**ОСНОВЫ АЛГОРИТМИЗАЦИИ И ПРОГРАММИРОВАНИЯ**

ОТЧЁТ ПО ЛАБОРАТОРНЫМ РАБОТАМ

*Щуплленков Илья Викторович*

*ФИТ, 1 курс, ДЭиВИ-9*

**Лабораторная работа № 1.** [**Рекурсивные алгоритмы**](#Лаб1)

**Вариант 15**

|  |
| --- |
| **Пояснение к алгоритму** |
|  |

**Код:**

#include <iostream>

using namespace std;

int S(int x)

{

if (x > 100)

return x + 10;

else

return S(S(x + 4));

}

void main()

{

int x;

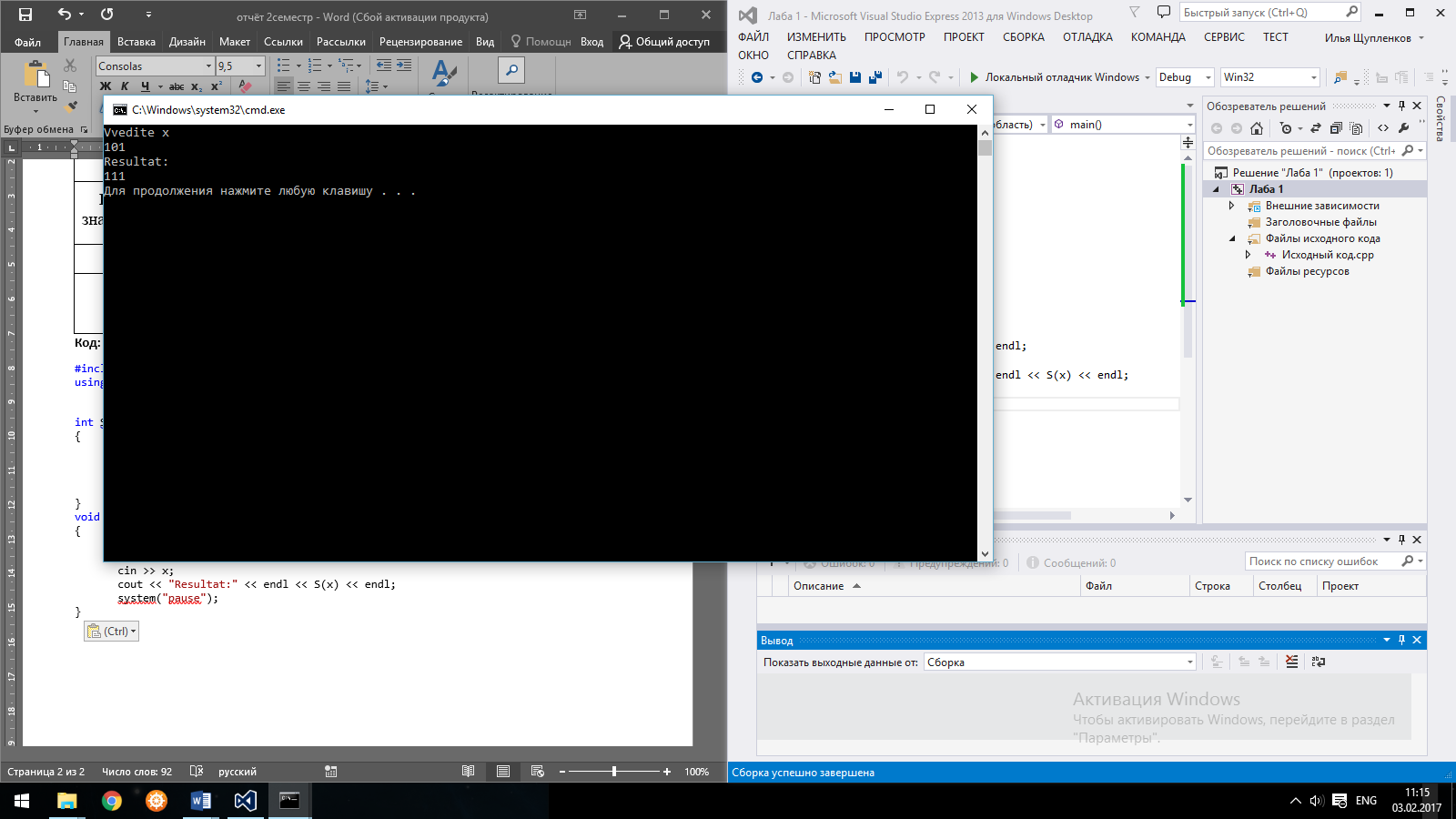
cout << "Vvedite x" << endl;

cin >> x;

cout << "Resultat:" << endl << S(x) << endl;

system("pause");

}



**Вариант 1**

|  |
| --- |
| **Пояснение к алгоритму** |
|  |

**Код:**

#include <iostream>

using namespace std;

int P(int m, int n);

int main()

{

setlocale(LC\_ALL, "rus");

int a = 0,m,n;

cout << "Введите m = "; cin >> m;

cout << "Введите n = "; cin >> n;

a = P(m, n);

cout << "Ответ = "<< a <<endl;

system("pause");

}

int P(int m, int n)

{

if (m == 1 | n == 1)

return 1;

else

if (n > m)

return P(m, m);

else

if (m == n)

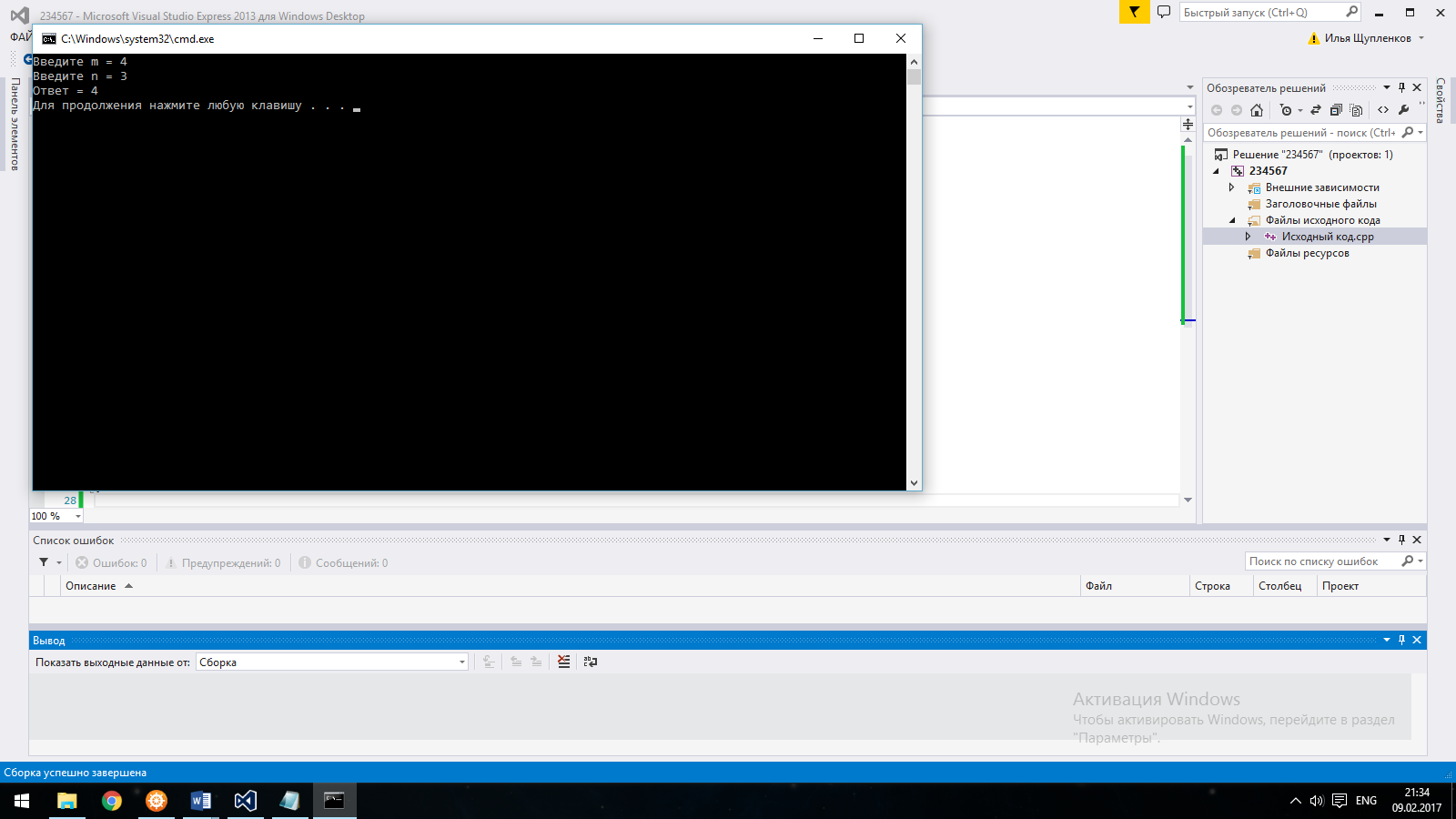
return P(m, m - 1) + 1;

else

if (n<m)

return P(m, n - 1) + P(m - n, n);

