Учреждение образования

«БЕЛОРУССКИЙГОСУДАРСТВЕННЫЙТЕХНОЛОГИЧЕСКИЙУНИВЕРСИТЕТ»

Н. Н. Пустовалова, Н. В. Пацей

# ОСНОВЫ АЛГОРИТМИЗАЦИИ И ПРОГРАММИРОВАНИЯ

**в 2-х частях**

**Часть 2**

**Лабораторный практикум**

**Лабораторная работа № 4. Представление информации в виде структуры**

**Лабораторная работа № 4. Представление информации в виде структуры**

|  |  |
| --- | --- |
| **Задание** | **Программа** |
| 1. Изучить способы организации данных в виде *структуры*, выполнив программу, записанную в данном пункте.  Дополнить структуру дополнительными сведениями о работниках и вывести их вместе с фамилиями на экран. |  |
| 2. В правой части приведен пример программы, в которой используется *указатель на структуру*.  Выполнить программу, меняя скорость вывода информации.  Написать пояснения к программе. |  |
| 3. В правой части приведен пример программы, которая осуществляет работу с данными, организованными в виде структуры, содержащей информацию о студентах: №,Фамилия, Год рождения, Факультет.  Проанализировать работу главной функции и функций, входящих в программу.  Добавить операторы выдачи сообщений об ошибках при вводе данных неправильного типа | #include <iostream>  #include <windows.h>  #include <string>  #define str\_len 30  #define size 30  using namespace std;  void enter\_new();  void del();  void change();  void out();  struct Student  {  char name[str\_len];  int year\_of\_birth;  char department[5];  };  struct Student list\_of\_student[size];  struct Student bad;  int current\_size = 0;  int choice;  int main()  {  string choicecheck;  SetConsoleCP(1251);  SetConsoleOutputCP(1251);  setlocale(LC\_CTYPE, "Russian");  cout << "Введите:" << endl;  cout << "1-для удаления записи" << endl;  cout << "2-для ввода новой записи" << endl;  cout << "3-для изменения записи" << endl;  cout << "4-для вывода записи(ей)" << endl;  cout << "5-для выхода" << endl;  cin >> choicecheck;  while (choicecheck != "1" && choicecheck != "2" && choicecheck != "3" && choicecheck != "4" && choicecheck != "5")  {  cout << "Ошибка ввода, попробуйте повторить ввод или нажмите 0 для выхода из программы"<<endl;  cin >> choicecheck;  if (choicecheck == "0")  {  return 0;  }  }  choice = stoi(choicecheck);  do  {  switch (choice)  {  case 1: del(); break;  case 2: enter\_new(); break;  case 3: change(); break;  case 4: out(); break;  }  } while (choice != 5);  return 0;  }  void enter\_new()  {  string checkyear;  int count = 0;  cout << "Ввод информации" << endl;  if (current\_size < size)  {  cout << "Строка номер ";  cout << current\_size + 1;  cout << endl << "Фамилия: ";  cin >> list\_of\_student[current\_size].name;  cout << "Год рождения: ";  while (true)  {  cin >> checkyear;  for (int i = 0; i < checkyear.length(); i++)  {  if (checkyear[i] >= '0' && checkyear[i] <= '9')  {  count++;  }  }  if (count == checkyear.length())  {  list\_of\_student[current\_size].year\_of\_birth = stoi(checkyear);  break;  }  else  {  cout << "Ошибка ввода, попробуйте повторить ввод, или нажмите 0 для выхода их программы ";  }  }  cout << "Факультет: ";  cin >> list\_of\_student[current\_size].department;  current\_size++;  }  else  cout << "Введено максимальное кол-во строк";  cout << "Что дальше?" << endl;  cin >> choice;  }  void del()  {  int d;  cout << "\nНомер строки, которую надо удалить (для удаления всех строк нажать 99)" << endl; cin >> d;  if (d != 99)  {  for (int de1 = (d - 1); de1 < current\_size; de1++)  {  list\_of\_student[de1] = list\_of\_student[de1 + 1];  }  current\_size = current\_size - 1;  }  if (d == 99)  for (int i = 0; i < size; i++)  {  list\_of\_student[i] = bad;  }  cout << "Что дальше?" << endl;  cin >> choice;  }  void change()  {  int n, per;  cout << "\nВведите номер строки" << endl; cin >> n;  do  {  cout << "Введите: " << endl;  cout << "1-для изменения фамилии" << endl;  cout << "2-для изменения года рождения" << endl;  cout << "3-для изменения