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

|  |  |  |
| --- | --- | --- |
| Задание | | Решение |
| 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. | | **#include <iostream>**  **using namespace std;**  **struct Worker**  **{ char lastName[30];**  **char position[30];**  **int year;**  **};**  **int main()**  **{ setlocale(LC\_ALL, "Russian");**  **const int size = 3;**  **const int currentYear = 2017;**  **int i, b, counter = 0;**  **Worker a[size];**  **for (i = 0; i < size; i++)**  **{ cout << "Введите фамилию " << i + 1 << "-ого работника " << endl;**  **cin >> a[i].lastName;**  **cout << endl << "Должность: ";**  **cin >> a[i].position;**  **cout<< endl << "Год поступления на работу: ";**  **cin >> a[i].year;**  **}**  **cout << endl << "Введите стаж работы ";**  **cin >> b;**  **cout << endl;**  **for (i = 0; i < size; i++)**  **{ if (b < currentYear - a[i].year)**  **{ cout << a[i].lastName << " ";**  **counter++;**  **}**  **else**  **if (i == size-1 && counter > 0)**  **cout << "Работников с более высоким стажем нет.";**  **}**  **return 0;**  **}**    #include <iostream>  using namespace std;  struct Worker  {  char lastName[30];  char position[30];  int year;  int age; // Новый атрибут - возраст  double salary; // Новый атрибут - зарплата  };  int main()  {  setlocale(LC\_ALL, "Russian");  const int size = 3;  const int currentYear = 2025;  int i, b, counter = 0;  Worker a[size];  for (i = 0; i < size; i++)  {  cout << "Введите фамилию " << i + 1 << "-ого работника: ";  cin >> a[i].lastName;  cout << "Должность: ";  cin >> a[i].position;  cout << "Год поступления на работу: ";  cin >> a[i].year;  cout << "Возраст: ";  cin >> a[i].age; // Ввод возраста  cout << "Зарплата: ";  cin >> a[i].salary; // Ввод зарплаты  }  cout << "\nВведите стаж работы: ";  cin >> b;  cout << endl;  for (i = 0; i < size; i++)  {  if (b < currentYear - a[i].year)  {  cout << "Фамилия: " << a[i].lastName<< ", Должность: " << a[i].position<< ", Год поступления: " << a[i].year<< ", Возраст: " << a[i].age<< ", Зарплата: " << a[i].salary << endl;  counter++;  }  }  if (counter == 0)  {  cout << "Работников с более высоким стажем нет." << endl;  }  return 0;  } |
| 2. В правой части приведен пример программы, в которой используется *указатель на структуру*.  Выполнить программу, меняя скорость вывода информации.  Написать пояснения к программе. | | #include <conio.h>  #include <iostream>  using namespace std;  #define DELAY 12800 // Определение задержки в цикле обновления  // Объявление функций для обновления и отображения времени  void update(struct MyTime\* t);  void display(struct MyTime\* t);  // Структура для хранения времени  struct MyTime  {  int hours; // Часы  int minutes; // Минуты  int seconds; // Секунды  };  int main(void)  {  struct MyTime St; // Создание переменной типа MyTime  St.hours = 0; // Инициализация часов  St.minutes = 0; // Инициализация минут  St.seconds = 0; // Инициализация секунд  for (; ;) // Бесконечный цикл  {  update(&St); // Обновление времени  display(&St); // Отображение времени  if (\_kbhit()) // Проверка, была ли нажата клавиша  return 0; // Выход из программы при нажатии клавиши  }  }  void update(struct MyTime\* t)  {  t->seconds++; // Увеличение секунд на 1  if (t->seconds == 60) { t->seconds = 0; t->minutes++; } // Если секунд 60, сбросить до 0 и увеличить минуты  if (t->minutes == 60) { t->minutes = 0; t->hours++; } // Если минут 60, сбросить до 0 и увеличить часы  if (t->hours == 24) // Если часов 24, сбросить до 0  t->hours = 0;  for (long int i = 1; i < DELAY; ++i); // Задержка для замедления обновления  }  void display(struct MyTime\* t)  {  cout << t->hours << '.' << t->minutes << '.' << t->seconds << endl; // Вывод текущего времени  } |