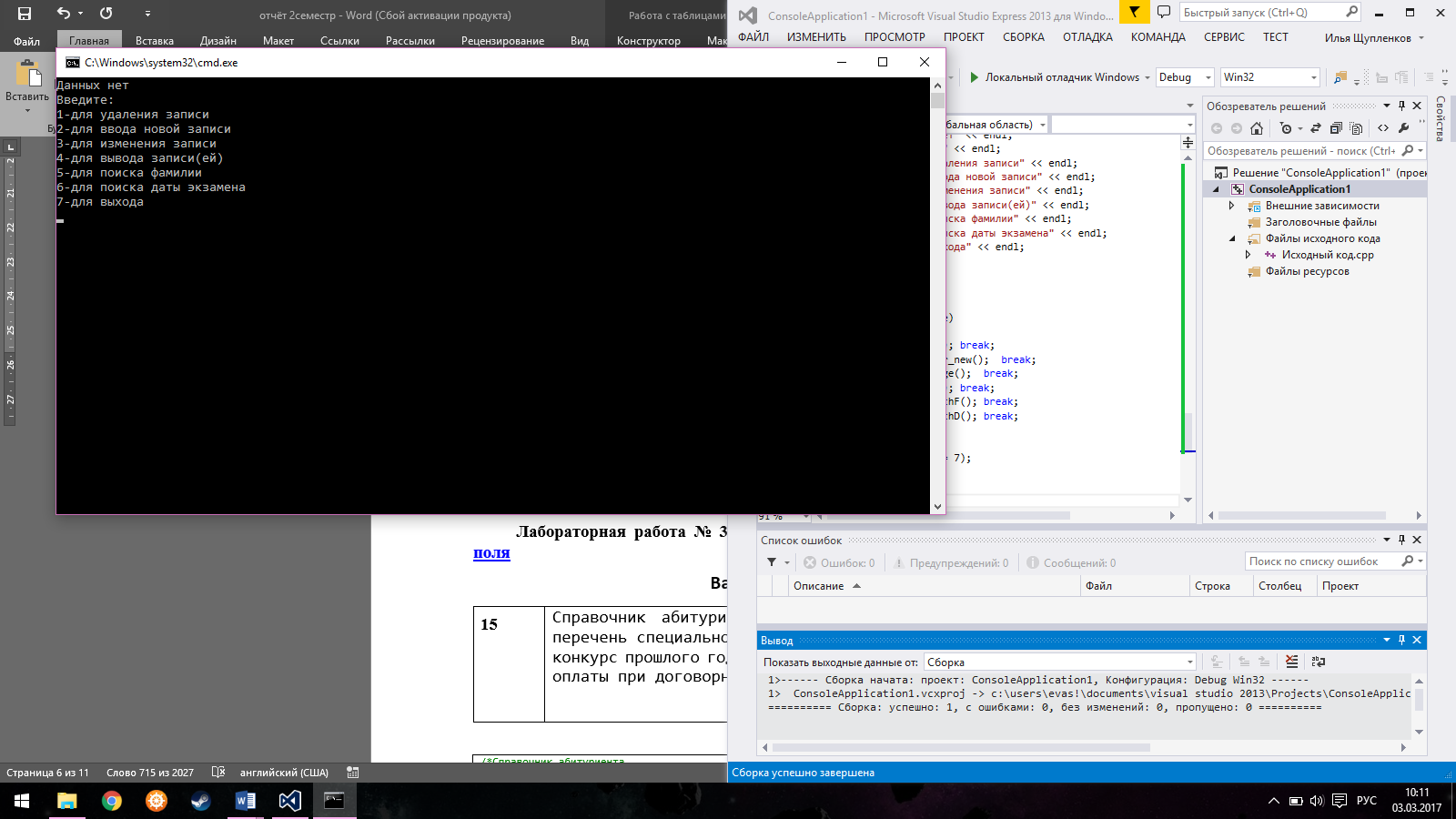
}



**Лабораторная работа № 2.** [**Представление информации в виде структуры**](#Лаб2)

**Вариант 15**

|  |  |
| --- | --- |
| **15** | **Преподаватели**. Фамилия преподавателя, название экзамена, дата экзамена. Выбор по фамилии, дате экзамена. |
| # include <iostream>  # include <windows.h>  # include <cstring>  # define str\_len 30  # define size 30  int choice;  using namespace std;  struct Student  {  char name[str\_len];  char year\_of\_birth[str\_len];  char department[5];  };  struct Student list\_of\_student[size];  struct Student bad;  int current\_size = 0;  void enter\_new()  {  setlocale(LC\_CTYPE, "rus");  cout << "Ввод информации" << endl;  if (current\_size < size)  {  cout << "Строка номер ";  cout << current\_size + 1;  cout << endl << "Фамилия преподавателя" << endl;  cin >> list\_of\_student[current\_size].name;  cout << "Название экзамена" << endl;  cin >> list\_of\_student[current\_size].department;  cout << "Дата экзамена" << endl;  cin >> list\_of\_student[current\_size].year\_of\_birth;  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 преподавателя" << endl;  cout << "2-для изменения названия экзамена" << endl;  cout << "3-для изменения даты экзамена" << endl;  cout << "4-конец\n";  cin >> per;  switch (per)  {  case 1: cout << "Новая фамилия преподавателя";  cin >> list\_of\_student[n - 1].name; break;  case 2: cout << "Новое название экзамена";  cin >> list\_of\_student[n - 1].department; break;  case 3: cout << "Новая дата экзамена";  cin >> list\_of\_student[n - 1].year\_of\_birth; 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 преподавателя";  cout << list\_of\_student[n - 1].name << endl;  cout << "Название экзамена";  cout << list\_of\_student[n - 1].department << endl;  cout << "Дата экзамена";  cout << list\_of\_student[n - 1].year\_of\_birth << endl;  }  if (sw == 2)  {  for (int i = 0; i < current\_size; i++)  {  cout << "Фамилия преподавателя";  cout << list\_of\_student[i].name << endl;  cout << "название экзамена";  cout << list\_of\_student[i].department << endl;  cout << "Дата экзамена";  cout << list\_of\_student[i].year\_of\_birth << endl;  }  }  cout << "Что дальше?" << endl;  cin >> choice;  }  void searchF()  {  setlocale(LC\_CTYPE, "rus");  char Fname[str\_len];  cout << "Для поиска фамиллии, введите её - " << endl;  cin >> Fname;  for (int i = 0; i < size; i++)  {  if (!strcmp(Fname, list\_of\_student[i].name))  {  cout << "Фамилия преподавателя: ";  cout << list\_of\_student[i].name << endl;  cout << "Название экзамена: ";  cout << list\_of\_student[i].department << endl;  cout << "Дата экзамена: ";  cout << list\_of\_student[i].year\_of\_birth << endl;  }  else if (i == size - 1) cout << "Пустовато ";  }  cout << "что дальше?" << endl;  cin >> choice;  }  void searchD()  {  setlocale(LC\_CTYPE, "rus");  char Dname[str\_len];  cout << "Для поиска даты, введите её - " << endl;  cin >> Dname;  for (int i = 0; i < size; i++)  {  if (!strcmp(Dname, list\_of\_student[i].year\_of\_birth))  {  cout << "Фамилия преподавателя: ";  cout << list\_of\_student[i].name << endl;  cout << "Название экзамена: ";  cout << list\_of\_student[i].department << endl;  cout << "Дата экзамена: ";  cout << list\_of\_student[i].year\_of\_birth << endl;  }  else if (i == size - 1) cout << "Пустовато ";  }  cout << "что дальше?" << endl;  cin >> choice;  }  int main()  {  setlocale(LC\_CTYPE, "Russian");  cout << "Данных нет" << endl;  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 поиска фамилии" << endl;  cout << "6-для поиска даты экзамена" << endl;  cout << "7-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;  case 5: searchF(); break;  case 6: searchD(); break;  }  } while (choice != 7);  } | |

