Белорусский государственный технологический университет

Факультет информационных технологий

Кафедра программной инженерии

Лабораторная работа 4

По дисциплине «Основы алгоритмизации и программирования»

На тему «Представление информации в виде структуры»

Выполнила:

Студентка 1 курса 2 группы

Глухова Д.В.

Преподаватель: асс. Андронова М.В.

2023, Минск

Вариант 2.

4. 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.

**Список клиентов гостиницы.** Паспортные данные, даты приезда и отъезда, номер, тип размещения (люкс, одноместный, двухместный, трехместный, апартаменты). Поиск гостя по фамилии.

|  |
| --- |
| Программный код |
| // Список клиентов гостиницы. Паспортные данные, даты приезда и отъезда, номер, тип размещения (люкс, одноместный, двухместный, трехместный, апартаменты). Поиск гостя по фамилии.  #include <iostream>  #include <windows.h>  #include <stdio.h>  #include <fstream>  #include <string>  using namespace std;  // объявляем структуру  struct HotelClients  {  string fio; // фамилия клиента гостиницы  string passNumber; // паспортные данные  string dateIn; // дата приезда  string dateOut; // дата отъезда  string room; // номер  string typeRoom; // тип размещения  };  HotelClients clients[20];  void output();  void input();  void find();  void del();  int main()  {  SetConsoleCP(1251);  SetConsoleOutputCP(1251);  setlocale(LC\_ALL, ".1251");  int choice;    do  {  cout << "\n1.Ввод данных с клавиатуры и запись в файл\n";  cout << "2.Вывод данных из файла в консольное окно\n";  cout << "3.Поиск по фамилии\n";  cout << "4.Удаление заданной структурированной переменной\n";  cout << "0.Выход из программы\n\n";  cout << "Введите номер операции: ";  cin >> choice;  switch (choice)  {  case 1: input(); break;  case 2: output(); break;  case 3: find(); break;  case 4: del(); break;  case 0: exit(0); break;  }  } while (choice != 0);  }  // функция для ввода и записи в файл  void input()  {  int num;  do {  cout << "Введите количество клиентов банка (не более 20)";  cin >> num;  } while (num < 1 || num > 20);  fflush(stdin);    ofstream file1("information.txt");    for (int i = 0; i < num; i++)  {  cout << "\tКлиент №" << i + 1 << endl;  cout << "\nВведите ФИО клиента:\n";  cin >> clients[i].fio;  file1 << "\tКлиент №" << i + 1 << "\n";  file1 << clients[i].fio;  file1 << "\n";  cout << "\nВведите номер паспорта:\n";  cin >> clients[i].passNumber;  file1 << clients[i].passNumber;  file1 << "\n";  cout << "\nВведите дату приезда:\n";  cin >> clients[i].dateIn;  file1 << clients[i].dateIn;  file1 << "\n";  cout << "\nВведите дату отъезда:\n";  cin >> clients[i].dateOut;  file1 << clients[i].dateOut;  file1 << "\n";  cout << "\nВведите комнату:\n";  cin >> clients[i].room;  file1 << clients[i].room;  file1 << "\n";  cout << "\nВведите тип комнаты:\n";  cin >> clients[i].typeRoom;  file1 << clients[i].typeRoom;  file1 << "\n";  cout << "\n\n";  }  file1.close();  }  // функция для вывода в консоль  void output()  {  int ClientNum;  cout << "Введите номер клиента, которого вы хотите вывести:\n";  cin >> ClientNum;  int i = ClientNum - 1;  cout << "\tКлиент №" << ClientNum << '\n';  cout << "ФИО: " << clients[i].fio << '\n';  cout << "Паспорт: " << clients[i].passNumber << '\n';  cout << "Дата приезда: " << clients[i].dateIn << '\n';  cout << "Дата отъезда: " << clients[i].dateOut << '\n';  cout << "Номер: " << clients[i].room << "\n\n\n";  cout << "Тип номера: " << clients[i].typeRoom << "\n\n\n";  }  // функция для удаления данных о клиенте  void del()  {  int ClientNum;  cout << "Введите порядковый номер клиента, информацию о котором Вы хотите удалить:\n";  cin >> ClientNum;  cout << '\n';  int i = ClientNum - 1;  clients[i].fio = ' ';  clients[i].passNumber = ' ';  clients[i].dateIn = ' ';  clients[i].dateOut = ' ';  clients[i].room = ' ';  clients[i].typeRoom = ' ';  cout << '\n';  }  // функция для поиска клиента  void find()  {  string fioSearch;  cout << "Введите фамилию искомого клиента:";  cin >> fioSearch;  for (int i = 0; i < 20; i++)  {  if (fioSearch == clients[i].fio)  {  cout << "\tКлиент №" << i + 1 << '\n';  cout << "ФИО: " << clients[i].fio << '\n';  cout << "Паспорт: " << clients[i].passNumber << '\n';  cout << "Дата приезда: " << clients[i].dateIn << '\n';  cout << "Дата отъезда: " << clients[i].dateOut << '\n';  cout << "Номер: " << clients[i].room << "\n\n\n";  cout << "Тип номера: " << clients[i].typeRoom << "\n\n\n";  }  }  cout << "\n\n";  } |
| Консоль отладки |
|  |
| Конечный файл |
|  |