факультета" << endl;  cout << "4-конец\n";  cin >> per;  switch (per)  {  case 1: cout << "Новая фамилия";  cin >> list\_of\_student[n - 1].name; break;  case 2: cout << "Новый год рождения";  cin >> list\_of\_student[n - 1].year\_of\_birth; break;  case 3: cout << "Новый факультет ";  cin >> list\_of\_student[n - 1].department; break;  }  } while (per != 4);  cout << "Что дальше?" << endl;  cin >> choice;  }  void out()  {  int sw, n;  cout << "1-вывод 1 строки" << endl;  cout << "2-вывод всех строк" << endl;  cin >> sw;  if (sw == 1)  {  cout << "Номер выводимой строки " << endl; cin >> n; cout << endl;  cout << "Фамилия ";  cout << list\_of\_student[n - 1].name << endl;  cout << "Год рождения ";  cout << list\_of\_student[n - 1].year\_of\_birth << endl;  cout << "Факультет ";  cout << list\_of\_student[n - 1].department << endl;  }  if (sw == 2)  {  for (int i = 0; i < current\_size; i++)  {  cout << "Фамилия ";  cout << list\_of\_student[i].name << endl;  cout << "Год рождения ";  cout << list\_of\_student[i].year\_of\_birth << endl;  cout << "Факультет ";  cout << list\_of\_student[i].department << endl;  }  }  cout << "Что дальше?" << endl;  cin >> choice;  } |
| 4. Выполнить программму, приведенную в правой части, несколько раз с различными исходными данными.  Убедиться, что данные, введенные с клавиатуры, записываются в файл с именем **base.bin**.  Записать условие задачи и комментарии к программе. | #include <iostream>  #include <windows.h>  using namespace std;  void input(int size);  void output();  void find(char lastName[]);  typedef struct Students  {  char fio[16];  char group[3];  } STUD;  int number; FILE\* f; errno\_t err;  int main()  {  SetConsoleCP(1251);  SetConsoleOutputCP(1251);  setlocale(LC\_ALL, "Russian");  int choice; char fio[16];  do  {  cout << "\n1.Ввод данных с клавиатуры и запись в файл\n";  cout << "2.Вывод данных из файла\n";  cout << "3.Поиск по фамилии\n";  cout << "0.Выход из программы\n\n";  cout << "Введите номер операции: ";  cin >> choice;  switch (choice)  {  case 1: cout << "Введите количество студентов: ";  cin >> number;  input(number); break;  case 2: output(); break;  case 3: {  cout << "Введите фамилию: ";  cin >> fio;  find(fio); break;  }  case 0: exit(0); break;  }  } while (choice != 0);  }  void input(int size)  {  STUD buf = { ' ', ' ' };  if (!fopen\_s(&f, "base.bin", "ab"))  {  for (int p = 0; p < size; p++)  {  cout << "Фамилия: "; cin >> buf.fio;  cout << "Группа: "; cin >> buf.group;  fwrite(&buf, sizeof(buf), 1, f);  }  fclose(f);  }  else {  cout << "Ошибка открытия файла";  return;  }  }  void output()  {  STUD buf;  if (!fopen\_s(&f, "base.bin", "rb"))  {  cout << "\nФамилия Группа\n";  fread(&buf, sizeof(buf), 1, f);  while (!feof(f))  {  cout << buf.fio << "\t " << buf.group << endl;  fread(&buf, sizeof(buf), 1, f);  }  cout << endl;  fclose(f);  }  else  {  cout << "Ошибка открытия файла";  return;  }  }  void find(char lastName[16])  {  bool flag = false; STUD buf;  if (!fopen\_s(&f, "base.bin", "rb"))  {  while (!feof(f))  {  fread(&buf, sizeof(buf), 1, f);  if (strcmp(lastName, buf.fio) == 0) //сравнение строк  {  cout << "\nФамилия Группа\n";  cout << buf.fio << "\t " << buf.group << endl;  flag = true; break;  }  }  fclose(f);  if (!flag) cout << "Ничего не найдено\n";  }  else  {  cout << "Ошибка открытия файла";  return;  }  } |
| 5. В соответствии со своим вариантом разработать программу для данных, приведенных в таблице ниже. Определить структурированный тип, разработать меню для работы с ***массивом структур***.  В программу должны войти функции:   * ввод элементов структуры с клавиатуры; * вывод элементов структуры в консольное окно; * удаление заданной структурированной переменной; * поиск информации; * запись информации в файл; * чтение данных из файла.   Вар.1  **Горожанин.** Ф.И.О., дата рождения, адрес, пол (м, ж). Реализовать выборку по году рождения. |  |

