****

### Kierunek: Informatyka

### Imię: Mariusz

### Nazwisko: Perzyński

### Grupa: I10\_L

Sprawozdanie z laboratorium do przedmiotu Podstawy Programowania

# Lista nr 1

## Zadanie 1

### 

### 1.1 Opis rozwiązania (oraz schemat blokowy):

Baza studentów. Na początku zdefiniowane klucz-klawisze do poszczególnych do użycia w pętli wyboru. Struktura osoby jako pojedynczego ucznia(studenta) i jego cech. Funkcje dodajace studentow na poczatku dzialania programu oraz pozniejszy wybor. Wybór : dodanie nowego, usuniecie jezeli ma srednia 2.5 lub mniej, zapis aktualnego stanu czyli inaczej zapis do pliku bez tego wszystkie operacje robione sa ,,na sucho,, , oraz sortowanie studentow alfabetycznie. Na koncu wyjscia zapytanie o to czy wychodzi, czy przed tym chce zobaczyc swoich studentow w pliku textowym, nastepnie nastepuje wyjscie i stosowany komunikat.

### 1.2 Kod źródłowy:

#include <iostream>

#include <string>

#include <stdio.h>

#include <vector>

#include <conio.h>

#include <algorithm>

#include <cctype>

#include <fstream>

#include <iterator> // wskaznik na okreslony element vectora

using namespace std;

#define KEY\_L 76 //- alfabetyczny porzadek

#define KEY\_K 75 //- zapis studentow i pliku

#define KEY\_U 85 //- usuniecie studenta z 2.5 srocen

#define KEY\_D 68 //- dodanie nowego studenta(im,nazw,wiek,sro,nralb)

#define KEY\_X 88 //- wyjscie

struct Osoba{

string imie;

string nazwisko;

int nrAlbumu;

float srOcen;

int wiek;

bool operator < (const Osoba &x)const

{

return imie>x.imie;

return nazwisko>x.nazwisko;

}

};

void ograniczenie(int &liczbaMax){

cout << "Ilu studentow chcesz dodac(nie wiecej niz 20)?" << endl;

cin >> liczbaMax;

if(liczbaMax > 20 ){

cout << endl;

cout << "nie moze byc wieksza niz 20!" << endl;

cout << "popraw sie: " << endl;

cin >> liczbaMax;

}

}

void menu(){

cout << "---------------------------------------" << endl;

cout << "---- U - USUWA STUDENTA ZE SREDNIA <= 2.5" << endl;

cout << "---- L - LISTUJE NAZWISKA I SORTUJE" << endl;

cout << "---- K - ZAPISUJE AKTUALNY STAN I WYCHODZI Z PROGRAMU" << endl;

cout << "---- D - DODAJE NOWEGO STUDENTA" << endl;

cout << "---- X - WYCHODZI CALKOWICIE Z PETLI" << endl;

cout << "---- NP. Q - PRZECHODZI DO WYJSCIA" << endl;

cout << "---------------------------------------" << endl;

}

void dodawanieAll(int liczbaMax, vector<Osoba>&);

void usuwanie(int liczbaMax, vector<Osoba>&);

void dodawanieNowego(int liczbaMax,int &liczbaMaxNowych, vector<Osoba>&);

void sortowanieStudentow(int liczbaMax,int liczbaMaxNowych, vector<Osoba>&);

void zapisDoPliku(int liczbaMax, int liczbaMaxNowych, vector<Osoba>&);

void odczytPliku(int liczbaMax, int liczbaMaxNowych, vector<Osoba>&);

int main(){

fstream plikBazy;

vector<Osoba> studenci(20);

int liczbaMax;

int liczbaMaxNowych;

int wybor;

ograniczenie(liczbaMax);

dodawanieAll(liczbaMax, studenci);

menu(); // bez paramaetrowy void z prostym menu

char key = getch();

int value = key;

int nowe;

while(value != KEY\_X){ // x - wyjscie z petli calkowicie

switch(getch()){

case KEY\_L : //listowanie + duza litera calych imion i nazwisk

sortowanieStudentow(liczbaMax, liczbaMaxNowych, studenci);

break;

case KEY\_D : //dodawanie ucznia > 20 uczniow w grupie/ na liscie

dodawanieNowego(liczbaMax,liczbaMaxNowych, studenci);

break;

case KEY\_U : //usuniecie studenta ze srocen 2.5 lub mniej

usuwanie(liczbaMax, studenci);

break;

case KEY\_K : //zapis aktualnego stanu - do pliku + zakonczenie programu

zapisDoPliku(liczbaMax,liczbaMaxNowych,studenci);

break;

default: //wyjscie

cout << "pamietaj o zapisie do pliku klucz kod - K , bo utracisz wszystkie zapisy" << endl;

cout << "wychodzisz czy chcesz zobaczyc swoich studentow? [ 1 - (studenci) / 0 - (wyjscie) ]" << endl;

cin >> wybor;

// odczyt z pliku

if(wybor == 0){

cout << "zakonczyles program, milego dnia..." << endl;

return 0;

} else {

odczytPliku(liczbaMax,liczbaMaxNowych,studenci);

}

}

}

key = getch();

value = key;

return 0;