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.

|  |
| --- |
| Программный код |
| // 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.  // СПЕРВА НУЖНО ПРОЧЕСТЬ ФАЙЛ(3), ЗАТЕМ ВЫПОЛНЯТЬ ОСТАЛЬНЫЕ ДЕЙСТВИЯ!!!  #include <iostream>  #include <windows.h>  #include <stdio.h>  #include <fstream>  #include <string>  using namespace std;  // объявляем структуру  struct Student {  string name;  int maths;  int physics;  int english;  int oap;  };  // вывод студентов сдал/не сдал на 4-5 и процент  void exams(Student\* students, int SIZE) {  int numStudents = 0;  for (int i = 0; i < SIZE; ++i) {  if ((students[i].maths == 5 || students[i].maths == 4) &&  (students[i].physics == 5 || students[i].physics == 4) &&  (students[i].english == 5 || students[i].english == 4) &&  (students[i].oap == 5 || students[i].oap == 4))  {  cout << i << ") Студент: " << students[i].name << " сдал экзамены на 4 или 5" << endl;  numStudents++;  }  else cout << i << ") Студент: " << students[i].name << " не сдал экзамены на 4 или 5" << endl;  }  cout << "Сотношение студентов, сдавших экзамен на 4 или 5, ко всем студентам " << ((double)numStudents / (double)SIZE) \* 100 << " % " << endl;  }  // функция вывода на экран информации о ствденте и резах экзамена  void displayInfo(Student\* students, int SIZE) {  for (int i = 0; i < SIZE; ++i) {  cout << "Имя студента: " << students[i].name << endl;  cout << "Математика: " << students[i].maths << endl;  cout << "Физика: " << students[i].physics << endl;  cout << "Английский: " << students[i].english << endl;  cout << "ОАП: " << students[i].oap << endl;  cout << "============================================" << endl;  }  }  // функция чтения из файла  void ifileStud(Student\* students, int SIZE) {  ifstream fin("information.txt");  if (!fin.is\_open()) {  cout << "Не удаётся открыть файл для чтения " << endl;  exit(EXIT\_FAILURE);  }  for (int i = 0; i < SIZE; ++i) {  fin >> students[i].maths;  fin >> students[i].physics;  fin >> students[i].english;  fin >> students[i].oap;  fin >> students[i].name;  }  cout << "\nВыведена информация из файла\n";  }  int main()  {  SetConsoleCP(1251);  SetConsoleOutputCP(1251);  setlocale(LC\_ALL, ".1251");  const int SIZE = 10;  Student students[SIZE];  short choose;  do {  cout << "\nВыберите, что сделать:\n1 — проверить, сколько студентов положительно сдали экзамены;\n2 — вывести в консоль информацию о студенте;\n3 — вывести информацию из файла;\n4 — выход из программы.\n";  cin >> choose;  switch (choose)  {  case 1: exams(students, SIZE); break;  case 2: displayInfo(students, SIZE); break;  case 3: ifileStud(students, SIZE); break;  case 4: exit(0); break;  }  } while (choose != 4);  } |
| Консоль отладки |
|  |
| Исходный файл |
|  |

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).