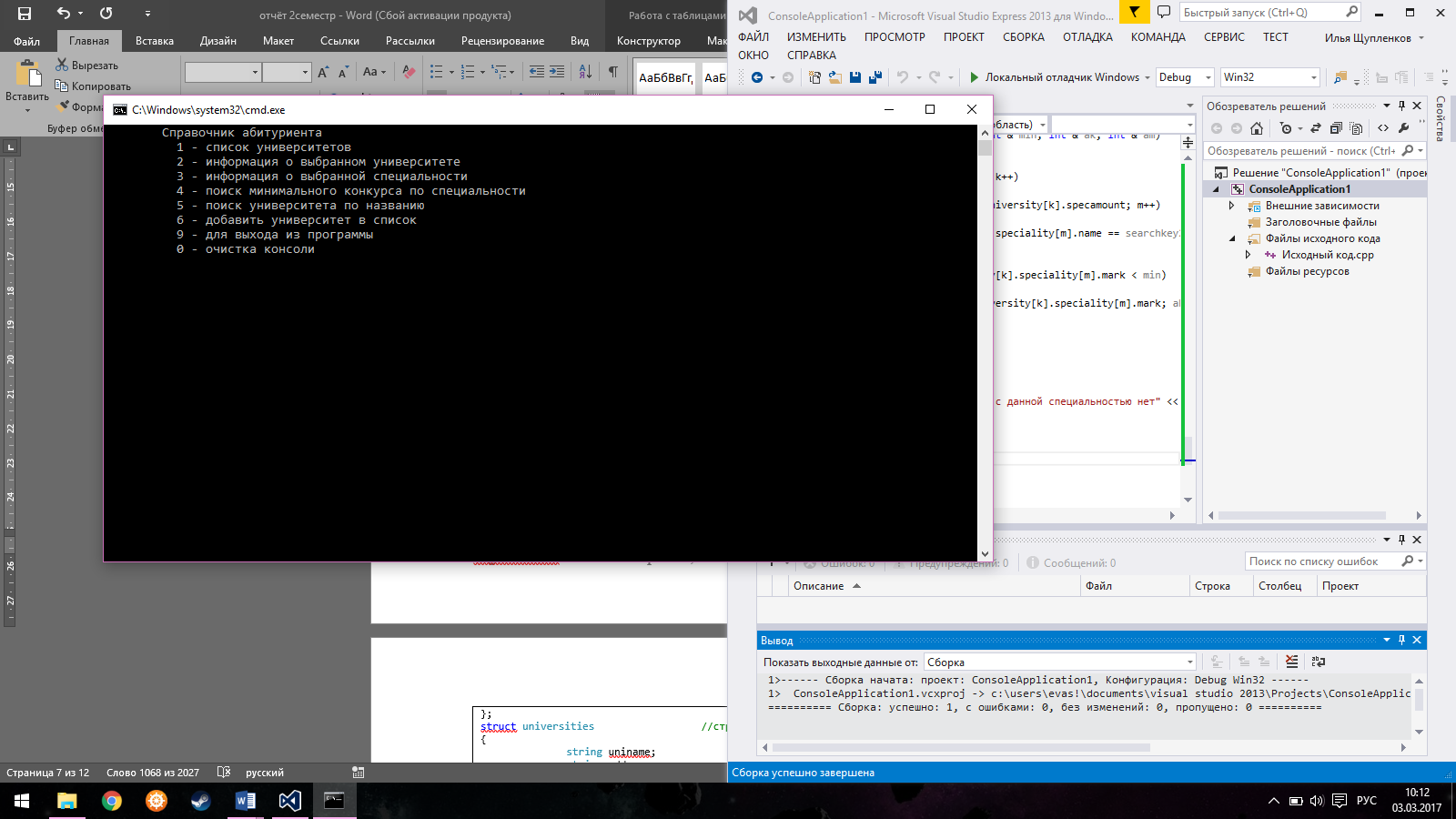


**Лабораторная работа № 3.** [**Объединения, перечисления, битовые поля**](#Лаб3)

**Вариант 15**

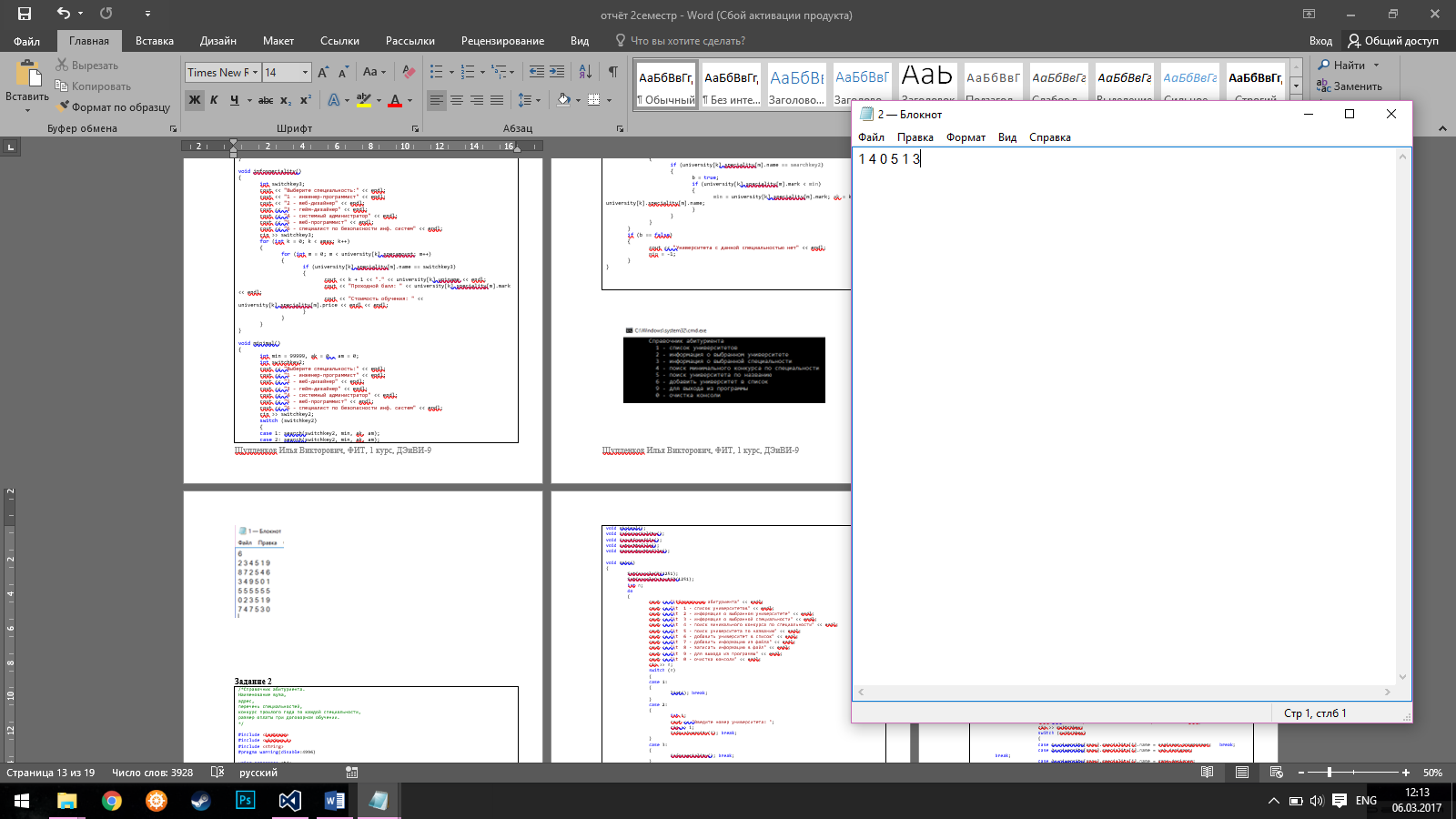
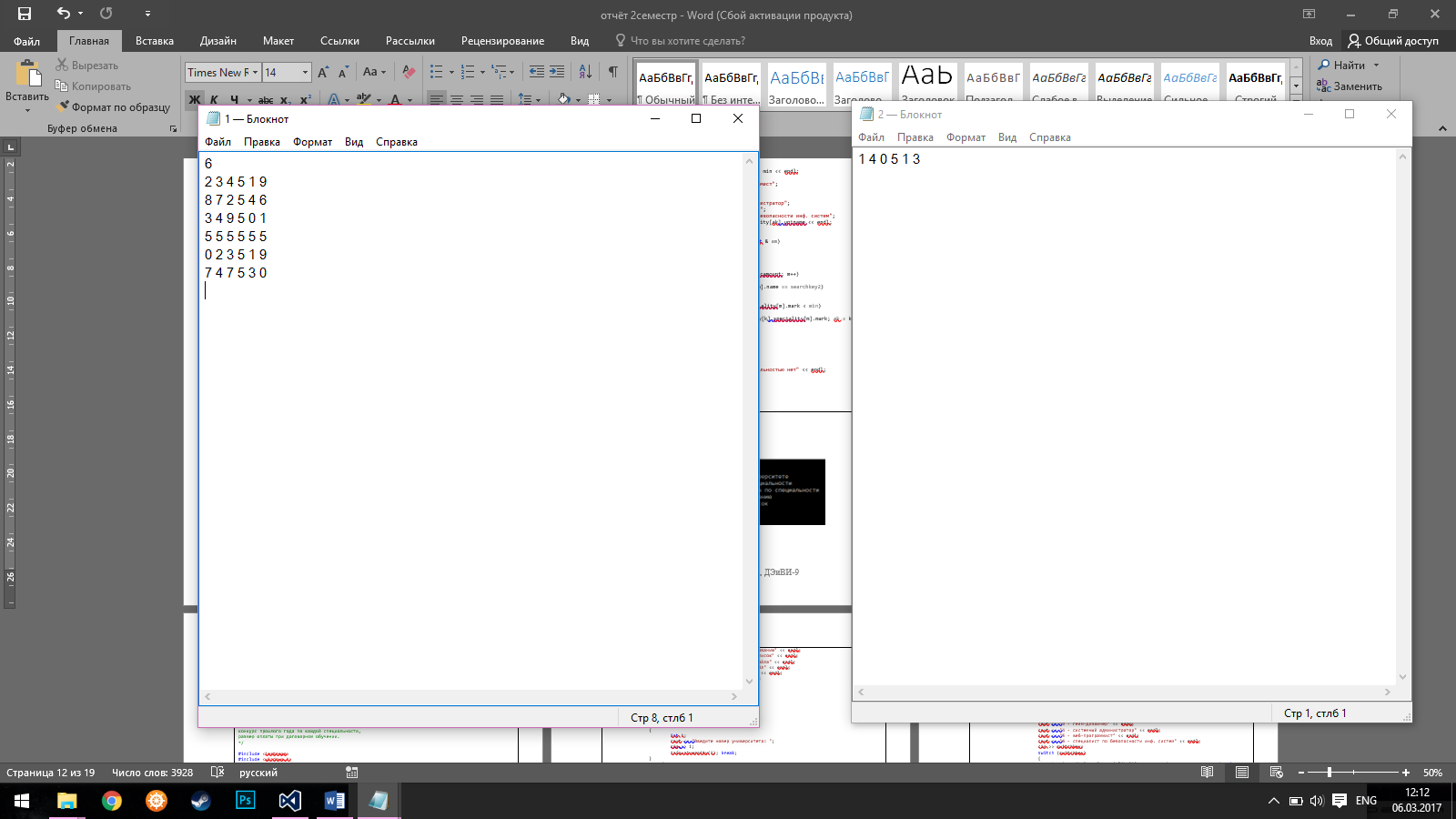
|  |  |
| --- | --- |
| **15** | Справочник абитуриента. Наименование вуза, адрес, перечень специальностей,  конкурс прошлого года по каждой специальности, размер оплаты при договорном обучении. |

|  |
| --- |
| /\*Справочник абитуриента.  Наименование вуза,  адрес,  перечень специальностей,  конкурс прошлого года по каждой специальности,  размер оплаты при договорном обучении.  \*/  #include <iostream>  #include <windows.h>  #include <string>  using namespace std;  const int max = 100;  int amax = 0;  enum specialities //перечисление  {  engineer\_programmer = 1,  web\_designer,  game\_designer,  system\_administrator,  web\_programmer,  security\_specialist  };  struct forspecialities //структура специальностей  {  int name;  unsigned mark : 9;  int price;  };  struct universities //структура универов  {  string uniname;  string address;  forspecialities speciality[30];  int specamount;  } university[max], cleaner;  void input();  void list();  void search(int switchkey2, int & min, int & ak, int & am);  void infouniversity(int i);  void searchUniname();  void minimal();  void infospeciality();  void main()  {  SetConsoleCP(1251);  SetConsoleOutputCP(1251);  int n;  do  {  cout << "\tСправочник абитуриента" << endl;  cout << "\t 1 - список университетов" << endl;  cout << "\t 2 - информация о выбранном университете" << endl;  cout << "\t 3 - информация о выбранной специальности" << endl;  cout << "\t 4 - поиск минимального конкурса по специальности" << endl;  cout << "\t 5 - поиск университета по названию" << endl;  cout << "\t 6 - добавить университет в список" << endl;  cout << "\t 9 - для выхода из программы" << endl;  cout << "\t 0 - очистка консоли" << endl;  cin >> n;  switch (n)  {  case 1:  {  list(); break;  }  case 2:  {  int i;  cout << "Введите номер университета: ";  cin >> i;  infouniversity(i); break;  }  case 3:  {  infospeciality(); break;  }  case 4:  {  minimal(); break;  }  case 5:  {  searchUniname(); break;  }  case 6:  {  input(); break;  }  case 0:  system("cls"); break;  }  } while (n != 9);  }  void input()  {  if (amax < max)  {  cin.clear();  while (cin.get() != '\n');  cout << "Введите название университета: " << endl;  getline(cin,university[amax].uniname);  cout << "Введите адрес университета: " << endl;  getline(cin,university[amax].address);  int specamount;  int switchkey;  int mark;  cout << "Введите количество специальностей: ";  cin >> specamount;  university[amax].specamount = specamount;  for (int i = 0; i < specamount; i++)  {  cout << "Введите номер специальности: " << endl;  cout << "1 - инженер-программист" << endl;  cout << "2 - веб-дизайнер" << endl;  cout << "3 - гейм-дизайнер" << endl;  cout << "4 - системный администратор" << endl;  cout << "5 - веб-программист" << endl;  cout << "6 - специалист по безопасности инф. систем" << endl;  cin >> switchkey;  switch (switchkey)  {  case 1:university[amax].speciality[i].name = engineer\_programmer; break;  case 2:university[amax].speciality[i].name = web\_designer; break;  case 3:university[amax].speciality[i].name = game\_designer; break;  case 4:university[amax].speciality[i].name = system\_administrator; break;  case 5:university[amax].speciality[i].name = web\_programmer; break;  case 6:university[amax].speciality[i].name = security\_specialist; break;  }  cout << "Введите проходной балл по данной специальности: ";  cin >> mark;  university[amax].speciality[i].mark = mark;  cout << "Введите стоимость обучения по данной специальности: ";  cin >> university[amax].speciality[i].price;  }  amax++;  }  else  cout << "ошибка" << endl;  }  void list()  {  for (int i = 0; i < amax; i++)  {  cout << i + 1 << ". " << university[i].uniname << endl;  }  }  void infouniversity(int i)  {  i--;  if (i < 0 || i > amax - 1) cout << "Выход за границы" << endl;  else  {  cout << "Название университета: " << university[i].uniname << endl;  cout << "Адрес: " << university[i].address << endl;  cout << "Специальности:" << endl;  for (int k = 0; k < university[i].specamount; k++)  {  if (university[i].speciality[k].name == 1) cout << "\tинженер-программист" << endl;  if (university[i].speciality[k].name == 2) cout << "\tвеб-дизайнер" << endl;  if (university[i].speciality[k].name == 3) cout << "\tгейм-дизайнер" << endl;  if (university[i].speciality[k].name == 4) cout << "\tсистемный администратор" << endl;  if (university[i].speciality[k].name == 5) cout << "\tвеб-программист" << endl;  if (university[i].speciality[k].name == 6) cout << "\tспециалист по безопасности инф. систем" << endl;  }  }  cout << endl;  }  void searchUniname()  {  cin.clear();  while (cin.get() != '\n');  string searchline;  bool a = false;  cout << "Введите название университета: ";  getline(cin, searchline);  for (int i = 0; i <= amax; i++)  {  if (searchline == university[i].uniname)  {  cout << "Название университета: " << university[i].uniname << endl;  cout << "Адрес: " << university[i].address << endl;  a = true;  }  }  if (a == false) cout << "Не найдено" << endl;  }  void infospeciality()  {  int switchkey3;  cout << "Выберите специальность:" << endl;  cout << "1 - инженер-программист" << endl;  cout << "2 - веб-дизайнер" << endl;  cout << "3 - гейм-дизайнер" << endl;  cout << "4 - системный администратор" << endl;  cout << "5 - веб-программист" << endl;  cout << "6 - специалист по безопасности инф. систем" << endl;  cin >> switchkey3;  for (int k = 0; k < amax; k++)  {  for (int m = 0; m < university[k].specamount; m++)  {  if (university[k].speciality[m].name == switchkey3)  {  cout << k + 1 << "." << university[k].uniname << endl;  cout << "Проходной балл: " << university[k].speciality[m].mark << endl;  cout << "Стоимость обучения: " << university[k].speciality[m].price << endl << endl;  }  }  }  }  void minimal()  {  int min = 99999, ak = 0 , am = 0;  int switchkey2;  cout << "Выберите специальность:" << endl;  cout << "1 - инженер-программист" << endl;  cout << "2 - веб-дизайнер" << endl;  cout << "3 - гейм-дизайнер" << endl;  cout << "4 - системный администратор" << endl;  cout << "5 - веб-программист" << endl;  cout << "6 - специалист по безопасности инф. систем" << endl;  cin >> switchkey2;  switch (switchkey2)  {  case 1: search(switchkey2, min, ak, am);  case 2: search(switchkey2, min, ak, am);  case 3: search(switchkey2, min, ak, am);  case 4: search(switchkey2, min, ak, am);  case 5: search(switchkey2, min, ak, am);  case 6: search(switchkey2, min, ak, am);  }  if (min == -1) exit(0);  cout << "Минимальный балл равен: " << min << endl;  cout << "Специальность ";  if (am == 1) cout << "инженер-программист";  if (am == 2) cout << "веб-дизайнер";  if (am == 3) cout << "гейм-дизайнер";  if (am == 4) cout << "системный администратор";  if (am == 5) cout << "веб-программист";  if (am == 6) cout << "специалист по безопасности инф. систем";  cout << " в университете " << university[ak].uniname << endl;  }  void search(int searchkey2, int & min, int & ak, int & am)  {  bool b = false;  for (int k = 0; k < amax; k++)  {  for (int m = 0; m < university[k].specamount; m++)  {  if (university[k].speciality[m].name == searchkey2)  {  b = true;  if (university[k].speciality[m].mark < min)  {  min = university[k].speciality[m].mark; ak = k; am = university[k].speciality[m].name;  }  }  }  }  if (b == false)  {  cout << "Университета с данной специальностью нет" << endl;  min = -1;  }  } |