#include <iostream>

#include <windows.h>

using namespace std;

FILE\* fdata;

typedef struct Human

{

int index;

char fio[100]; // Фамилия, имя, отчество

int year\_of\_birth; // Год рождения

char address[200]; // Адрес

char gender[2]; // Пол

} PERSON;

int get\_index()

{

PERSON buf;

int maxIndex = 0; // Хранит максимальный индекс найденной записи

if (!fopen\_s(&fdata, "data.bin", "rb")) // Открываем файл для чтения

{

while (fread(&buf, sizeof(PERSON), 1, fdata)) // Проходим по всем записям в файле

{

if (buf.index > maxIndex)

{

maxIndex = buf.index; // Обновляем максимальный индекс

}

}

fclose(fdata);

return maxIndex + 1; // Возвращаем следующий свободный индекс

}

else

{

return 1; // Если файл не открыт - он еще не создан

}

}

void input(int size)

{

PERSON buf;

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

{

buf.index = get\_index();

fopen\_s(&fdata, "data.bin", "ab");

cout << "Введите фамилию, имя, отчество гражданина (через пробел):\n";

cin.ignore();

cin.getline(buf.fio, 100);

cout << "Введите год рождения гражданина:\n";

cin >> buf.year\_of\_birth;

cout << "Введите адрес гражданина:\n";

cin.ignore();

cin.getline(buf.address, 200);

cout << "Введите пол гражданина (м или ж):\n";

cin >> buf.gender;

fwrite(&buf, sizeof(buf), 1, fdata);

fclose(fdata);

}

}

void showlist()

{

PERSON buf;

if (!fopen\_s(&fdata, "data.bin", "rb"))

{

while (fread(&buf, sizeof(buf), 1, fdata)) // Проверяем успешность чтения

{

cout << buf.index << ". " << buf.fio << "\t\t" << buf.year\_of\_birth << "\t\t" << buf.address << " \t" << buf.gender << endl;

}

fclose(fdata);

}

else

{

cout << "Ошибка открытия файла\n";

}

}

void del()

{

int index; // Индекс записи для удаления

cout << "Введите индекс человека, которого хотите удалить из списка: ";

cin >> index;

PERSON buf;

if (fopen\_s(&fdata, "data.bin", "rb")) // Открываем исходный файл для чтения

{

cout << "Ошибка открытия файла для чтения!\n";

return;

}

FILE\* tempFile; // Временный файл для перезаписи

if (fopen\_s(&tempFile, "temp.bin", "wb"))

{

cout << "Ошибка создания временного файла!\n";

fclose(fdata);

return;

}

bool found = false; // Флаг, что запись найдена

int newIndex = 1; // Новый индекс для переиндексации

while (fread(&buf, sizeof(PERSON), 1, fdata)) // Чтение всех записей из исходного файла и запись в новый файл

{

if (buf.index == index)

{

found = true; // Пропускаем удаляемую запись

continue;

}

buf.index = newIndex++; // Обновляем индекс для оставшихся записей

fwrite(&buf, sizeof(PERSON), 1, tempFile);

}

fclose(fdata);

fclose(tempFile);

remove("data.bin"); // Заменяем исходный файл временным

rename("temp.bin", "data.bin");

if (found)

{

cout << "Запись с индексом " << index << " успешно удалена.\n";

}

else

{

cout << "Запись с индексом " << index << " не найдена.\n";

}

}

void find()

{

cout << "Введите год рождения ";

int year;

cin >> year;

PERSON buf;

if (!fopen\_s(&fdata, "data.bin", "rb"))

{

while (fread(&buf, sizeof(buf), 1, fdata)) // Проверяем успешность чтения

{

if (year == buf.year\_of\_birth)

{

cout << "==========================\nФамилия: " << buf.fio << "\nГод рождения: " << buf.year\_of\_birth

<< "\nАдрес: " << buf.address << "\nПол: " << buf.gender << "\nИндекс в сипске: " << buf.index << "\n==========================\n";

}

}

fclose(fdata);

}

else

{

cout << "Ошибка открытия файла\n";

}

}

int main()

{

int size, user\_output;

setlocale(LC\_CTYPE, "Russian");

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

cout << "Выберите операцию:\n" << "1 - Ввод новой информации\n";

cout << "2 - Вывод заполненой информации\n" << "3 - Удаление информации\n";

cout << "4 - Поиск по году рождения\n";

do

{

cin >> user\_output;

switch (user\_output)