| 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>  # 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()  {  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 >> choice;  do  {  switch (choice)  {  case 1: del(); break;  case 2: enter\_new(); break;  case 3: change(); break;  case 4: out(); break;  }  } while (choice != 5);  }  void enter\_new()  {  cout << "Ввод информации" << endl;  if (current\_size < size)  {  cout << "Строка номер ";  cout << current\_size + 1;  cout << endl << "Фамилия " << endl;  cin >> list\_of\_student[current\_size].name;  cout << "Год рождения " << endl;  cin >> list\_of\_student[current\_size].year\_of\_birth;  cout << "Факультет " << endl;  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;  }    #include <iostream>  #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() {  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 >> choice;  do {  switch (choice) {  case 1: del(); break;  case 2: enter\_new(); break;  case 3: change(); break;  case 4: out(); break;  }  } while (choice != 5);  }  void enter\_new() {  cout << "Ввод информации" << endl;  if (current\_size < size) {  cout << "Строка номер " << current\_size + 1 << endl;  cout << "Фамилия " << endl;  cin >> list\_of\_student[current\_size].name;  cout << "Год рождения " << endl;  while (!(cin >> list\_of\_student[current\_size].year\_of\_birth)) {  cout << "Ошибка ввода! Пожалуйста, введите год рождения (число): " << endl;  cin.clear(); // Clear the error flag  cin.ignore(10000, '\n'); // Discard invalid input  }  cout << "Факультет " << endl;  cin >> list\_of\_student[current\_size].department;  current\_size++;  }  else {  cout << "Введено максимальное кол-во строк" << endl;  }  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 - 1; de1++) {  list\_of\_student[de1] = list\_of\_student[de1 + 1];  }  current\_size--;  }  else {  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: ";  while (!(cin >> list\_of\_student[n - 1].year\_of\_birth)) {  cout << "Ошибка ввода! Пожалуйста, введите год рождения (число): " << endl;  cin.clear();  cin.ignore(10000, '\n'); // Discard invalid input  }  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 << "Фамилия: " << list\_of\_student[n - 1].name << endl;  cout << "Год рождения: " << list\_of\_student[n - 1].year\_of\_birth << endl;  cout << "Факультет: " << list\_of\_student[n - 1].department << endl;  }  else if (sw == 2) {  for (int i = 0; i < current\_size; i++) {  cout << "Фамилия: " << list\_of\_student[i].name << endl;  cout << "Год рождения: " << list\_of\_student[i].year\_of\_birth << endl;  cout << "Факультет: " << 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>  using namespace std;  // Функции для ввода, вывода и поиска данных о студентах  void input(int size);  void output();  void find(char lastName[]);  // Структура для хранения информации о студенте  typedef struct Students {  char fio[16]; // Фамилия студента  char group[3]; // Группа студента  } STUD;  int number; // Количество студентов  FILE\* f; // Указатель на файл  errno\_t err; // Переменная для хранения кода ошибки  int main() {  setlocale(LC\_ALL, "Russian"); // Установка локали для работы с русским языком  int choice; // Переменная для хранения выбора пользователя  char fio[16]; // Переменная для хранения фамилии  do {  // Меню программы  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);  }  // Функция для ввода данных о студентах  void input(int size) {  STUD buf = { ' ', ' ' }; // Инициализация структуры студента  if (!fopen\_s(&f, "base.bin", "ab")) { // Открытие файла для добавления данных  for (int p = 0; p < size; p++) {  cout << "Фамилия: ";  cin >> buf.fio; // Ввод фамилии  cout << "Группа: ";  cin >> buf.group; // Ввод группы  fwrite(&buf, sizeof(buf), 1, f); // Запись данных в файл  }  fclose(f); // Закрытие файла  }  else {  cout << "Ошибка открытия файла"; // Обработка ошибки открытия файла  return;  }  }  // Функция для вывода данных о студентах  void output() {  STUD buf; // Структура для хранения данных из файла  if (!fopen\_s(&f, "base.bin", "rb")) { // Открытие файла для чтения  cout << "\nФамилия Группа\n";  fread(&buf, sizeof(buf), 1, f); // Чтение данных из файла  while (!feof(f)) {  cout << buf.fio << "\t " << buf.group << endl; // Вывод данных  fread(&buf, sizeof(buf), 1, f); // Чтение следующей записи  }  cout << endl;  fclose(f); // Закрытие файла  }  else {  cout << "Ошибка открытия файла"; // Обработка ошибки открытия файла  return;  }  }  // Функция для поиска студента по фамилии  void find(char lastName[16]) {  bool flag = false; // Флаг для отслеживания нахождения студента  STUD buf; // Структура для хранения данных из файла  if (!fopen\_s(&f, "base.bin", "rb")) { // Открытие файла для чтения  while (!feof(f)) {  fread(&buf, sizeof(buf), 1, f); // Чтение данных из файла  if (strcmp(lastName, buf.fio) == 0) { // Сравнение фамилий  cout << "\nФамилия Группа\n";  cout << buf.fio << "\t " << buf.group << endl; // Вывод найденного студента  flag = true;  break;  }  }  fclose(f); // Закрытие файла  if (!flag)  cout << "Ничего не найдено\n"; // Если студент не найден  }  else {  cout << "Ошибка открытия файла"; // Обработка ошибки 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. | | |
| №Варианта  10 | | |
| Задание | Код | |
| **Справочник автомобилей.** Марка автомобиля, цвет, заводской номер, дата выпуска, тип кузова (седан, универсал и т. п.), дата последнего техосмотра, владелец. Выбор транспортных средств по владельцу. | #include <iostream>  #include <fstream>  #include <iomanip>  using namespace std;  struct Cars {  char brand[20];  char color[20];  int serial;  char release[20];  char type[20];  char check[20];  char owner[20];  };  void readMas(Cars all[20], int& size) {  cin >> all[size].brand >> all[size].color >> all[size].serial >> all[size].release >> all[size].type >> all[size].check >> all[size].owner;  size++;  }  void writeMas(Cars all[20], int size) {  cout << left << setw(15) << "Марка"<< setw(10) << "Цвет"<< setw(15) << "Заводской номер"<< setw(15) << "Дата выпуска"<< setw(15) << "Тип кузова"<< setw(20) << "Дата техосмотра"<< setw(15) << "Владелец"<< endl;  for (int i = 0; i < size; i++) {  cout << left<< setw(15) << all[i].brand<< setw(10) << all[i].color<< setw(15) << all[i].serial<< setw(15) << all[i].release<< setw(15) << all[i].type<< setw(20) << all[i].check<< setw(15) << all[i].owner<< endl;  }  }  void writeFile(Cars all[20], int size, const char filename[]) {  ofstream fout(filename);  for (int i = 0; i < size; i++) {  fout << all[i].brand << "\t" << all[i].color << "\t" << all[i].serial << "\t" << all[i].release << "\t" << all[i].type << "\t" << all[i].check << "\t" << all[i].owner << "\n";  }  fout.close();  }  void readFile(Cars all[20], int& size, const char filename[]) {  ifstream fin(filename);  Cars newall[20];  size = 0;  while (fin >> newall[size].brand) {  fin >> newall[size].color >> newall[size].serial >> newall[size].release >> newall[size].type >> newall[size].check >> newall[size].owner;  size++;  }  for (int i = 0; i < size; i++) {  all[i] = newall[i];  }  fin.close();  }  int searchMas(Cars all[20], int size, const char\* item) {  for (int i = 0; i < size; i++) {  bool flag = true;  for (int j = 0; item[j] != '\0' && all[i].owner[j] != '\0'; j++) {  if (item[j] != all[i].owner[j]) {  flag = false;  }  }  if (flag) {  return i;  }  }  return -1;  }  void deleteElement(Cars all[20], int& size, int nomer) {  for (int i = nomer; i < size - 1; i++) {  all[i] = all[i + 1];  }  size--;  }  int main() {  setlocale(LC\_ALL, "rus");  Cars all[20];  int size = 0;  const char filename[] = "Var10.txt";  readFile(all, size, filename);  int choice;  do {  cout << "\n1.Ввод данных с клавиатуры и запись в массив\n";  cout << "2.Вывод данных из массива\n";  cout << "3.Перенос данных из массива в файл\n";  cout << "4.Вывод данных из файла\n";  cout << "5.Поиск по владельцу\n";  cout << "6.Удаление данных из массива\n";  cout << "0.Выход из программы\n\n";  cout << "Введите номер операции: ";  cin >> choice;  cin.ignore();  int ans = 0;  switch (choice) {  case 1: {  readMas(all, size);  break;  }  case 2: {  writeMas(all, size);  break;  }  case 3:  writeFile(all, size, filename);  break;  case 4: {  readFile(all, size, filename);  writeMas(all, size);  break;  }  case 5: {  cout << "Введите владельца\n";  char buf[256];  cin.getline(buf, 256);  int ans = searchMas(all, size, buf);  if (ans == -1) {  cout << "Элемент не найден\n";  }  else {  cout << "Элемент номер: " << ans + 1 << '\n';  }  break;  }  case 6: {  cout << "Введите номер элемента для удаления: ";  int k;  cin >> k;  deleteElement(all, size, k - 1);  cout << "Элемент удалён\n";  break;  }  case 0:  break;  default: {  cout << "Неверный ввод\n";  break;  }  }  } while (choice != 0);  return 0;  } | |

Допы

|  |  |
| --- | --- |
| 1 | |
| Задание | Код/результат |
| 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. | #include <iostream>  #include <iomanip>  #include <string>  using namespace std;  struct Student {  string fullName;  int numExams;  int grades[10];  };  bool allExamsPassedOnGoodGrades(const Student& student) {  for (int i = 0; i < student.numExams; i++) {  if (student.grades[i] < 4) {  return false;  }  }  return true;  }  void calculatePerformance(Student students[], int size) {  int passedCount = 0;  for (int i = 0; i < size; i++) {  if (allExamsPassedOnGoodGrades(students[i])) {  passedCount++;  }  }  double performance = (double(passedCount) / size) \* 100;  cout << fixed << setprecision(2);  cout << "Процент студентов, сдавших все экзамены на 4 и 5: " << performance << "%" << endl;  }  void displayStudentInfo(const Student& student) {  cout << setw(30) << student.fullName << setw(10) << student.numExams;  for (int i = 0; i < student.numExams; i++) {  cout << setw(5) << student.grades[i];  }  cout << endl;  }  void displayAllStudents(Student students[], int size) {  cout << setw(30) << "ФИО студента" << setw(10) << "Экзамены" << setw(15) << "Оценки" << endl;  for (int i = 0; i < size; i++) {  displayStudentInfo(students[i]);  }  }  int main() {  setlocale(LC\_ALL, "rus");  int n;  cout << "Введите количество студентов: ";  cin >> n;  cin.ignore();  Student \*students=new Student[n];  for (int i = 0; i < n; i++) {  cout << "\nВведите ФИО студента " << i + 1 << ": ";  getline(cin, students[i].fullName);  cout << "Введите количество экзаменов: ";  cin >> students[i].numExams;  cout << "Введите оценки за экзамены: ";  for (int j = 0; j < students[i].numExams; j++) {  cin >> students[i].grades[j];  }  cin.ignore();  }  displayAllStudents(students, n);  calculatePerformance(students, n);  delete[] students;  return 0;  } |