|  |
| --- |
| Программный код |
| #include <string>  #include <fstream>  #include <iostream>  #include <Windows.h>  using namespace std;  #define str\_len 40  #define size 8  string A = "file2.txt"; // файл который будет использоваться  char str[512];  // структура  struct Trains  {  char name[str\_len]; //  int num;  int time;  };  struct Trains list\_of\_trains[size], free\_train, help\_train;  int currsize = 0;  void enter\_new\_tr();  void del\_tr(int d);  void out\_tr();  void write\_file\_tr();  void sorting();  void read\_file\_tr();  void main()  {  SetConsoleCP(1251);  SetConsoleOutputCP(1251);  int choice = 1, d;  while (choice != 0)  {  cout << " \n1. Новая запись\n2. Удалить запись\n3. Вывести запись в консоль\n4. Запись в файл\n5. Чтение из файла\n6. Сортировка\n0. Завершить работу\n(Введите выбор) ";  cin >> choice;  switch (choice)  {  case(1):  enter\_new\_tr();  break;  case(2):  cout << " Номер строки, которую надо удалить(для удаления всех строк введите -1): ";  cin >> d;  del\_tr(d);  break;  case(3):  out\_tr();  break;  case(4):  write\_file\_tr();  break;  case(5):  read\_file\_tr();  break;  case(6):  sorting();  break;  default:  break;  }  }  }  // функция ввода  void enter\_new\_tr()  {  int num;  do  {  cout << "Введите количество поездов (не более " << size << "): ";  cin >> num;  } while (num < 1 || num > size);  fflush(stdin);  for (int i = 0; i < num; i++)  {  cout << "Запись №" << currsize + 1 << endl;  cout << "Введите названия пункта назначения" << endl;  cin >> list\_of\_trains[currsize].name;  cout << "Введите номер поезда" << endl;  cin >> list\_of\_trains[currsize].num;  cout << "Введите время отправления в формате ЧЧММ" << endl;  cin >> list\_of\_trains[currsize].time;  currsize++;  }  }  // для удаления  void del\_tr(int d)  {  if (d == -1)  {  for (int i = 0; i < size; i++)  {  list\_of\_trains[i] = free\_train;  }  currsize = 0;  }  else  {  for (int i = d - 1; i < currsize; i++)  {  list\_of\_trains[i] = list\_of\_trains[i + 1];  }  currsize--;  }  }  // функция вывода после введенного времени  void out\_tr()  {  int t, q = 0;  cout << "Введите время: ";  cin >> t;  for (int i = 0; i < currsize; i++)  {  if (list\_of\_trains[i].time >= t)  {  cout << " Запись №" << i + 1 << " | название пункта назначения - " << list\_of\_trains[i].name << " | номер поезда - " << list\_of\_trains[i].num << " | время отправления - " << list\_of\_trains[i].time << endl;  q++;  }  }  if (q == 0) cout << "Таких поездов нет" << endl;  }  // запись в файл информации  void write\_file\_tr()  {  ofstream fAout(A);  if (fAout.is\_open())  {  for (int i = 0; i < currsize; i++)  {  fAout << list\_of\_trains[i].name << endl;;  fAout << list\_of\_trains[i].num << endl;;  fAout << list\_of\_trains[i].time << endl;;  }  }  else  {  cout << " Невозможно открыть файл!" << endl;  }  fAout.close();  }  // чтение файла  void read\_file\_tr()  {  del\_tr(-1);  char str2[200], str3[200];  ifstream fAin(A);  if (fAin.is\_open())  {  while (true)  {  fAin.getline(list\_of\_trains[currsize].name, sizeof(list\_of\_trains[currsize].name));  fAin.getline(str2, sizeof(str2));  fAin.getline(str3, sizeof(str3));  if (fAin.eof() != 0) break;  list\_of\_trains[currsize].num = atoi(str2);  list\_of\_trains[currsize].time = atoi(str3);  currsize++;  }  }  else  {  cout << " Невозможно открыть файл!" << endl;  }  fAin.close();  }  // сортировка  void sorting()  {  int q, i;  for (i = 0; i < currsize - 1; i++)  {  q = (strlen(list\_of\_trains[i].name) > strlen(list\_of\_trains[i + 1].name)) ? strlen(list\_of\_trains[i + 1].name) : strlen(list\_of\_trains[i].name);  for (int j = 0; j < q; j++)  {  if (list\_of\_trains[i].name[j] < list\_of\_trains[i + 1].name[j]) break;  if (list\_of\_trains[i].name[j] > list\_of\_trains[i + 1].name[j])  {  help\_train = list\_of\_trains[i + 1];  list\_of\_trains[i + 1] = list\_of\_trains[i];  list\_of\_trains[i] = help\_train;  i = -1;  break;  }  }  }  } |
| Консоль отладки |
|  |
| Конечный файл |
|  |