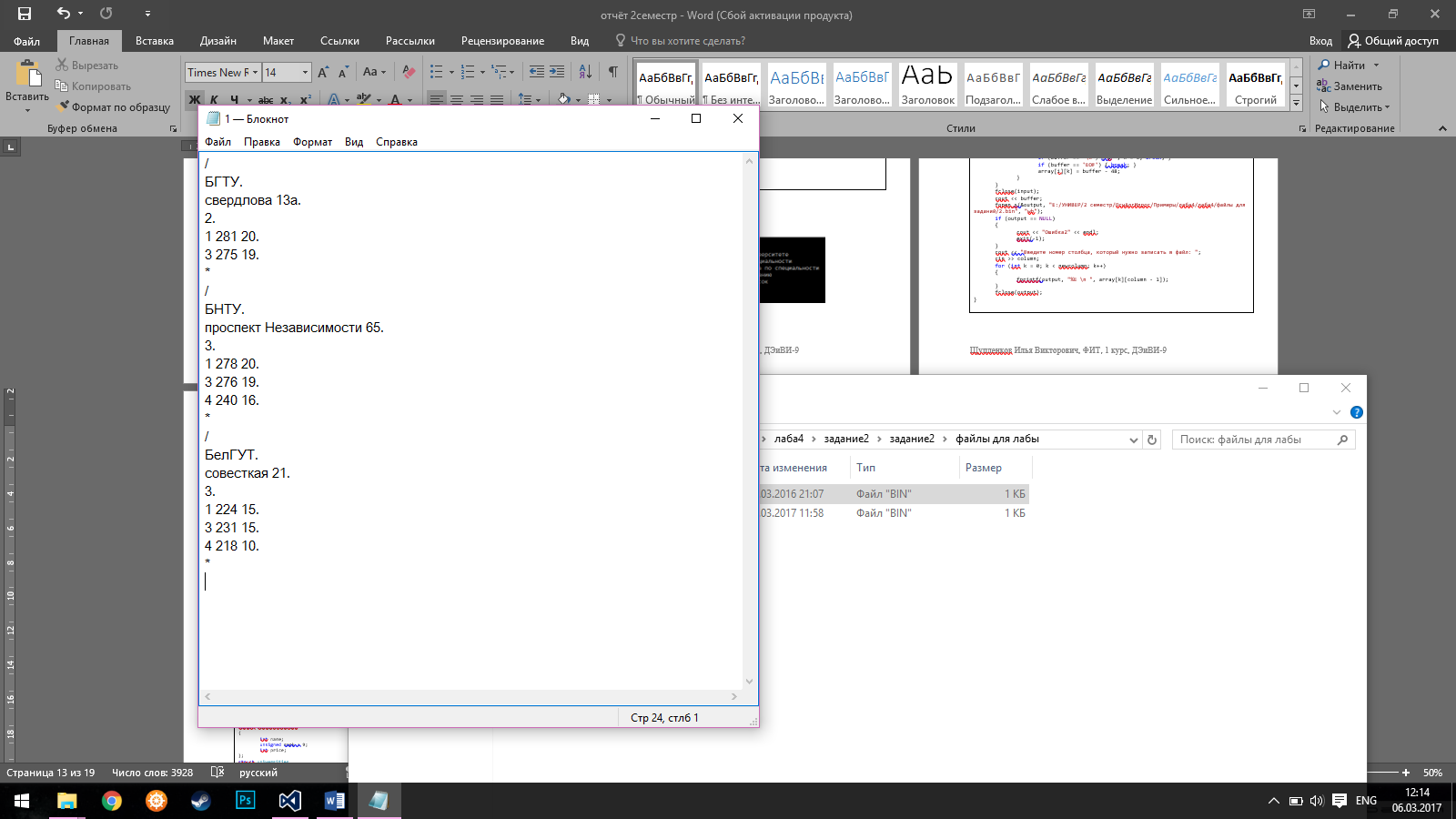
**Лабораторная работа № 4. [Работа с файлами на языке С](#Лаб4)**

|  |
| --- |
| #include <conio.h>  #include <stdio.h>  #include <stdlib.h>  #include <fstream>  #include <iostream>  using namespace std;  void main()  {  setlocale(LC\_ALL, "RUS");  FILE \*input = NULL;  FILE \*output = NULL;  int array[6][6];  char columnindex;  int newcolumn, column;  char buffer;  fopen\_s(&input, "E:/УНИВЕР/2 семестр/ОснАлгИпрог/Примеры/лаба4/лаба4/файлы для заданий/1.bin", "rb");  if (input == NULL)  {  cout << "Ошибка" << endl;  exit(-1);  }  fread(&columnindex, 1, 1, input);  newcolumn = (int)columnindex - 48;  fread(&buffer, 1, 1, input);  for (int i = 0; i < newcolumn; i++)  {  for (int k = 0; k < newcolumn; k++)  {  fread(&buffer, 1, 1, input);  if (buffer == ' ') { k--; continue; }  if (buffer == '\n') { i--; k = 0; break; }  if (buffer == 'EOF') { break; }  array[i][k] = buffer - 48;  }  }  fclose(input);  cout << buffer;  fopen\_s(&output, "E:/УНИВЕР/2 семестр/ОснАлгИпрог/Примеры/лаба4/лаба4/файлы для заданий/2.bin", "wb");  if (output == NULL)  {  cout << "Ошибка2" << endl;  exit(-1);  }  cout << "Введите номер столбца, который нужно записать в файл: ";  cin >> column;  for (int k = 0; k < newcolumn; k++)  {  fprintf(output, "%E \n ", array[k][column - 1]);  }  fclose(output);  } |



