num)

{

if (num >= 0 && num < EquipmentVector.size())

{

auto iterator = EquipmentVector.begin();

EquipmentVector.erase(iterator + num);

ofstream file("Equipment List.txt", ios::binary | ios::out | ios::app);

if (!file)

cout << "File does not open";

else {

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

{

file.write((char\*)&EquipmentVector[i], sizeof(AccountingForEquipmentComposition));

}

}

file.close();

cout << "Item successfully deleted\n";

}

else cout << "Dont have so text\n";

}

void edit(vector<AccountingForEquipmentComposition >& EquipmentVector, int num)

{

if (num >= 0 && num < EquipmentVector.size())

{

cout << "Before change: \n";

getInfo(EquipmentVector, num);

AccountingForEquipmentComposition EquipmentElement;

char Equipment[30];

int NumberOfEquipment;

int SerialNumber;

int Price;

cout << endl << "Equipment: ";

cin.getline(EquipmentElement.Equipment, sizeof(Equipment));

while (EquipmentElement.Equipment == "")

{

cout << "You have not entered anything";

cin.getline(EquipmentElement.Equipment, sizeof(Equipment));

}

cout << "Serial number: ";

cin >> EquipmentElement.SerialNumber;

cin.ignore();

cout << "Price: ";

cin >> EquipmentElement.Price;

cin.ignore();

cout << "Number of equipment: ";

cin >> EquipmentElement.NumberOfEquipment;

cin.ignore();

auto iterator = EquipmentVector.begin();

\*(iterator + num) = EquipmentElement;

ofstream file("Equipment List.txt", ios::binary | ios::out | ios::app);

if (!file)

cout << "File does not open";

else {

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

{

file.write((char\*)&EquipmentVector[i], sizeof(AccountingForEquipmentComposition));

}

}

file.close();

cout << endl << "After change: \n";

getInfo(EquipmentVector, num);

}

else cout << "Wrong number\n";

}

void search(vector<AccountingForEquipmentComposition>&EquipmentVector)

{

int num;

bool found = false;

cout << "Enter serial number: ";

cin >> num;

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

{

if (num == EquipmentVector[i].SerialNumber)

{

found = true;

break;

}

}

if (found == true)

{

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

{

if (num == EquipmentVector[i].SerialNumber)

{

cout << endl << "Equipment: " << EquipmentVector[i].Equipment << endl;

cout << "Serial number: " << EquipmentVector[i].SerialNumber << endl;

}

}

}

else cout << "Any equipment with so serial number" << endl;

}

void sorting(vector<AccountingForEquipmentComposition>&EquipmentVector)

{

for (int i = 0; i < EquipmentVector.size() - 1; i++)

{

for (int j = 0; j < EquipmentVector.size() - i - 1; j++)

{

if (EquipmentVector[j].Price < EquipmentVector[j + 1].Price)

{

swap(EquipmentVector[j], EquipmentVector[j + 1]);

}

}

}

outputAll(EquipmentVector);

}

void priceForAll(vector<AccountingForEquipmentComposition>&EquipmentVector)

{

int price = 0;

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

{

price = EquipmentVector[i].Price \* EquipmentVector[i].NumberOfEquipment;

cout << "SerialNumber: " << EquipmentVector[i].SerialNumber << "\tPrice for all equipment: " << price << endl;

}

}

int main()

{

fstream file("Equipment List.txt", ios::binary | ios::out | ios::app);

streampos begin, end;

begin = file.tellg();

file.seekg(0, ios::end);

end = file.tellg();

int size = (end - begin) / sizeof(AccountingForEquipmentComposition);

file.close();

vector<AccountingForEquipmentComposition>EquipmentVector;

AccountingForEquipmentComposition EquipmentElement;

for (int i = 0; i < size; i++) EquipmentVector.push\_back(EquipmentElement);

file.open("Equipment List.txt", ios::binary | ios::in | ios::app);

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

{

file.read((char\*)&EquipmentVector[i], sizeof(AccountingForEquipmentComposition));

}

int choice;

do

{

cout << endl;

choice = showMenu(menu);

if (EquipmentVector.size() == 0)

{

switch (choice)

{

case 1:

cin.ignore();

addNew(EquipmentVector);

break;

case 9:

break;

default:

cout << "Anything in base" << endl;

}

}

else {

switch (choice)

{

case 1:

cin.ignore();

addNew(EquipmentVector);

break;

case 2:

int num0;

cout << "Enter number to delete: ";

cin >> num0;

cin.ignore();

remove(EquipmentVector, num0 - 1);

break;

case 3:

int num1;

cout << "Enter number in base: ";

cin >> num1;

cin.ignore();

getInfo(EquipmentVector, num1 - 1);

break;

case 4:

outputAll(EquipmentVector);

break;

case 5:

int num2;

cout << "Enter number of item you want to change: ";

cin >> num2;

cin.ignore();

edit(EquipmentVector, num2 - 1);

break;

case 6:

search(EquipmentVector);

break;

case 7:

sorting(EquipmentVector);

break;

case 8:

priceForAll(EquipmentVector);

break;

case 9:

break;

default:

cout << "Wrong number" << endl;

}

}

} while (choice != 9);

}