МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ

“БРЕСТСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ”

**КАФЕДРА ИИТ**

ОТЧЁТ

по лабораторной работе №10

**«Структуры, перечисления, объединения»**

Выполнил:

студент 1 курса

группы ПО-9

Харитонович Захар Сергеевич

Проверила:

Хацкевич М. В.

Брест 2022

**Цель работы:** Изучить синтаксис и правила работы со структурами. Реализовать программу с применением структур, перечислений и объединений.

**Порядок выполнения работы**

**Текст задания**

Создать тип структуры согласно варинту, организовать поля этой структуры так, чтобы они содержали объединение, перечисление (можно добавить дополнительные поля) и битовое поле.

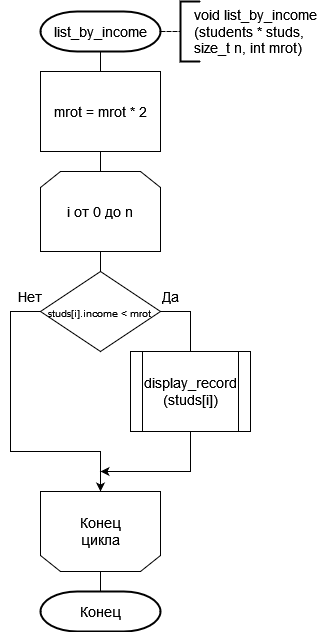
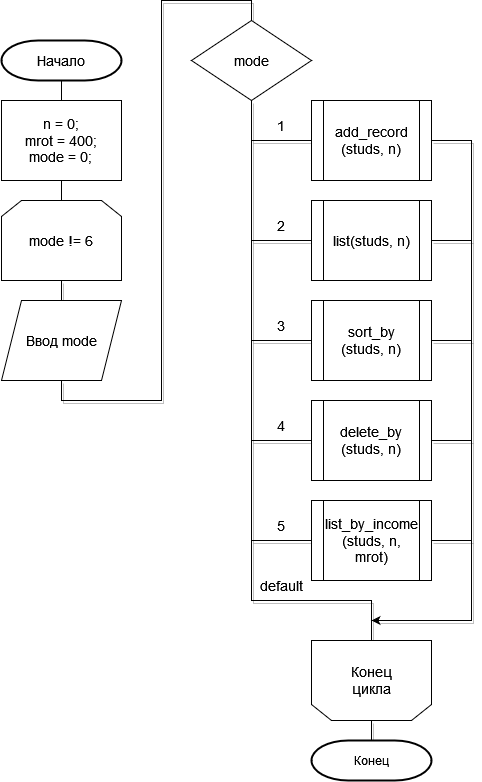
Создать массив структур, содержащий информацию согласно варианту индивидуального задания.

Реализовать работу с массивом структур через меню: ввод данных в массив, вывод собержимого массива на экран, сортировка по одному полю, удаления записи по заданному значению поля, выборка записей согласно индивидуального задания.

*Вариант 3*

Для получения места в общежитии формируется список студентов, который включает ФИО студента, номер группы, средний балл, доход на члена семьи. Вывести фамилии студентов, у которых доход на члена семьи меньше двух минимальных зарплат.

**Блок-схема**



#include <iostream>

#include <stdlib.h>

#include <string.h>

using namespace std;

struct students {

char surname[256];

char name[256];

char patronymic[256];

enum groups {po8, po9, ii21, ii22, unknown} grp;

double avrMark;

int income;

short bit : 4;

union {

int integer;

double real;

};

};

void display\_record(students);

void add\_record(students \*&, size\_t &);

void list(students \*, size\_t);

void sort\_by(students \*, size\_t);

void delete\_by(students \*&, size\_t &);

void list\_by\_income(students \*, size\_t, int);

int main() {

students \* studs = new students[1];

size\_t n = 0;

int mrot = 400;

cout << "Welcome to students database.\n";

short mode = 0;

while (mode != 6) {

cout << "Choose mode:\n1 - add a record. 2 - list records. 3 - sort by.\n4 - delete by. 5 - list records with small income. 6 - exit.\n> ";

cin >> mode;

switch (mode) {

case 1:

add\_record(studs, n);

break;

case 2:

list(studs, n);

break;

case 3:

sort\_by(studs, n);

break;

case 4:

delete\_by(studs, n);

break;

case 5:

list\_by\_income(studs, n, mrot);

case 6:

cout << "Exit\n";

break;

default:

cout << "Unknown.\n";

break;

}

}

delete[] studs;

return 0;

}

void add\_record(students \*&studs, size\_t &n) {

students \* temp = new students[n+1];

for (size\_t i = 0; i < n; i++) temp[i] = studs[i];

delete[] studs;

studs = temp;

cout << "Add a record. Fields:\nName. Surname. Patronymic. Group. Average mark. Income per family member.\n";

char group[256];

cin >> studs[n].surname >> studs[n].name >> studs[n].patronymic >> group >> studs[n].avrMark >> studs[n].income;

if (!strcmp(group, "po-8") || !strcmp(group, "PO-8")) {

studs[n].grp = students::po8;