}

void dodawanieAll(int liczbaMax, vector<Osoba>& studenci){

int i;

for(i = 1; i < liczbaMax + 1; i++){

cout << "podaj imie " << i << " studenta:"<< endl;

cin >> studenci[i].imie;

cout << "podaj nazwisko " << i << " studenta:"<< endl;

cin >> studenci[i].nazwisko;

cout << "podaj numer albumu " << i << " studenta:"<< endl;

cin >> studenci[i].nrAlbumu;

cout << "podaj srednia ocen " << i << " studenta:"<< endl;

cin >> studenci[i].srOcen;

cout << "podaj wiek " << i << " studenta:"<< endl;

cin >> studenci[i].wiek;

}

cout << "Twoi studenci: " << endl;

cout << endl;

for(int i = 1; i < liczbaMax + 1; i++){

cout <<"Studenci: "<< studenci[i].imie <<" "<< studenci[i].nazwisko <<" ma lat "<< studenci[i].wiek <<" numer albumu: "<< studenci[i].nrAlbumu <<" oraz srednia ocen to: "<< studenci[i].srOcen << endl;

}

}

void usuwanie(int liczbaMax, vector<Osoba>& studenci){

int i;

for(int i = 1; i < liczbaMax + 1; i++){

if(studenci[i].srOcen <= 2.5){

cout << "usuniesz studenta ze srednia mniejsza lub rowna 2.5" << endl;

cout << endl;

cout <<"Student: "<< studenci[i].imie <<" "<< studenci[i].nazwisko <<" ma lat "<< studenci[i].wiek <<" numer albumu: "<< studenci[i].nrAlbumu <<" oraz srednia ocen to: "<< studenci[i].srOcen << endl;

cout << endl;

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

cout << "student zostal usuniety" << endl;

} else {

cout << "nie znaleziono studenta ze srednia ocen mniejsza lub rowna 2.5" << endl;

}

}

return;

}

void dodawanieNowego(int liczbaMax,int &liczbaMaxNowych, vector<Osoba>& studenci){

cout << "Ilu nowych studentow chcesz dodac(nie wiecej niz 20)?" << endl;

cin >> liczbaMaxNowych;

if(liczbaMaxNowych > 20 ){

cout << endl;

cout << "nie moze byc wieksza niz 20!" << endl;

cout << "popraw sie: " << endl;

cin >> liczbaMaxNowych;

}

int i;

for(i = 1 + liczbaMax; i < liczbaMaxNowych + 1 + liczbaMax; i++){

cout << "podaj imie " << i - liczbaMax << " nowego studenta:"<< endl;

cin >> studenci[i].imie;

cout << "podaj nazwisko " << i - liczbaMax << " nowego studenta:"<< endl;

cin >> studenci[i].nazwisko;

cout << "podaj numer albumu " << i - liczbaMax << " nowego studenta:"<< endl;

cin >> studenci[i].nrAlbumu;

cout << "podaj srednia ocen " << i - liczbaMax << " nowego studenta:"<< endl;

cin >> studenci[i].srOcen;

cout << "podaj wiek " << i - liczbaMax << " nowego studenta:"<< endl;

cin >> studenci[i].wiek;

}

cout << "nowy student / studenci zostal / zostali dodani do bazy studentow!" << endl;

cout << "wrociles do petli - menu, po zakonczeniu tego switcha" << endl;

return;

}

void sortowanieStudentow(int liczbaMax, int liczbaMaxNowych, vector<Osoba>& studenci){

cout << "Sortowanie studentow!" << endl;

for(int i = 1; i < liczbaMax + liczbaMaxNowych + 1; i++){

cout <<"Studenci przed sortowaniem: "<< studenci[i].imie <<" "<< studenci[i].nazwisko <<" ma lat "<< studenci[i].wiek <<" numer albumu: "<< studenci[i].nrAlbumu <<" oraz srednia ocen to: "<< studenci[i].srOcen << endl;

}

sort(studenci.begin(),studenci.end());

cout << endl;

for(int i = 1; i < liczbaMax + liczbaMaxNowych + 1; i++){

cout <<"Studenci po sortowaniu: "<< studenci[i].imie <<" "<< studenci[i].nazwisko <<" ma lat "<< studenci[i].wiek <<" numer albumu: "<< studenci[i].nrAlbumu <<" oraz srednia ocen to: "<< studenci[i].srOcen << endl;

}

cout << endl;

return;

}

void zapisDoPliku(int liczbaMax, int liczbaMaxNowych, vector<Osoba>& studenci){

ofstream plikBazy;

plikBazy.open( "base.txt", ios::out | ios::app);

for(int i = 1; i < liczbaMax + liczbaMaxNowych + 1; i++){

plikBazy << studenci[i].imie <<" "<< studenci[i].nazwisko <<" ma lat "<< studenci[i].wiek <<" numer albumu: "<< studenci[i].nrAlbumu <<" oraz srednia ocen to: "<< studenci[i].srOcen << endl;

}

plikBazy.close();

cout << "zapisano!" << endl;

return;

}

void odczytPliku(int liczbaMax, int liczbaMaxNowych, vector<Osoba>& studenci){

ifstream plikBazy;

plikBazy.open( "base.txt", ios::in | ios::app);

if(plikBazy.is\_open())

{

cout << "Aktualna lista studentow wyglada nastepujaco: " << endl;

cout << endl;

char wiersz[1000];

while(plikBazy.getline(wiersz,1000))

{

cout<< wiersz << endl;

}

cout << endl;

}

plikBazy.close();

return;

}