**Задание 2**

|  |
| --- |
| /\*Справочник абитуриента.  Наименование вуза,  адрес,  перечень специальностей,  конкурс прошлого года по каждой специальности,  размер оплаты при договорном обучении.  \*/  #include <iostream>  #include <windows.h>  #include <string>  #pragma warning(disable:4996)  using namespace std;  const int max = 100;  int amax = 0;  enum specialities  {  engineer\_programmer = 1,  web\_designer,  game\_designer,  system\_administrator,  web\_programmer,  security\_specialist  };  struct forspecialities  {  int name;  unsigned mark : 9;  int price;  };  struct universities  {  string uniname;  string address;  forspecialities speciality[30];  int specamount;  } university[max], cleaner;  void input();  void list();  void search(int switchkey2, int & min, int & ak, int & am);  void infouniversity(int i);  void searchUniname();  void minimal();  void infospeciality();  void inputfromfile();  void outputtofile();  void massoutputtofile();  void main()  {  SetConsoleCP(1251);  SetConsoleOutputCP(1251);  int n;  do  {  cout << "\tСправочник абитуриента" << endl;  cout << "\t 1 - список университетов" << endl;  cout << "\t 2 - информация о выбранном университете" << endl;  cout << "\t 3 - информация о выбранной специальности" << endl;  cout << "\t 4 - поиск минимального конкурса по специальности" << endl;  cout << "\t 5 - поиск университета по названию" << endl;  cout << "\t 6 - добавить университет в список" << endl;  cout << "\t 7 - добавить информацию из файла" << endl;  cout << "\t 8 - записать информацию в файл" << endl;  cout << "\t 9 - для выхода из программы" << endl;  cout << "\t 0 - очистка консоли" << endl;  cin >> n;  switch (n)  {  case 1:  {  list(); break;  }  case 2:  {  int i;  cout << "Введите номер университета: ";  cin >> i;  infouniversity(i); break;  }  case 3:  {  infospeciality(); break;  }  case 4:  {  minimal(); break;  }  case 5:  {  searchUniname(); break;  }  case 6:  {  input(); break;  }  case 7:  {  inputfromfile(); break;  }  case 8:  {  int switchkey21;  do  {  cout << "1 - вывести информацию об одном университете" << endl;  cout << "2 - вывести информацию о всех университетах" << endl;  cout << "3 - отмена" << endl;  cin >> switchkey21;  switch (switchkey21)  {  case 1:  {  outputtofile(); switchkey21 = 3; break;  }  case 2:  {  massoutputtofile(); switchkey21 = 3; break;  }  }  } while (switchkey21 != 3);  }  case 0:  system("cls"); break;  }  } while (n != 9);  }  void input()  {  if (amax < max)  {  cin.clear();  while (cin.get() != '\n');  cout << "Введите название университета: " << endl;  getline(cin, university[amax].uniname);  cout << "Введите адрес университета: " << endl;  getline(cin, university[amax].address);  int specamount;  int switchkey;  int mark;  cout << "Введите количество специальностей: ";  cin >> specamount;  university[amax].specamount = specamount;  for (int i = 0; i < specamount; i++)  {  cout << "Введите номер специальности: " << endl;  cout << "1 - инженер-программист" << endl;  cout << "2 - веб-дизайнер" << endl;  cout << "3 - гейм-дизайнер" << endl;  cout << "4 - системный администратор" << endl;  cout << "5 - веб-программист" << endl;  cout << "6 - специалист по безопасности инф. систем" << endl;  cin >> switchkey;  switch (switchkey)  {  case 1:university[amax].speciality[i].name = engineer\_programmer; break;  case 2:university[amax].speciality[i].name = web\_designer; break;  case 3:university[amax].speciality[i].name = game\_designer; break;  case 4:university[amax].speciality[i].name = system\_administrator; break;  case 5:university[amax].speciality[i].name = web\_programmer; break;  case 6:university[amax].speciality[i].name = security\_specialist; break;  }  cout << "Введите проходной балл по данной специальности: ";  cin >> mark;  university[amax].speciality[i].mark = mark;  cout << "Введите стоимость обучения по данной специальности: ";  cin >> university[amax].speciality[i].price;  }  amax++;  }  else  cout << "ошибка" << endl;  }  void list()  {  for (int i = 0; i < amax; i++)  {  cout << i + 1 << ". " << university[i].uniname << endl;  }  }  void infouniversity(int i)  {  i--;  if (i < 0 || i > amax - 1) cout << "Выход за границы" << endl;  else  {  cout << "Название университета: " << university[i].uniname << endl;  cout << "Адрес: " << university[i].address << endl;  cout << "Специальности:" << endl;  for (int k = 0; k < university[i].specamount; k++)  {  if (university[i].speciality[k].name == 1) cout << "\tинженер-программист" << endl;  if (university[i].speciality[k].name == 2) cout << "\tвеб-дизайнер" << endl;  if (university[i].speciality[k].name == 3) cout << "\tгейм-дизайнер" << endl;  if (university[i].speciality[k].name == 4) cout << "\tсистемный администратор" << endl;  if (university[i].speciality[k].name == 5) cout << "\tвеб-программист" << endl;  if (university[i].speciality[k].name == 6) cout << "\tспециалист по безопасности инф. систем" << endl;  }  }  cout << endl;  }  void searchUniname()  {  cin.clear();  while (cin.get() != '\n');  string searchline;  bool a = false;  cout << "Введите название университета: ";  getline(cin, searchline);  for (int i = 0; i <= amax; i++)  {  if (searchline == university[i].uniname)  {  cout << "Название университета: " << university[i].uniname << endl;  cout << "Адрес: " << university[i].address << endl;  a = true;  }  }  if (a == false) cout << "Совпадений нет" << endl;  }  void infospeciality()  {  int switchkey3;  cout << "Выберите специальность:" << endl;  cout << "1 - инженер-программист" << endl;  cout << "2 - веб-дизайнер" << endl;  cout << "3 - гейм-дизайнер" << endl;  cout << "4 - системный администратор" << endl;  cout << "5 - веб-программист" << endl;  cout << "6 - специалист по безопасности инф. систем" << endl;  cin >> switchkey3;  for (int k = 0; k < amax; k++)  {  for (int m = 0; m < university[k].specamount; m++)  {  if (university[k].speciality[m].name == switchkey3)  {  cout << k + 1 << "." << university[k].uniname << endl;  cout << "Проходной балл: " << university[k].speciality[m].mark << endl;  cout << "Стоимость обучения: " << university[k].speciality[m].price << endl << endl;  }  }  }  }  void minimal()  {  int min = 99999, ak = 0, am = 0;  int switchkey2;  cout << "Выберите специальность:" << endl;  cout << "1 - инженер-программист" << endl;  cout << "2 - веб-дизайнер" << endl;  cout << "3 - гейм-дизайнер" << endl;  cout << "4 - системный администратор" << endl;  cout << "5 - веб-программист" << endl;  cout << "6 - специалист по безопасности инф. систем" << endl;  cin >> switchkey2;  switch (switchkey2)  {  case 1: search(switchkey2, min, ak, am);  case 2: search(switchkey2, min, ak, am);  case 3: search(switchkey2, min, ak, am);  case 4: search(switchkey2, min, ak, am);  case 5: search(switchkey2, min, ak, am);  case 6: search(switchkey2, min, ak, am);  }  if (min == -1) exit(0);  cout << "Минимальный балл равен: " << min << endl;  cout << "Специальность ";  if (am == 1) cout << "инженер-программист";  if (am == 2) cout << "веб-дизайнер";  if (am == 3) cout << "гейм-дизайнер";  if (am == 4) cout << "системный администратор";  if (am == 5) cout << "веб-программист";  if (am == 6) cout << "специалист по безопасности инф. систем";  cout << " в университете " << university[ak].uniname << endl;  }  void search(int searchkey2, int & min, int & ak, int & am)  {  bool b = false;  for (int k = 0; k < amax; k++)  {  for (int m = 0; m < university[k].specamount; m++)  {  if (university[k].speciality[m].name == searchkey2)  {  b = true;  if (university[k].speciality[m].mark < min)  {  min = university[k].speciality[m].mark; ak = k; am = university[k].speciality[m].name;  }  }  }  }  if (b == false)  {  cout << "Университета с данной специальностью нет" << endl;  min = -1;  }  }  void inputfromfile()  {  char buffer = NULL;  FILE \*input = NULL;  fopen\_s(&input, "E:/УНИВЕР/2 семестр/ОснАлгИпрог/Примеры/лаба4/задание2/задание2/файлы для лабы/1.bin", "rb");  if (input == NULL)  {  cout << "Ошибка" << endl;  exit(-1);  }  while (buffer != '&')  {  fread(&buffer, 1, 1, input);  if (buffer == '/')  {  fread(&buffer, 1, 1, input);  while (buffer != '.') // считывание названия университета  {  if (buffer == '\r') { fread(&buffer, 1, 1, input); continue; }  fread(&buffer, 1, 1, input);  if (buffer == '.') break;  university[amax].uniname += buffer;  } if (buffer == '.') { fread(&buffer, 1, 1, input); fread(&buffer, 1, 1, input); }  while (buffer != '\r') // считывание адреса университета  {  fread(&buffer, 1, 1, input);  if (buffer == '.') break;  university[amax].address += buffer;  } fread(&buffer, 1, 1, input); fread(&buffer, 1, 1, input); fread(&buffer, 1, 1, input); // сдвиг на specamount  buffer = buffer - 48; university[amax].specamount = buffer;  fread(&buffer, 1, 1, input); fread(&buffer, 1, 1, input); // сдвиг на '\n' перед первой специальностью  for (int i = 0; i < university[amax].specamount; i++)  {  int bufferint;  fscanf\_s(input, "%d", &bufferint);  if (bufferint == 1) university[amax].speciality[i].name = engineer\_programmer;  if (bufferint == 2) university[amax].speciality[i].name = web\_designer;  if (bufferint == 3) university[amax].speciality[i].name = game\_designer;  if (bufferint == 4) university[amax].speciality[i].name = system\_administrator;  if (bufferint == 5) university[amax].speciality[i].name = web\_programmer;  if (bufferint == 6) university[amax].speciality[i].name = security\_specialist;  fread(&buffer, 1, 1, input); // сдвиг на ' ' перед mark  fscanf\_s(input, "%d", &bufferint); university[amax].speciality[i].mark = bufferint;  fread(&buffer, 1, 1, input); // сдвиг на ' ' перед price  fscanf\_s(input, "%d", &bufferint); university[amax].speciality[i].price = bufferint;  fread(&buffer, 1, 1, input); fread(&buffer, 1, 1, input);  }  while (buffer != '\*')  {  fread(&buffer, 1, 1, input);  } amax++;  }  }  fclose(input);  }  void massoutputtofile()  {  char buffer;  FILE \*output = NULL;  fopen\_s(&output, "E:/УНИВЕР/2 семестр/ОснАлгИпрог/Примеры/лаба4/задание2/задание2/файлы для лабы/2.bin", "wb");  if (input == NULL)  {  cout << "Ошибка" << endl;  exit(-1);  }  fprintf(output, "\tСправочник абитуриента\n");  for (int i = 0; i < amax; i++)  {  char \*strbuffer1 = new char[university[i].uniname.length() + 1];  char \*strbuffer2 = new char[university[i].address.length() + 1];  strcpy(strbuffer1, university[i].uniname.c\_str());  fprintf(output, "================== %d ==================\n", i + 1);  fprintf(output, "Название университета: %s\n", strbuffer1);  strcpy(strbuffer2, university[i].address.c\_str());  fprintf(output, "Адрес: %s\n", strbuffer2);  fprintf(output, "Специальности: \n");  for (int k = 0; k < university[i].specamount; k++)  {  if (university[i].speciality[k].name == 1) fprintf(output, "\tинженер-программист\n");  if (university[i].speciality[k].name == 2) fprintf(output, "\tвеб-дизайнер\n");  if (university[i].speciality[k].name == 3) fprintf(output, "\tгейм-дизайнер\n");  if (university[i].speciality[k].name == 4) fprintf(output, "\tсистемный администратор\n");  if (university[i].speciality[k].name == 5) fprintf(output, "\tвеб-программист\n");  if (university[i].speciality[k].name == 6) fprintf(output, "\tспециалист по безопасности инф. систем\n");  }  fprintf(output, "=======================================\n");  delete[] strbuffer1;  delete[] strbuffer2;  }  fclose(output);  }  void outputtofile()  {  int i;  char buffer;  FILE \*output = NULL;  fopen\_s(&output, "E:/УНИВЕР/2 семестр/ОснАлгИпрог/Примеры/лаба4/задание2/задание2/файлы для лабы/2.bin", "wb");  if (input == NULL)  {  cout << "Ошибка" << endl;  exit(-1);  }  fprintf(output, "\tСправочник абитуриента\n");  cout << "Введите номер университета: " << endl;  cin >> i;  i--;  char \*strbuffer1 = new char[university[i].uniname.length() + 1];  char \*strbuffer2 = new char[university[i].address.length() + 1];  strcpy(strbuffer1, university[i].uniname.c\_str());  fprintf(output, "================== %d ==================\n", i + 1);  fprintf(output, "Название университета: %s\n", strbuffer1);  strcpy(strbuffer2, university[i].address.c\_str());  fprintf(output, "Адрес: %s\n", strbuffer2);  fprintf(output, "Специальности: \n");  for (int k = 0; k < university[i].specamount; k++)  {  if (university[i].speciality[k].name == 1) fprintf(output, "\tинженер-программист\n");  if (university[i].speciality[k].name == 2) fprintf(output, "\tвеб-дизайнер\n");  if (university[i].speciality[k].name == 3) fprintf(output, "\tгейм-дизайнер\n");  if (university[i].speciality[k].name == 4) fprintf(output, "\tсистемный администратор\n");  if (university[i].speciality[k].name == 5) fprintf(output, "\tвеб-программист\n");  if (university[i].speciality[k].name == 6) fprintf(output, "\tспециалист по безопасности инф. систем\n");  }  fprintf(output, "=======================================\n");  delete[] strbuffer1;  delete[] strbuffer2;  fclose(output);  } |