} else if (!strcmp(group, "po-9") || !strcmp(group, "PO-9")) {

studs[n].grp = students::po9;

} else if (!strcmp(group, "ii-21") || !strcmp(group, "II-21")) {

studs[n].grp = students::ii21;

} else if (!strcmp(group, "ii-22") || !strcmp(group, "II-22")) {

studs[n].grp = students::ii22;

} else {

studs[n].grp = students::unknown;

}

++n;

void display\_record(students cur) {

cout << cur.surname << " " << cur.name << " " << cur.patronymic;

switch (cur.grp) {

case students::po8:

cout << " PO-8 ";

break;

case students::po9:

cout << " PO-9 ";

break;

case students::ii21:

cout << " II-21 ";

break;

case students::ii22:

cout << " II-22 ";

break;

default:

cout << " Unknown group ";

break;

}

cout << cur.avrMark << " " << cur.income << endl;

void list(students \*studs, size\_t n) {

cout << "Students list. Fields:\nName. Surname. Patronymic. Group. Average mark. Income per family member.\n";

for (size\_t i = 0; i < n; i++) {

cout << i + 1 << ") "; display\_record(studs[i]);

}

}

void sort\_by(students \* studs, size\_t n) {

cout << "Sort by:\n1 - surname. 2 - group. 3 - mark. 4 - income.\n> ";

short mode; cin >> mode;

switch (mode) {

case 1:

for(size\_t i = 1; i < n; i++)

for (size\_t j = i; j > 0 && strcmp(studs[j].surname, studs[j-1].surname) < 0; j--) swap(studs[j], studs[j-1]);

break;

case 2:

for(size\_t i = 1; i < n; i++)

for (size\_t j = i; j > 0 && studs[j-1].grp > studs[j].grp; j--)

swap(studs[j], studs[j-1]);

break;

case 3:

for(size\_t i = 1; i < n; i++)

for (size\_t j = i; j > 0 && studs[j-1].avrMark < studs[j].avrMark; j--) swap(studs[j], studs[j-1]);

break;

case 4:

for(size\_t i = 1; i < n; i++) {

for (size\_t j = i; j > 0 && studs[j-1].income > studs[j].income; j--) swap(studs[j], studs[j-1]);

break;

default:

cout << "Unknown.\n";

break;

}

}

void delete\_by(students \*&studs, size\_t &n) {

cout << "Delete by:\n1 - surname. 2 - group. 3 - mark. 4 - income.\n> ";

short mode; cin >> mode;

switch (mode) {

case 1:

cout << "Surname: ";

char toDel[256]; cin >> toDel;

for (size\_t i = 0; i < n; i++)

if (!strcmp(studs[i].surname, toDel)) {

n--;

for (size\_t j = i; j < n; j++) studs[j] = studs[j+1];

}

break;

case 2:

cout << "Group: ";

char group[256]; cin >> group;

short tempGrp;

if (!strcmp(group, "po-8") || !strcmp(group, "PO-8")) {

tempGrp = students::po8;

} else if (!strcmp(group, "po-9") || !strcmp(group, "PO-9")) {

tempGrp = students::po9;

} else if (!strcmp(group, "ii-21") || !strcmp(group, "II-21")) {

tempGrp = students::ii21;

} else if (!strcmp(group, "ii-22") || !strcmp(group, "II-22")) {

tempGrp = students::ii22;

} else {

tempGrp = students::unknown;

}

for (size\_t i = 0; i < n; i++) {

if (studs[i].grp == tempGrp) {

n--;

for (size\_t j = i; j < n; j++) studs[j] = studs[j+1];

}

}

break;

case 3:

cout << "Average mark: ";

double mark; cin >> mark;

for (size\_t i = 0; i < n; i++)

if (studs[i].avrMark == mark) {

n--;

for (size\_t j = i; j < n; j++) studs[j] = studs[j+1];

}

break;

case 4:

cout << "Income: ";

int income; >> income;

for (size\_t i = 0; i < n; i++)

if (studs[i].income == income) {

n--;

for (size\_t j = i; j < n; j++) studs[j] = studs[j+1];

}

break;

default:

cout << "Unknown.\n";

return;

}

students \* temp = new students[n];

for(size\_t i = 0; i < n; i++) {

temp[i] = studs[i];

}

delete[] studs;

studs = temp;

}

void list\_by\_income(students \* studs, size\_t n, int mrot) {

mrot \*= 2;

size\_t j = 1;

for (size\_t i = 0; i < n; i++)

if (studs[i].income < mrot) {

cout << j << ") ";

display\_record(studs[i]);

j++;

}

}

**Вывод:** был изучен синтаксис и принципы работы со структурами. Согласно данному заданию была реализована программа с использованием структура, перечислений и объединений.