|  |  |
| --- | --- |
| №2 | |
| Задание | Код/результат |
| 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). | #include <iostream>  #include <iomanip>  #define N 8  using namespace std;  struct TRAIN {  char where[20];  int nomer;  int when;  char whenTime[20];  };  bool compStr(char a[], char b[]) {  int i=0;  for (; a[i] != '\0' && b[i] != '\0'; i++) {  if (tolower(a[i]) < tolower(b[i])) {  return true;  }  if (tolower(a[i]) > tolower(b[i])) {  return false;  }  }  if (a[i] == '\0') {  return true;  }  return false;  }  void sortTrain(TRAIN all[]) {  for (int i = 0; i < N; i++) {  for (int j = i; j < N; j++) {  if (!compStr(all[i].where, all[j].where)) {  TRAIN temp = all[i];  all[i] = all[j];  all[j] = temp;  }  }  }  }  int main() {  TRAIN all[N];  for (int i = 0; i < N; i++) {  cin >> all[i].where >> all[i].nomer;  cin >> all[i].whenTime;  char temp[20] = "";  strcpy\_s(temp, all[i].whenTime);  all[i].when = ((temp[0] - '0') \* 10 + (temp[1] - '0')) \* 60 + (temp[3] - '0') \* 10 + (temp[4] - '0');  }  cout << '\n';  sortTrain(all);  for (int i = 0; i < N; i++) {  cout<<left <<setw(20)<< all[i].where << setw(15) << all[i].nomer << setw(10) << all[i].whenTime << '\n';  }  cout << '\n';  char time[15];  cin >> time;  int min = ((time[0] - '0') \* 10 + (time[1] - '0')) \* 60 + (time[3] - '0') \* 10 + (time[4] - '0');  for (int i = 0; i < N; i++) {  if (all[i].when - min > 0) {  cout << left << setw(20) << all[i].where << setw(15) << all[i].nomer << setw(10) << all[i].whenTime << '\n';  }  }  return 0;  } |
| №3 | |
| Задание | Код/результат |
| 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. | #include <iostream>  #include <iomanip>  #define N 13  using namespace std;  struct SANIT {  char name[25];  char where[20];  char profil[20];  int col;  };  bool compStr(char a[], char b[]) {  int i = 0;  for (; a[i] != '\0' && b[i] != '\0'; i++) {  if (tolower(a[i]) < tolower(b[i])) {  return true;  }  if (tolower(a[i]) > tolower(b[i])) {  return false;  }  }  if (a[i] == '\0') {  return true;  }  return false;  }  void sortProfil(SANIT all[]) {  for (int i = 0; i < N; i++) {  for (int j = i; j < N; j++) {  if (!compStr(all[i].profil, all[j].profil)) {  SANIT temp = all[i];  all[i] = all[j];  all[j] = temp;  }  }  }  }  void sortName(SANIT all[]) {  int start = 0;  for (int k = 1; k <= N; k++) {  if (k == N) {  for (int j = start; j < k; j++) {  for (int i = j; i < k; i++) {  if (compStr(all[i].where, all[j].where)) {  SANIT temp = all[i];  all[i] = all[j];  all[j] = temp;  }  }  }  }  else {  if (strcmp(all[k].profil, all[k - 1].profil)!=0) {  for (int j = start; j < k; j++) {  for (int i = j; i < k; i++) {  if (compStr(all[i].where, all[j].where)) {  SANIT temp = all[i];  all[i] = all[j];  all[j] = temp;  }  }  }  start = k;  }    }  }  }  int main() {  SANIT all[N];  for (int i = 0; i < N; i++) {  cin >> all[i].name >> all[i].where >> all[i].profil >> all[i].col;  }  cout << '\n';  sortProfil(all);  for (int i = 0; i < N; i++) {  cout << left << setw(20) << all[i].name << setw(20) << all[i].where << setw(20) << all[i].profil << setw(7) << all[i].col << '\n';  }  cout << '\n';  sortName(all);  for (int i = 0; i < N; i++) {  cout << left << setw(20) << all[i].name << setw(20) << all[i].where << setw(20) << all[i].profil << setw(7) << all[i].col << '\n';  }  return 0;  } |