{

case 1:

cout << "Введите количество новых жителей города: ";

cin >> size;

input(size);

break;

case 2:

showlist();

break;

case 3:

del();

break;

case 4:

find();

break;

}

cout << "Что дальше?\n";

} while (user\_output != 0);

}

|  |  |
| --- | --- |
| 5. Дополнительные задания  1. Определить структуру для представления информации о сданных студентом экзаменах, содержащую поля: ФИО студента, число экзаменов, полученные оценки. Определить функции для обработки отдельного объекта (например, для проверки, сданы ли все экзамены на 4 и 5). Написать функцию для обработки массива структур. В результате обработки требуется вычислить характеристику успеваемости студентов, то есть отношение числа студентов, сдавших экзамены на 4 и 5, к общему числу студентов, в процентах.  2. Описать структуру с именем TRAIN, содержащую поля: названия пункта назначения, номер поезда, время отправления. Написать программу, выполняющую ввод с клавиатуры данных в массив, состоящий из восьми элементов типа TRAIN (записи должны быть размещены в алфавитном порядке по названиям пунктов назначения); вывод на экран информации о поездах, отправляющихся после введенного с клавиатуры времени (если таких поездов нет, то вывести сообщение об этом).  3. Определить структуру для представления информации о наличии санаторных путевок, содержащую следующие поля: Название санатория, Место расположения, Лечебный профиль, Количество путевок. Представить введенные данные в виде таблицы, сгруппировав их по лечебным профилям санаториев. В пределах каждой группы данные отсортировать по названиям санаториев. Организовать поиск информации и вывод результатов. | **5.1**    **5.2**      **5.3** |

**5.1**

#include <iostream>

#include <windows.h>

using namespace std;

FILE\* fdata;

typedef struct Student

{

int index; //Порядковый номер студента

char fio[100]; // Фамилия, имя, отчество

int grademat; // Оценка по вышмату

int gradeoaip; // Оценка по оаип

int gradeopi; // Оценка по опи

int gradebel; //Оценка по бел язу

} STU;

int get\_index()

{

STU buf;

int maxIndex = 0; // Хранит максимальный индекс найденной записи

if (!fopen\_s(&fdata, "data.bin", "rb")) // Открываем файл для чтения

{

while (fread(&buf, sizeof(STU), 1, fdata)) // Проходим по всем записям в файле

{

if (buf.index > maxIndex)

{

maxIndex = buf.index; // Обновляем максимальный индекс

}

}

fclose(fdata);

return maxIndex + 1; // Возвращаем следующий свободный индекс

}

else

{

return 1; // Если файл не открыт - он еще не создан

}

}

void input(int size)

{

STU buf;

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

{

buf.index = get\_index();

fopen\_s(&fdata, "data.bin", "ab");

cout << "Введите фамилию, имя, отчество гражданина (через пробел):\n";

cin.ignore();

cin.getline(buf.fio, 100);

cout << "Введите оценку студента за экзамен по выш.мату:\n";

cin >> buf.grademat;

cout << "Введите оценку студента за экзамен по оаип:\n";

cin >> buf.gradeoaip;

cout << "Введите оценку студента за экзамен по опи:\n";

cin >> buf.gradeopi;

cout << "Введите оценку студента за экзамен по бел. язу:\n";

cin >> buf.gradebel;

fwrite(&buf, sizeof(buf), 1, fdata);

fclose(fdata);

}

}

void showlist()

{

STU buf;

cout << " Фио студента: \t\t\t|\tОаип\t|\tОпи\t|\tВыш.мат\t|\tБел.яз\n";

cout << "\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n";

if (!fopen\_s(&fdata, "data.bin", "rb"))

{

while (fread(&buf, sizeof(buf), 1, fdata)) // Проверяем успешность чтения

{

cout << buf.index << ". " << buf.fio << "\r\t\t\t\t\t|\t" << buf.gradeoaip << "\t|\t" << buf.gradeopi << "\t|\t" << buf.grademat << "\t|\t" << buf.gradebel << endl;

}

fclose(fdata);

}

else

{

cout << "Ошибка открытия файла\n";

}

}

void getperf(int& quanoaip, int& quanopi, int& quanbel, int& quanmat,int& generalquan)

{

STU buf;

if (!fopen\_s(&fdata, "data.bin", "rb"))

{

while (fread(&buf, sizeof(STU), 1, fdata)) // Проходим по всем записям в файле

{

if (buf.gradeoaip >= 4)

{

quanoaip++;

}

if (buf.gradeopi >= 4)

{

quanopi++;

}

if (buf.gradebel >= 4)

{

quanbel++;

}

if (buf.grademat >= 4)