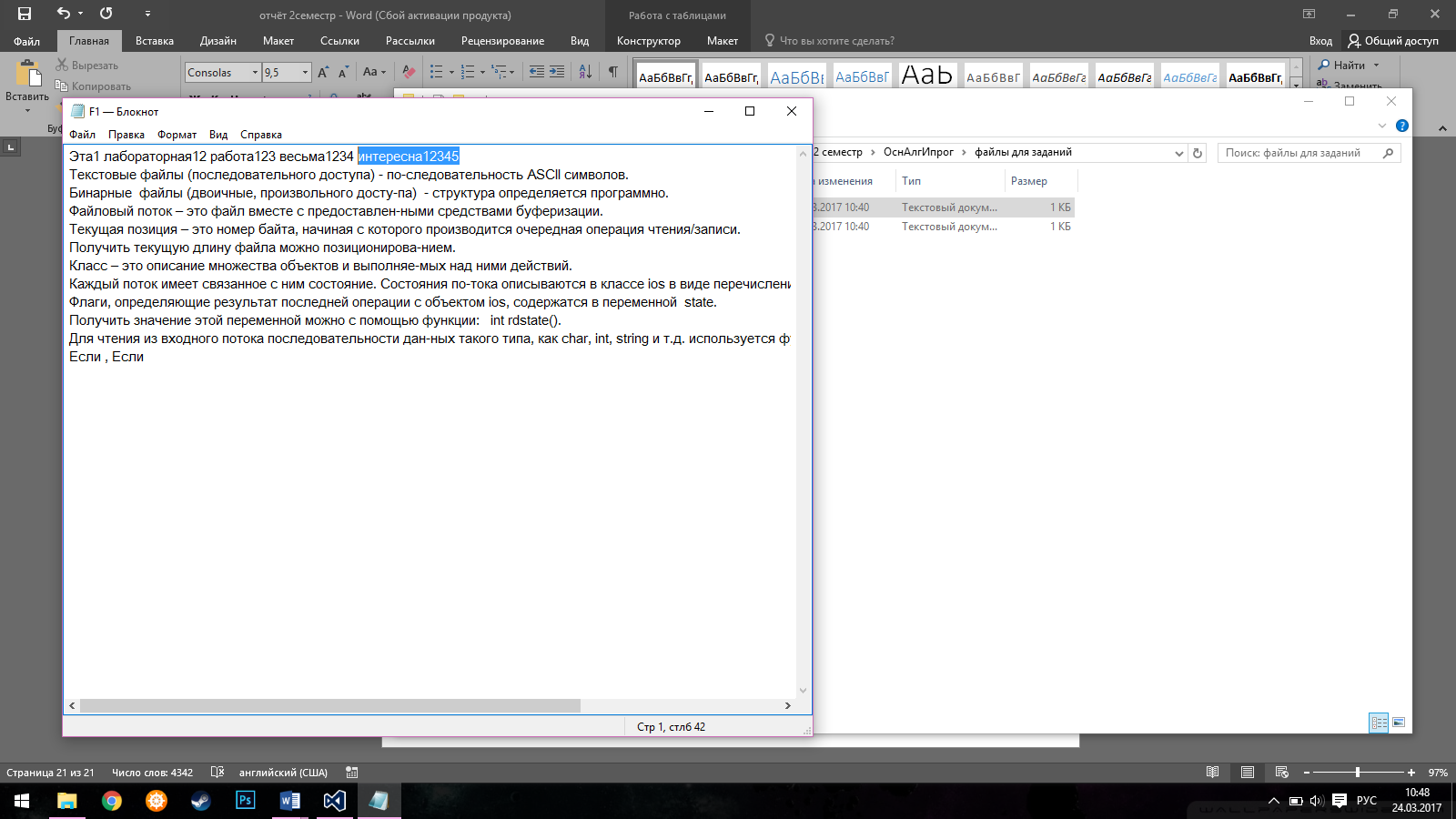


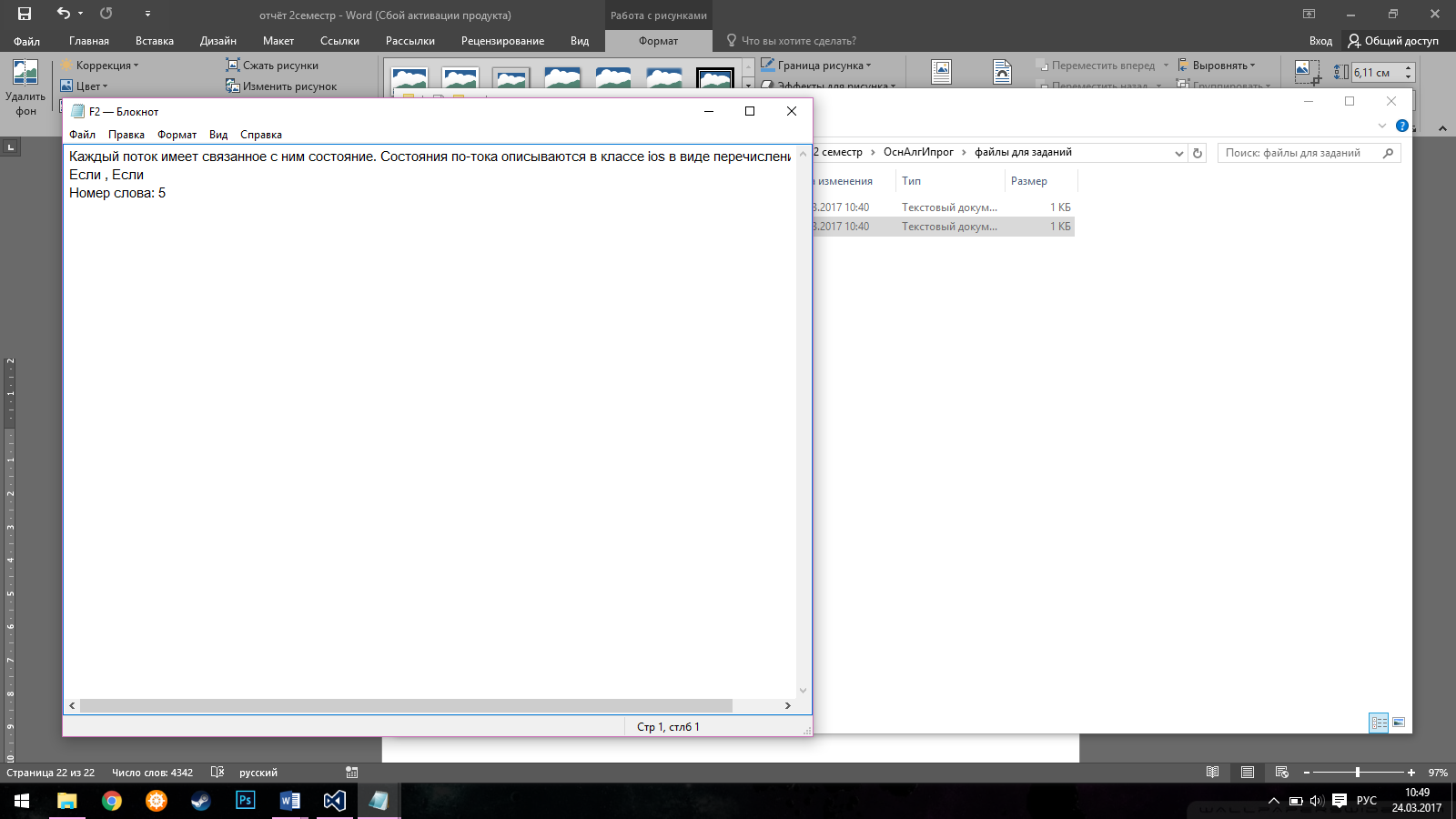
**Лабораторная работа № 5.** [**Работа с файлами на языке С++**](#Лаб5)

**Задание 1**

|  |  |
| --- | --- |
| **вариант**  **15** | В соответствии со своим вариантом разработать программу для работы с файлами на языке **С++**. Предварительно создать текстовый файл **F1** не менее чем из 10 строк и записать в него информацию.  Скопировать из файла F1 в файл F2 все строки, в которых содержится не менее двух одинаковых слов. Определить номер слова, в котором больше всего цифр. |

|  |
| --- |
| #include <fstream>  #include <iostream>  #include <string>  #include <cstring>  #include <cstdio>  #pragma warning(disable:4996)  bool myisdigit(char a);  using namespace std;  int wamax = 0;  int amax = 0;  int am = 0;  int ai = 0;  struct datastorage  {  string sentence;  int wmax;  bool sentencecheck = false;  struct wordstorage  {  string word;  int count = 0;  } words[100];  } dstorage[100];  void main()  {  setlocale(LC\_ALL, "RUS");  ifstream input("E:/УНИВЕР/2 семестр/ОснАлгИпрог/файлы для заданий/F1.txt", ios\_base::in);  while (getline(input, dstorage[amax].sentence)) amax++;  input.close();  for (int i = 0; i < amax; i++)  {  char \*strbuffer = new char[dstorage[i].sentence.length() + 1];  strcpy(strbuffer, dstorage[i].sentence.c\_str());  char \*separator = strtok(strbuffer, " ");  int k = 0;  while (separator != NULL)  {  dstorage[i].words[k].word = separator;  separator = strtok(NULL, " ");  k++;  }  dstorage[i].wmax = k;  delete[]strbuffer;  }  string buffer;  for (int i = 0; i < amax; i++)  {  for (int m = 0; m < dstorage[i].wmax; m++)  {  buffer = dstorage[i].words[m].word;  if (dstorage[i].sentencecheck == true) break;  for (int l = 0; l < dstorage[i].wmax; l++)  {  if (m == l) continue;  if (buffer == dstorage[i].words[l].word) { dstorage[i].sentencecheck = true; break; }  }  }  }  for (int i = 0; i < amax; i++)  {  for (int m = 0; m < dstorage[i].wmax; m++)  {  char \*strbuffer = new char[dstorage[i].words[m].word.length() + 1];  strcpy(strbuffer, dstorage[i].words[m].word.c\_str());  for (int k = 0; strbuffer[k] != '\0'; k++)  {  if (myisdigit(strbuffer[k])) dstorage[i].words[m].count++;  }  delete[]strbuffer;  }  }  int max = 0;  for (int i = 0; i < amax; i++)  {  for (int m = 0; m < dstorage[i].wmax; m++)  {  if (dstorage[i].words[m].count > max) { max = dstorage[i].words[m].count; ai = i + 1; am = m + 1; }  }  }  for (int i = 0; i < ai - 1; i++)  {  wamax += dstorage[i].wmax;  } wamax += am;  ofstream output("E:/УНИВЕР/2 семестр/ОснАлгИпрог/файлы для заданий/F2.txt", ios\_base::out);  for (int i = 0; i < amax; i++)  {  if (dstorage[i].sentencecheck == true) output << dstorage[i].sentence << '\n';  }  output << "Номер слова: " << wamax;  output.close();  system("pause");  }  bool myisdigit(char a)  {  if (a >= '0' && a <= '9') return true;  else return false;  } |





**Задание 2**

|  |  |
| --- | --- |
| **Вариант**  **15** | В соответствии со своим вариантом разработать программу с использованием структуры в виде *объединения*, для работы с данными из таблицы, приведенной ниже.  Реализовать функции ввода, вывода, редактирования, удаления, поиска элементов. Предусмотреть ввод данных, как с клавиатуры, так и из файла. Вывод данных осуществить как на экран, так и в файл на языке С++.  **Справочник абитуриента.** Наименование вуза, адрес, перечень специальностей, конкурс прошлого года по каждой специальности, размер оплаты при договорном обучении. Выбор по разным критериям: все о данном вузе; все о данной специальности, поиск минимального конкурса по данной специальности. |