{

quanmat++;

}

if (buf.gradeoaip >= 4 && buf.gradeopi >= 4 && buf.gradebel >= 4 && buf.grademat >= 4)

{

generalquan++;

}

}

}

else

{

cout << "Ошибка открытия файла\n";

}

fclose(fdata);

}

void showperf(int& quanoaip, int& quanopi, int& quanbel, int& quanmat, int& generalquan)

{

double perf;

int quanstudents = get\_index() - 1;

int userinput;

cout << "Введите по какому предмету вас интересует успеваемость:\n";

cout << "1 - Оаип\n" << "2 - Опи\n" << "3 - Выш.мат\n" << "4 - Бел.яз\n" << "5 - Общая успеваемость\n" << "(0 - Для выхода из программы)\n";

do

{

cin >> userinput;

switch (userinput)

{

case 1:

perf = 100 \* quanoaip / quanstudents;

cout << "Успеваемость студентов по предмету оаип: " << perf << "%\n";

break;

case 2:

perf = 100 \* quanopi / quanstudents;

cout << "Успеваемость студентов по предмету опи: " << perf << "%\n";

break;

case 3:

perf = 100 \* quanmat / quanstudents;

cout << "Успеваемость студентов по предмету выш.мат: " << perf << "%\n";

break;

case 4:

perf = 100 \* quanbel / quanstudents;

cout << "Успеваемость студентов по предмету бел.яз: " << perf << "%\n";

break;

case 5:

perf = 100 \* generalquan / quanstudents;

cout << "Общая успеваемость студентов: " << perf << "%\n";

break;

case 0:

break;

default:

cout << "Ошибка ввода";

break;

}

} while (userinput != 0);

}

int main()

{

int size, user\_output, quanoaip = 0, quanopi = 0, quanbel = 0, quanmat = 0, generalquan = 0;

setlocale(LC\_CTYPE, "Russian");

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

cout << "Выберите операцию:\n" << "1 - Ввод новой информации\n";

cout << "2 - Вывод заполненой информации\n" << "3 - Показать сведения об общей успеваемости\n";

cout << "0 - Для выхода из программы\n";

do

{

cin >> user\_output;

switch (user\_output)

{

case 1:

cout << "Введите количество Студентов: ";

cin >> size;

input(size);

break;

case 2:

showlist();

break;

case 3:

getperf(quanoaip, quanopi, quanbel, quanmat, generalquan);

showperf(quanoaip, quanopi, quanbel, quanmat, generalquan);

break;

}

cout << "Что дальше?\n";

} while (user\_output != 0);

}

**5.2**

#include <iostream>

#include <windows.h>

using namespace std;

FILE\* fdata;

typedef struct Student

{

int index; //Порядковый номер студента

char fio[100]; // Фамилия, имя, отчество

int grademat; // Оценка по вышмату

int gradeoaip; // Оценка по оаип

int gradeopi; // Оценка по опи

int gradebel; //Оценка по бел язу

} STU;

int get\_index()

{

STU buf;

int maxIndex = 0; // Хранит максимальный индекс найденной записи

if (!fopen\_s(&fdata, "data.bin", "rb")) // Открываем файл для чтения

{

while (fread(&buf, sizeof(STU), 1, fdata)) // Проходим по всем записям в файле

{

if (buf.index > maxIndex)

{

maxIndex = buf.index; // Обновляем максимальный индекс

}

}

fclose(fdata);

return maxIndex + 1; // Возвращаем следующий свободный индекс

}

else

{

return 1; // Если файл не открыт - он еще не создан

}

}

void input(int size)

{

STU buf;

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

{

buf.index = get\_index();

fopen\_s(&fdata, "data.bin", "ab");

cout << "Введите фамилию, имя, отчество студента (через пробел):\n";

cin.ignore();

cin.getline(buf.fio, 100);

cout << "Введите оценку студента за экзамен по выш.мату:\n";

cin >> buf.grademat;

cout << "Введите оценку студента за экзамен по оаип:\n";

cin >> buf.gradeoaip;

cout << "Введите оценку студента за экзамен по опи:\n";

cin >> buf.gradeopi;

cout << "Введите оценку студента за экзамен по бел. язу:\n";

cin >> buf.gradebel;

fwrite(&buf, sizeof(buf), 1, fdata);

fclose(fdata);

}

}

void showlist()

{

STU buf;

cout << " Фио студента: \t\t\t|\tОаип\t|\tОпи\t|\tВыш.мат\t|\tБел.яз\n";

cout << "\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n";

if (!fopen\_s(&fdata, "data.bin", "rb"))

{

while (fread(&buf, sizeof(buf), 1, fdata)) // Проверяем успешность чтения

{

cout << buf.index << ". " << buf.fio << "\r\t\t\t\t\t|\t" << buf.gradeoaip << "\t|\t" << buf.gradeopi << "\t|\t" << buf.grademat << "\t|\t" << buf.gradebel << endl;

}

fclose(fdata);

}

else

{

cout << "Ошибка открытия файла\n";

}

}

void getperf(int& quanoaip, int& quanopi, int& quanbel, int& quanmat, int& generalquan)

{

STU buf;

if (!fopen\_s(&fdata, "data.bin", "rb"))

{

while (fread(&buf, sizeof(STU), 1, fdata)) // Проходим по всем записям в файле

{

if (buf.gradeoaip >= 4)

{

quanoaip++;

}

if (buf.gradeopi >= 4)

{

quanopi++;

}

if (buf.gradebel >= 4)

{

quanbel++;

}

if (buf.grademat >= 4)

{

quanmat++;

}

if (buf.gradeoaip >= 4 && buf.gradeopi >= 4 && buf.gradebel >= 4 && buf.grademat >= 4)

{

generalquan++;

}

}

}

else

{

cout << "Ошибка открытия файла\n";

}

fclose(fdata);

}

void showperf(int& quanoaip, int& quanopi, int& quanbel, int& quanmat, int& generalquan)

{

double perf;

int quanstudents = get\_index() - 1;

int userinput;

cout << "Введите по какому предмету вас интересует успеваемость:\n";

cout << "1 - Оаип\n" << "2 - Опи\n" << "3 - Выш.мат\n" << "4 - Бел.яз\n" << "5 - Общая успеваемость\n" << "(0 - Для выхода из программы)\n";

do

{

cin >> userinput;

switch (userinput)

{

case 1:

perf = 100 \* quanoaip / quanstudents;

cout << "Успеваемость студентов по предмету оаип: " << perf << "%\n";

break;

case 2:

perf = 100 \* quanopi / quanstudents;

cout << "Успеваемость студентов по предмету опи: " << perf << "%\n";

break;

case 3:

perf = 100 \* quanmat / quanstudents;

cout << "Успеваемость студентов по предмету выш.мат: " << perf << "%\n";

break;

case 4:

perf = 100 \* quanbel / quanstudents;

cout << "Успеваемость студентов по предмету бел.яз: " << perf << "%\n";

break;

case 5:

perf = 100 \* generalquan / quanstudents;

cout << "Общая успеваемость студентов: " << perf << "%\n";

break;

case 0:

break;

default:

cout << "Ошибка ввода";

break;

}

} while (userinput != 0);

}

int main()

{

int size, user\_output, quanoaip = 0, quanopi = 0, quanbel = 0, quanmat = 0, generalquan = 0;

setlocale(LC\_CTYPE, "Russian");

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

cout << "Выберите операцию:\n" << "1 - Ввод новой информации\n";

cout << "2 - Вывод заполненой информации\n" << "3 - Показать сведения об общей успеваемости\n";

cout << "0 - Для выхода из программы\n";

do

{

cin >> user\_output;

switch (user\_output)

{

case 1:

cout << "Введите количество Студентов: ";

cin >> size;

input(size);

break;

case 2:

showlist();

break;

case 3:

getperf(quanoaip, quanopi, quanbel, quanmat, generalquan);

showperf(quanoaip, quanopi, quanbel, quanmat, generalquan);

break;

}

cout << "Что дальше?\n";

} while (user\_output != 0);

}

**5.2**

#include <iostream>

#include <windows.h>

#include <cstring>

using namespace std;

FILE\* fdata;

typedef struct TRAIN {

int index; // Номер поезда

char pose[100]; // Пункт назначения

char time[5]; // Время отправления

} TR;

TR\* sorttrain(TR buf[], int size);

void input() {

const int SIZE = 8;

TR buf[SIZE];

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

{

buf[i].index = i + 1;

cout << "Введите информацию о " << buf[i].index << " поезде:\n";

cout << "Пункт назначения:\n";

cin.ignore();

cin.getline(buf[i].pose, 100);

cout << "Время отправления:\n";

cin >> buf[i].time;

}

fopen\_s(&fdata, "data.bin", "ab");

fwrite(buf, sizeof(TR), SIZE, fdata); // Записываем каждый поезд по отдельности

fclose(fdata);

cout << "Информация успешно записана\n";

}

void showlist() {

const int SIZE = 8;