|  |
| --- |
| /\*Справочник абитуриента.  Наименование вуза,  адрес,  перечень специальностей,  конкурс прошлого года по каждой специальности,  размер оплаты при договорном обучении.  \*/  #include <fstream>  #include <iostream>  #include <windows.h>  #include <string>  #include <sstream>  #pragma warning(disable:4996)  using namespace std;  const int max = 100;  int amax = 0;  enum specialities  {  engineer\_programmer = 1,  web\_designer,  game\_designer,  system\_administrator,  web\_programmer,  security\_specialist  };  struct forspecialities  {  int name;  unsigned mark : 9;  int price;  };  struct universities  {  string uniname;  string address;  forspecialities speciality[30];  int specamount;  } university[max], cleaner;  union bufferunion  {  int bufferint;  char cbuffer[1];  } buffer;  void input();  void list();  void search(int switchkey2, int & min, int & ak, int & am);  void infouniversity(int i);  void searchUniname();  void minimal();  void infospeciality();  void inputfromfile();  void outputtofile();  void massoutputtofile();  void main()  {  SetConsoleCP(1251);  SetConsoleOutputCP(1251);  int n;  do  {  cout << "\tСправочник абитуриента" << endl;  cout << "\t 1 - список университетов" << endl;  cout << "\t 2 - информация о выбранном университете" << endl;  cout << "\t 3 - информация о выбранной специальности" << endl;  cout << "\t 4 - поиск минимального конкурса по специальности" << endl;  cout << "\t 5 - поиск университета по названию" << endl;  cout << "\t 6 - добавить университет в список" << endl;  cout << "\t 7 - добавить информацию из файла" << endl;  cout << "\t 8 - записать информацию в файл" << endl;  cout << "\t 9 - для выхода из программы" << endl;  cout << "\t 0 - очистка консоли" << endl;  cin >> n;  switch (n)  {  case 1:  {  list(); break;  }  case 2:  {  int i;  cout << "Введите номер университета: ";  cin >> i;  infouniversity(i); break;  }  case 3:  {  infospeciality(); break;  }  case 4:  {  minimal(); break;  }  case 5:  {  searchUniname(); break;  }  case 6:  {  input(); break;  }  case 7:  {  inputfromfile(); break;  }  case 8:  {  int switchkey21;  do  {  cout << "1 - вывести информацию об одном университете" << endl;  cout << "2 - вывести информацию о всех университетах" << endl;  cout << "3 - отмена" << endl;  cin >> switchkey21;  switch (switchkey21)  {  case 1:  {  outputtofile(); switchkey21 = 3; break;  }  case 2:  {  massoutputtofile(); switchkey21 = 3; break;  }  }  } while (switchkey21 != 3);  }  case 0:  system("cls"); break;  }  } while (n != 9);  }  void input()  {  if (amax < max)  {  cin.clear();  while (cin.get() != '\n');  cout << "Введите название университета: " << endl;  getline(cin, university[amax].uniname);  cout << "Введите адрес университета: " << endl;  getline(cin, university[amax].address);  int specamount;  int switchkey;  int mark;  cout << "Введите количество специальностей: ";  cin >> specamount;  university[amax].specamount = specamount;  for (int i = 0; i < specamount; i++)  {  cout << "Введите номер специальности: " << endl;  cout << "1 - инженер-программист" << endl;  cout << "2 - веб-дизайнер" << endl;  cout << "3 - гейм-дизайнер" << endl;  cout << "4 - системный администратор" << endl;  cout << "5 - веб-программист" << endl;  cout << "6 - специалист по безопасности инф. систем" << endl;  cin >> switchkey;  switch (switchkey)  {  case 1:university[amax].speciality[i].name = engineer\_programmer; break;  case 2:university[amax].speciality[i].name = web\_designer; break;  case 3:university[amax].speciality[i].name = game\_designer; break;  case 4:university[amax].speciality[i].name = system\_administrator; break;  case 5:university[amax].speciality[i].name = web\_programmer; break;  case 6:university[amax].speciality[i].name = security\_specialist; break;  }  cout << "Введите проходной балл по данной специальности: ";  cin >> mark;  university[amax].speciality[i].mark = mark;  cout << "Введите стоимость обучения по данной специальности: ";  cin >> university[amax].speciality[i].price;  }  amax++;  }  else  cout << "ошибка" << endl;  }  void list()  {  for (int i = 0; i < amax; i++)  {  cout << i + 1 << ". " << university[i].uniname << endl;  }  }  void infouniversity(int i)  {  i--;  if (i < 0 || i > amax - 1) cout << "Выход за границы" << endl;  else  {  cout << "Название университета: " << university[i].uniname << endl;  cout << "Адрес: " << university[i].address << endl;  cout << "Специальности:" << endl;  for (int k = 0; k < university[i].specamount; k++)  {  if (university[i].speciality[k].name == 1) cout << "\tинженер-программист" << endl;  if (university[i].speciality[k].name == 2) cout << "\tвеб-дизайнер" << endl;  if (university[i].speciality[k].name == 3) cout << "\tгейм-дизайнер" << endl;  if (university[i].speciality[k].name == 4) cout << "\tсистемный администратор" << endl;  if (university[i].speciality[k].name == 5) cout << "\tвеб-программист" << endl;  if (university[i].speciality[k].name == 6) cout << "\tспециалист по безопасности инф. систем" << endl;  }  }  cout << endl;  }  void searchUniname()  {  cin.clear();  while (cin.get() != '\n');  string searchline;  bool a = false;  cout << "Введите название университета: ";  getline(cin, searchline);  for (int i = 0; i <= amax; i++)  {  if (searchline == university[i].uniname)  {  cout << "Название университета: " << university[i].uniname << endl;  cout << "Адрес: " << university[i].address << endl;  a = true;  }  }  if (a == false) cout << "Совпадений нет" << endl;  }  void infospeciality()  {  int switchkey3;  cout << "Выберите специальность:" << endl;  cout << "1 - инженер-программист" << endl;  cout << "2 - веб-дизайнер" << endl;  cout << "3 - гейм-дизайнер" << endl;  cout << "4 - системный администратор" << endl;  cout << "5 - веб-программист" << endl;  cout << "6 - специалист по безопасности инф. систем" << endl;  cin >> switchkey3;  for (int k = 0; k < amax; k++)  {  for (int m = 0; m < university[k].specamount; m++)  {  if (university[k].speciality[m].name == switchkey3)  {  cout << k + 1 << "." << university[k].uniname << endl;  cout << "Проходной балл: " << university[k].speciality[m].mark << endl;  cout << "Стоимость обучения: " << university[k].speciality[m].price << endl << endl;  }  }  }  }  void minimal()  {  int min = 99999, ak = 0, am = 0;  int switchkey2;  cout << "Выберите специальность:" << endl;  cout << "1 - инженер-программист" << endl;  cout << "2 - веб-дизайнер" << endl;  cout << "3 - гейм-дизайнер" << endl;  cout << "4 - системный администратор" << endl;  cout << "5 - веб-программист" << endl;  cout << "6 - специалист по безопасности инф. систем" << endl;  cin >> switchkey2;  switch (switchkey2)  {  case 1: search(switchkey2, min, ak, am);  case 2: search(switchkey2, min, ak, am);  case 3: search(switchkey2, min, ak, am);  case 4: search(switchkey2, min, ak, am);  case 5: search(switchkey2, min, ak, am);  case 6: search(switchkey2, min, ak, am);  }  if (min == -1) exit(0);  cout << "Минимальный балл равен: " << min << endl;  cout << "Специальность ";  if (am == 1) cout << "инженер-программист";  if (am == 2) cout << "веб-дизайнер";  if (am == 3) cout << "гейм-дизайнер";  if (am == 4) cout << "системный администратор";  if (am == 5) cout << "веб-программист";  if (am == 6) cout << "специалист по безопасности инф. систем";  cout << " в университете " << university[ak].uniname << endl;  }  void search(int searchkey2, int & min, int & ak, int & am)  {  bool b = false;  for (int k = 0; k < amax; k++)  {  for (int m = 0; m < university[k].specamount; m++)  {  if (university[k].speciality[m].name == searchkey2)  {  b = true;  if (university[k].speciality[m].mark < min)  {  min = university[k].speciality[m].mark; ak = k; am = university[k].speciality[m].name;  }  }  }  }  if (b == false)  {  cout << "Университета с данной специальностью нет" << endl;  min = -1;  }  }  void inputfromfile()  {  string sbuffer;  unsigned int uibuffer;  ifstream input("E:/УНИВЕР/2 семестр/ОснАлгИпрог/файлы для заданий/1.txt", ios\_base::in);  if (!input.is\_open()) cout << "Ошибка!\n";  else  {  int uniamount;  getline(input, sbuffer);  strcpy(buffer.cbuffer, sbuffer.c\_str());  uniamount = buffer.cbuffer[0] - '0';  for (int i = 0; i < uniamount; i++)  {  getline(input, sbuffer);  university[i].uniname = sbuffer;  getline(input, sbuffer);  university[i].address = sbuffer;  getline(input, sbuffer);  strcpy(buffer.cbuffer, sbuffer.c\_str());  university[i].specamount = buffer.cbuffer[0] - '0';  for (int k = 0; k < university[i].specamount; k++)  {  getline(input, sbuffer);  stringstream stream(sbuffer);  stream >> university[i].speciality[k].name; stream.get();  stream >> uibuffer; university[i].speciality[k].mark = uibuffer; stream.get();  stream >> university[i].speciality[k].price;  }  amax++;  }  }  input.close();  }  void massoutputtofile()  {  ofstream output("E:/УНИВЕР/2 семестр/ОснАлгИпрог/файлы для заданий/2.txt", ios\_base::out);  if (!output.is\_open()) cout << "Ошибка!\n";  else  {  for (int i = 0; i < amax; i++)  {  output << "================" << i + 1 << "================" << endl;  output << "Название: " << university[i].uniname << endl;  output << "Адрес: " << university[i].address << endl;  output << "Специальности: " << endl;  for (int k = 0; k < university[i].specamount; k++)  {  if (university[i].speciality[k].name == 1) output << "\tинженер-программист\n";  if (university[i].speciality[k].name == 2) output << "\tвеб-дизайнер\n";  if (university[i].speciality[k].name == 3) output << "\tгейм-дизайнер\n";  if (university[i].speciality[k].name == 4) output << "\tсистемный администратор\n";  if (university[i].speciality[k].name == 5) output << "\tвеб-программист\n";  if (university[i].speciality[k].name == 6) output << "\tспециалист по безопасности инф. систем\n";  }  output << "=================================" << endl;  }  }  output.close();  }  void outputtofile()  {  ofstream output("E:/УНИВЕР/2 семестр/ОснАлгИпрог/файлы для заданий/2.txt", ios\_base::out);  if (!output.is\_open()) cout << "Ошибка!\n";  else  {  cout << "Введите номер университета: ";  cin >> buffer.bufferint; buffer.bufferint;  output << "================" << buffer.bufferint + 1 << "================" << endl;  output << "Название: " << university[buffer.bufferint].uniname << endl;  output << "Адрес: " << university[buffer.bufferint].address << endl;  output << "Специальности: " << endl;  for (int k = 0; k < university[buffer.bufferint].specamount; k++)  {  if (university[buffer.bufferint].speciality[k].name == 1) output << "\tинженер-программист\n";  if (university[buffer.bufferint].speciality[k].name == 2) output << "\tвеб-дизайнер\n";  if (university[buffer.bufferint].speciality[k].name == 3) output << "\tгейм-дизайнер\n";  if (university[buffer.bufferint].speciality[k].name == 4) output << "\tсистемный администратор\n";  if (university[buffer.bufferint].speciality[k].name == 5) output << "\tвеб-программист\n";  if (university[buffer.bufferint].speciality[k].name == 6) output << "\tспециалист по безопасности инф. систем\n";  }  output << "=================================" << endl;  }  output.close();  } |

**Лабораторная работа № 6.** [**Динамические структуры данных. Списки**](#Лаб6)

|  |  |
| --- | --- |
| **вариант**  **15** | FindMax – функция поиска максимального элемента списка по одному из выбранных полей. |

|  |
| --- |
| #include <iostream>  #include <fstream>  #include <cstdio>  #include <Windows.h>  #include <string>  #pragma warning(disable:4996)  using namespace std;  struct Adr  {  char name[30];  char street[40];  char city[20];  struct Adr \*next;  struct Adr \*prev;  };  struct Adr \*head; struct Adr \*last;  int Menu(void)  {  char s[80]; int c; cout << endl;  cout << "1. Ввод имени" << endl;  cout << "2. Удаление имени" << endl;  cout << "3. Вывод на экран" << endl;  cout << "4. Поиск" << endl;  cout << "5. Сохранить в файл" << endl;  cout << "6. Загрузить из файла" << endl;  cout << "7. Нахождение максимального элемента списка по имени" << endl;  cout << "8. Нахождение максимального элемента списка по улице" << endl;  cout << "9. Нахождение максимального элемента списка по городу" << endl;  cout << "0. Выход" << endl << endl;  do  {  cout << "Ваш выбор: "; gets\_s(s); cout << endl;  c = atoi(s);  } while (c < 0 || c > 9);  return c;  }  void Sozdat(Adr \*i, Adr \*\*head, Adr \*\*last)  {  struct Adr \*old, \*p;  if (\*last == NULL)  {  i->next = NULL; i->prev = NULL;  \*last = i; \*head = i; return;  }  p = \*head; old = NULL;  while (p)  {  if (strcmp(p->name, i->name) < 0)  {  old = p;  p = p->next;  }  else  {  if (p->prev)  {  p->prev->next = i; i->next = p;  i->prev = p->prev; p->prev = i;  return;  }  i->next = p; i->prev = NULL;  p->prev = i; \*head = i;  return;  }  }  old->next = i; i->next = NULL;  i->prev = old; \*last = i;  }  void Vvod(char \*prompt, char \*s, int count)  {  char p[255];  do  {  cout << (prompt); fgets(p, 254, stdin);  if (strlen(p) > count)  cout << ("Слишком длинная строка"); //длина строки  } while (strlen(p) > count);  p[strlen(p) - 1] = 0; strcpy(s, p);  }  void VvodSp(void) // Ввод строки  {  struct Adr \*t; int i;  t = new (struct Adr);  if (!t) { cout << ("Нет свободной памяти"); return; }  Vvod("Введите имя: ", t->name, 30);  Vvod("Введите улицу: ", t->street, 40);  Vvod("Введите город: ", t->city, 20);  Sozdat(t, &head, &last);  }  void VyvodSp(void) //Вывод списка на экран  {  struct Adr \*t; t = head;  while (t)  {  cout << t->name << ' ' << t->street << ' ' << t->city << endl;  t = t->next;  }  cout << "" << endl;  }  void Poisk(void) // Поиск имени в списке  {  char name[40]; struct Adr \*t; t = head;  cout << "Введите имя: "; gets\_s(name);  while (t)  {  if (!strcmp(name, t->name)) break; t = t->next;  }  if (!t) cout << "Имя не найдено" << endl;  else cout << t->name << ' ' << t->street << ' ' << t->city << endl;  }  void Udalit(Adr \*\*head, Adr \*\*last) // Удаление имени из списка  {  struct Adr \*t; char name[40]; t = \*head;  cout << "Введите имя: "; gets\_s(name);  while (t)  {  if (!strcmp(name, t->name)) break; t = t->next;  }  if (t)  {  if (\*head == t)  {  \*head = t->next;  if (\*head) (\*head)->prev = NULL;  else \*last = NULL;  }  else  {  t->prev->next = t->next;  if (t != \*last) t->next->prev = t->prev;  else \*last = t->prev;  } delete t;  }  }  void Zapisat(void) //Запись в файл  {  struct Adr \*t;  ofstream output("E:/УНИВЕР/2 семестр/ОснАлгИпрог/файлы для заданий/лаба 6/2.txt", ios\_base::out);  if (!output.is\_open()) { cout << "Файл не открывается" << endl; exit(1); }  cout << "Сохранение в файл" << endl;  t = head;  while (t)  {  output << t->name << endl;  output << t->street << endl;  output << t->city << endl;  output << endl;  t = t->next;  }  output.close();  }  void Schitat() //Считывание из файла  {  struct Adr \*t; string sbuffer;  ifstream input("E:/УНИВЕР/2 семестр/ОснАлгИпрог/файлы для заданий/лаба 6/1.txt", ios\_base::in);  if (!input.is\_open()) { cout << "Файл не открывается" << endl; exit(1); }  while (head)  {  last = head->next; delete head; head = last;  }  head = last = NULL;  cout << "Загрузка из файла" << endl;  do  {  t = new (struct Adr); int i = 0;  if (!t) { cout << "Нет свободной памяти" << endl; return; }  getline(input, sbuffer); for (i = 0; sbuffer.c\_str()[i] != '\0'; i++) t->name[i] = sbuffer.c\_str()[i]; for (i; i < 30; i++) t->name[i] = '\0'; i = 0;  getline(input, sbuffer); for (i = 0; sbuffer.c\_str()[i] != '\0'; i++) t->street[i] = sbuffer.c\_str()[i]; for (i; i < 40; i++) t->street[i] = '\0'; i = 0;  getline(input, sbuffer); for (i = 0; sbuffer.c\_str()[i] != '\0'; i++) t->city[i] = sbuffer.c\_str()[i]; for (i; i < 20; i++) t->city[i] = '\0'; i = 0;  Sozdat(t, &head, &last);  } while (getline(input, sbuffer));  input.close();  }  void FindMaxName()  {  int max = 0; string buffer;  struct Adr \*t; t = head;  while (t)  {  int size = 0;  for (int i = 0; t->name[i] != '\0'; i++) size++;  if (size > max) max = size;  t = t->next;  }  t = head;  while (t)  {  int size = 0;  for (int i = 0; t->name[i] != '\0'; i++) size++;  if (size == max) { buffer = t->name; cout << "Самое длинное имя: " << buffer << endl; } t = t->next;  }  }  void FindMaxStreet()  {  int max = 0; string buffer;  struct Adr \*t; t = head;  while (t)  {  int size = 0;  for (int i = 0; t->street[i] != '\0'; i++) size++;  if (size > max) max = size;  t = t->next;  }  t = head;  while (t)  {  int size = 0;  for (int i = 0; t->street[i] != '\0'; i++) size++;  if (size == max) { buffer = t->street; cout << "Самое длинное название улицы: " << buffer << endl; } t = t->next;  }  }  void FindMaxCity()  {  int max = 0; string buffer;  struct Adr \*t; t = head;  while (t)  {  int size = 0;  for (int i = 0; t->city[i] != '\0'; i++) size++;  if (size > max) max = size;  t = t->next;  }  t = head;  while (t)  {  int size = 0;  for (int i = 0; t->city[i] != '\0'; i++) size++;  if (size == max) { buffer = t->city; cout << "Самое длинное название города: " << buffer << endl; } t = t->next;  }  }  int main(void)  {  SetConsoleCP(1251);  SetConsoleOutputCP(1251);  head = last = NULL;  for (;;)  {  switch (Menu())  {  case 1: VvodSp(); break;  case 2: Udalit(&head, &last); break;  case 3: VyvodSp(); break;  case 4: Poisk(); break;  case 5: Zapisat(); break;  case 6: Schitat(); break;  case 7: FindMaxName(); break;  case 8: FindMaxStreet(); break;  case 9: FindMaxCity(); break;  case 0: exit(0);  }  }  return 0;  } |