TR buf[SIZE];

cout << "Пункт назначения: \t\t\t|\tВремя отправления\t|\tНомер поезда\t|\n";

cout << "\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n";

if (!fopen\_s(&fdata, "data.bin", "rb")) {

fread(buf, sizeof(TR), SIZE, fdata); // Читаем все записи

fclose(fdata);

}

else {

cout << "Ошибка открытия файла для чтения\n";

return;

}

TR\* sortedBuf = sorttrain(buf, SIZE); // Сортируем данные

for (int i = 0; i < SIZE; i++) {

cout << sortedBuf[i].pose << "\r\t\t\t\t\t|\t\t" << sortedBuf[i].time << "\t\t|\t\t" << sortedBuf[i].index << "\t|\n";

}

}

TR\* sorttrain(TR buf[], int size) {

TR sort;

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

{

for (int j = i + 1; j < size; j++)

{

if (strcmp(buf[i].pose, buf[j].pose) > 0) // Сравниваем строки (пункты назначения)

{

sort = buf[i];

buf[i] = buf[j];

buf[j] = sort;

} // Меняем местами все поля

}

}

return buf;

}

bool checktime(char time[], TR buf)

{

for (int i = 0; i < 5; i++)

{

if (i == 2)

{

continue;

}

if (buf.time[i] > time[i])

{

return 1;

}

else if (buf.time[i] < time[i])

{

return 0;

}

}

}

void searchtrain(char time[])

{

int check=0, SIZE = 8;

TR buf[8];

if (!fopen\_s(&fdata, "data.bin", "rb"))

{

while (fread(buf, sizeof(TR), SIZE, fdata));

}

else

{

cout << "Не удалось открыть файл";

}

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

{

if (checktime(time,buf[i]))

{

cout << buf[i].pose << "\t\t" << buf[i].time << "\t\t" << buf[i].index << endl;

}

}

}

int main() {

setlocale(LC\_CTYPE, "Russian");

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

char time[5];

int user\_input;

cout << "Выберите операцию:\n" << "1 - Ввод новой информации\n";

cout << "2 - Вывод информации о всех поездах\n" << "3 - Поиск поезда\n";

cout << "0 - Для выхода из программы\n";

do {

cin >> user\_input;

switch (user\_input) {

case 1:

input();

break;

case 2:

showlist();

break;

case 3:

cout << "Введите время, после которого вы свободны\n";

cin >> time;

searchtrain(time);

break;

case 0:

break;

}

cout << "Что дальше?\n";

} while (user\_input != 0);

return 0;

}

**5.3**

#include <iostream>

#include <windows.h>

#include <string>

using namespace std;

FILE\* fdata, \* fnewdata;

typedef struct Voucher

{

int index; //Индекс

char name[100]; //Название санатория

int vouchers; //Количество путевок

char location[100]; //Расположение

char profile[100]; //Лечебный профиль

int price; //Цена

} VOUCH;

int get\_index()

{

VOUCH buf;

int maxIndex = 0; // Хранит максимальный индекс найденной записи

if (!fopen\_s(&fdata, "data.bin", "rb")) // Открываем файл для чтения

{

while (fread(&buf, sizeof(VOUCH), 1, fdata)) // Проходим по всем записям в файле

{

if (buf.index > maxIndex)

{

maxIndex = buf.index; // Обновляем максимальный индекс

}

}

fclose(fdata);

return maxIndex + 1; // Возвращаем следующий свободный индекс

}

else

{

return 1; // Если файл не открыт - он еще не создан

}

}

void input(int size)

{

VOUCH buf;

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

{

buf.index = get\_index();

fopen\_s(&fdata, "data.bin", "ab");

cout << "Введите название санатория: ";

cin.ignore();

cin.getline(buf.name, 100);

cout << "Введите расположение санатория: ";

cin.getline(buf.location, 100);

cout << "Введите какой у санатория лечебный профиль: ";

cin.getline(buf.profile, 100);

cout << "Введите цену: ";

cin >> buf.price;

cout << "Введите количество путевок: ";

cin >> buf.vouchers;

fwrite(&buf, sizeof(buf), 1, fdata);

fclose(fdata);

}

}

void showlist()

{

VOUCH buf;

cout << " Название санатория\t|\tРасположение\t|\tЛечебный профиль\t|\tЦена\t| Количество путевок\n";

cout << "\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n";

if (!fopen\_s(&fdata, "data.bin", "rb"))

{

while (fread(&buf, sizeof(buf), 1, fdata)) // Проверяем успешность чтения

{

cout << buf.index << ". " << buf.name << "\r\t\t\t|" << buf.location;

cout << "\r\t\t\t\t\t\t|" << buf.profile << "\r\t\t\t\t\t\t\t\t\t\t|\t" << buf.price << "\t|\t" << buf.vouchers << endl;

}

fclose(fdata);

}

else

{

cout << "Ошибка открытия файла\n";

}

}

void searchVoucher()

{

VOUCH buf;

cout << "Введите желаемый профиль: ";

string profile;

bool search = false;

cin.ignore();

getline(cin, profile);

cout << "Найденые путевки:\n";

if (!fopen\_s(&fdata, "data.bin", "rb"))

{

while (fread(&buf, sizeof(VOUCH), 1, fdata))

{

string profile2 = string(buf.profile);

if (profile == profile2)

{

cout << "\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n";

cout << "Лечебный профиль санатория: " << buf.profile << "\n" << "Цена путевки: " << buf.price << "\n";

cout << "Расположение санатория: " << buf.location << "\n" << "Название санатория:" << buf.name << "\n";

cout << "Количество путевок: " << buf.vouchers << "\n" << "Индекс для покупки: #" << buf.index << endl;

cout << "\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n";

search = true;

}

}

}

else

{

cout << "Ошибка открытия файла";

}

if (!search)

{

cout << "Путевок с заданым лечебным профилем не найдено\n";

}

}

void numOfVouchers(int index)

{

VOUCH buf;

if (!fopen\_s(&fnewdata, "newdata.bin", "ab"))

{

if (!fopen\_s(&fdata, "data.bin", "rb"))

{

while (fread(&buf, sizeof(buf), 1, fdata))

{

if (index == buf.index)

{

buf.vouchers--;

}

fwrite(&buf, sizeof(buf), 1, fnewdata);

}

}

else

{

cout << "Ошибка открытия файла";

}

}

else

{

cout << "Ошибка создания нового файла ";

}

fclose(fnewdata);

fclose(fdata);

remove("data.bin");

rename("newdata.bin", "data.bin");

}

void buyVoucher()

{

VOUCH buf;

int index, budget, userInput, summa;

bool successfulBuy = false;

cout << "Введите ваш бюджет: ";

cin >> budget;

do

{

cout << "Введите индекс желаемой путевки: ";

cin >> index;

if (!fopen\_s(&fdata, "data.bin", "rb"))

{

while (fread(&buf, sizeof(buf), 1, fdata))

{

if (index == buf.index)

{

if (budget >= buf.price)

{

budget -= buf.price;

successfulBuy = true;

cout << "Покупка произошла успешно\n";

cout << "\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n";

cout << "Лечебный профиль санатория: " << buf.profile << "\n" << "Цена путевки: " << buf.price << "\n";

cout << "Расположение санатория: " << buf.location << "\n" << "Название санатория:" << buf.name << "\n";

cout << "Количество путевок: " << buf.vouchers << "\n" << "Индекс для покупки: #" << buf.index << endl;

cout << "\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n";

}

else

{

cout << "Недостаточно средств\n";

}

}

}

}

else

{

cout << "Ошибка открытия файла\n";

}

fclose(fdata);

if (successfulBuy)

{

numOfVouchers(index);

}

cout << "Продолжить покупку?\n" << "1 - Да,продолжить покупку без пополнения баланса\n";

cout << "2 - Да,продолжить покупку и пополнить баланс\n" << "3 - Узнать баланс\n" << "4 - Нет, выход из меню покупки\n";

cin >> userInput;

if (userInput == 2)

{

cout << "Введите сумму на которую хотите пополнить бюджет: ";

cin >> summa;

budget += summa;

}

if (userInput == 3)

{

cout << "Ваш баланс: " << budget << endl;

}

} while (userInput != 4);

}

int main()

{

int size, user\_output;

setlocale(LC\_CTYPE, "Russian");

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

cout << "Выберите операцию:\n" << "1 - Ввод новой информации\n";

cout << "2 - Вывод заполненой информации\n" << "3 - Поиск санатория по лечебному профилю\n";

cout << "4 - Приобрести путевку по индексу\n" << "0 - Для выхода из программы\n";

do

{

cin >> user\_output;

switch (user\_output)

{

case 1:

cout << "Введите сколько новых санаториев хотите записать\n";

cin >> size;

cout << "Введите информацию о санаториях: \n";

input(size);

break;

case 2:

showlist();

break;

case 3:

searchVoucher();

break;

case 4:

buyVoucher();

case 0:

break;

}

cout << "Что дальше?\n";

} while (user\_output != 0);

}