**Лабораторная работа № 7.** [**Разработка проекта с использованием списков**](#Лаб7)

|  |  |
| --- | --- |
| **вариант**  **15** | AddLEnd – функция добавления в конец списка всех элементов списка L. |

|  |
| --- |
| Заголовочный файл **List.h** |
| #pragma once  struct Element // Элемент списка  {  Element\* Prev; // указатель на предыдущий элемент  Element\* Next; // указатель на следующий элемент  void\* Data; // данные  Element(Element\* prev, void\* data, Element\* next) // конструктор  {  Prev = prev;  Data = data;  Next = next;  }  Element\* GetNext() // получить следующий  {  return Next;  };  Element\* GetPrev() // получить предыдущий  {  return Prev;  };  };  struct Person  {  char name[20];  char exam[20];  int day; int month; int year;  Person \*next;  };  struct Object // Блок управления списком  {  Element\* Head; // указатель на начало списка  Object()  {  Head = NULL;  };  Element\* GetFirst() // получить первый элемент списка  {  return Head;  };  Element\* GetLast(); // получить последний элемент списка  Element\* Search(void\* data); // найти первый элемент по данным  bool Insert(); // добавить элемент в начало  bool InsertEnd(void\* data); // добавить в конец  bool Delete(Element\* e); // удалить по адресу элемента  bool Delete(void\* data); // удалить первый по данным  void DeleteList(); // очистить список  void Object::PrintList(void(\*f)(void\*));  void Object::PrintList(void(\*f)(void\*), Element\*);  void Object::Deletek();  int Object::CountList();  void Object::Add();  bool Object::DeleteDouble();  };  Object Create(); // создать список |

|  |
| --- |
| Файлы исходного кода **List.cpp** |
| #include "stdafx.h"  #include "List.h"  #include <iostream>  using namespace std;  bool Object::Insert() // Вставка в начало  {  Person\* a = new Person;  cout << "Введите фамилию преподавателя:" << endl;  cin >> a->name;  cout << "Введите название экзамена:" << endl;  cin >> a->exam;  cout << "Введите дату экзамена" << endl;  cout << "День:" << endl;  cin >> a->day;  cout << "Месяц:" << endl;  cin >> a->month;  cout << "Год:" << endl;  cin >> a->year;  bool rc = 0;  if (Head == NULL) { Head = new Element(NULL, a, Head); rc = true; }  else { Head = (Head->Prev = new Element(NULL, a, Head)); rc = true; }  return rc;  }  Element\* Object::Search(void\* data) // Найти заданный элемент  {  char search[30];  cout << "Введите фамилию преподавателя: ";  cin >> search;  Element\* rc = Head;  Person\* a;  while ((rc != NULL) && (rc->Data != data))  {  a = (Person\*)rc->Data;  if (!strcmp(a->name, search))  {  cout << "Фамилия: " << a->name << endl;  cout << "Экзамен: " << a->exam << endl;  cout << "Дата: " << a->day << "." << a->month << "." << a->year << endl;  return rc;  }  else rc = rc->Next;  }  cout << "Совпадений нет" << endl;  return rc;  }  void Object::PrintList(void(\*f)(void\*)) // Вывод  {  Element\* e = Head;  if (!e)  {  std::cout << "Нет записей!" << endl;  }  while (e != NULL)  {  f(e->Data); e = e->GetNext();  };  }  void Object::PrintList(void(\*f)(void\*), Element \*e)  {  f(e->Data);  }  bool Object::Delete(Element\* e) // Удалить по ссылке  {  bool rc = 0;  if (rc = (e != NULL))  {  if (e->Next != NULL) e->Next->Prev = e->Prev;  if (e->Prev != NULL) e->Prev->Next = e->Next;  else Head = e->Next;  delete e;  }  return rc;  }  bool Object::Delete(void\* data) // Удалить по значению  {  return Delete(Search(data));  };  void Object::DeleteList()  {  Element\* e = Head;  Element\* next;  while (e)  {  next = e->GetNext();  Delete(e);  e = next;  }  delete Head;  Head = NULL;  }  void Object::Deletek()  {  int k;  cout << "Введите количество элементов, которые нужно удалить: ";  cin >> k;  Element\* e = Head;  Element\* next;  for (int i = 0; i < k; i++)  {  if (!e) break;  next = e->GetNext();  Delete(e);  e = next;  }  }  int Object::CountList()  {  int n = 0;  Element\* e = Head;  Element\* next;  while (e)  {  n++;  next = e->GetNext();  e = next;  }  std::cout << "В списке " << n << " элементов\n";  return n;  }  Element\* Object::GetLast()  {  Element\* e = Head, \*rc = e;  while (e != NULL)  {  rc = e;  e = e->GetNext();  };  return rc;  }  Object Create()  {  return \*(new Object());  } |

|  |
| --- |
| Исходный код |
| #include "stdafx.h"  #include "List.h"  #include <iostream>  #include <Windows.h>  using namespace std;  Object L = Create(); // создать список L2  void f(void\* b) //функция используется при выводе  {  Person \*a = (Person\*)b;  cout << "имя: " << a->name << " экзамен:" << a->exam << " дата: " << a->day << "." << a->month << "." << a->year << "." << endl;  }  int \_tmain(int argc, \_TCHAR\* argv[])  {  SetConsoleCP(1251);  SetConsoleOutputCP(1251);  int key;  do  {  cout << "1 - добавить элемент в список" << endl;  cout << "2 - найти заданный элемент" << endl;  cout << "3 - удалить элемент" << endl;  cout << "4 - удалить список" << endl;  cout << "5 - посчитать количество элементов списка" << endl;  cout << "6 - удалить k - первых элементов списка" << endl;  cout << "7 - вывести список" << endl;  cout << "0 - выйти из программы" << endl;  cin >> key;  switch (key)  {  case 1: {L.Insert(); break; }  case 2: {L.Search(&L); break; }  case 3: {L.Delete(&L); break; }  case 4: {L.DeleteList(); break; }  case 5: {L.CountList(); break; }  case 6: {L.Deletek(); break; }  case 7: {L.PrintList(f); break; }  }  } while (key != 0);  return 0;  } |

**Лабораторная работа № 8.** [**Полустатические структуры данных: стеки**](#Лаб8)

|  |
| --- |
| Разработать функцию, которая удаляет элемент стека. |
| #include "stdafx.h"  #include <Windows.h>  #include <iostream>  #include "Stack.h"  #include <string>  using namespace std;  int \_tmain(int argc, \_TCHAR\* argv[])  {  Stack(a);  SetConsoleCP(1251);  SetConsoleOutputCP(1251);  int key;  do  {  cout << "1 - добавить элемент в стек" << endl;  cout << "2 - удалить элемент" << endl;  cout << "3 - очистить стек" << endl;  cout << "4 - удалить первый элемент, значение которого превышает 10" << endl;  cout << "5 - вывести стек" << endl;  cout << "6 - записать в файл" << endl;  cout << "7 - считать из файла" << endl;  cout << "0 - выйти из программы" << endl;  cin >> key;  switch (key)  {  case 1: { a.push(&a); break; }  case 2: { a.pop(&a); break; }  case 3: { a.clear(&a); break; }  case 4: { a.deletefirst(&a); break; }  case 5: { a.print(&a); break; }  case 6: { a.fwrite(&a); break; }  case 7: { a.fread(&a); break; }  }  } while (key != 0);  return 0;  } |

|  |
| --- |
| Stack |
| #pragma once  #include "stdafx.h"  #include <Windows.h>  #include <iostream>  #include <fstream>  #include "Stack.h"  #include <string>  void Stack::push(Stack\* a)  {  system("cls");  Person \*tmp = new Person;  cout << "Введите фамилию преподавателя: ";  cin >> tmp->name; //доступ к переменной списка  cout << "Введите название экзамена: ";  cin >> tmp->exam;  cout << "Введите дату экзамена" << endl;  cout << "День: ";  cin >> tmp->day;  cout << "Месяц: ";  cin >> tmp->month;  cout << "Год: ";  cin >> tmp->year;  tmp->next = NULL;  if (top != NULL) tmp->next = top;  top = tmp;  }  void Stack::pop(Stack\* a)  {  system("cls");  Person \*tmp;  if (top == NULL)  {  cout << "Стек пуст" << endl;  return;  }  tmp = top;  top = top->next;  delete tmp;  }  void Stack::print(Stack\* a)  {  system("cls");  Person \*tmp = top;  if (top == NULL) cout << "Стек пуст" << endl;  while (tmp != NULL)  {  int i = 0;  cout << "Фамилия: ";  while (tmp->name[i] != 'Н')  {  cout << tmp->name[i];  i++;  }  cout << " " << "экзамен: ";  i = 0;  while (tmp->exam[i] != 'Н')  {  cout << tmp->exam[i];  i++;  }  cout << " " << "дата: " << tmp->day << "." << tmp->month << "." << tmp->year << endl;  tmp = tmp->next;  }  }  void Stack::clear(Stack\* a)  {  system("cls");  Person \*tmp = top;  Person \*tmp2;  while (tmp != NULL)  {  tmp2 = tmp->next;  delete tmp;  tmp = tmp2;  }  top = NULL;  }  void Stack::deletefirst(Stack\* a)  {  system("cls");  Person \*tmp = top;  Person \*tmp2;  while (tmp != NULL)  {  if (tmp->day >= 10) { top = tmp->next; delete tmp; return; }  if (tmp->next->day >= 10) break;  tmp = tmp->next;  }  tmp2 = tmp->next;  tmp->next = tmp2->next;  delete tmp2;  }  void Stack::fread(Stack\* a)  {  system("cls");  string sbuffer;  ifstream input("C:/Users/nitro/Desktop/проекты вижлы/Projects/семестр2/лаба8/лаба8/1.txt", ios\_base::in);  do  {  Person \*tmp = new Person;  int i = 0;  if (!input.is\_open()) { cout << "Файл не открывается" << endl; exit(-1); }  getline(input, sbuffer); for (i = 0; sbuffer.c\_str()[i] != '\0'; i++) tmp->name[i] = sbuffer.c\_str()[i]; tmp->name[i + 1] = '\0';  getline(input, sbuffer); for (i = 0; sbuffer.c\_str()[i] != '\0'; i++) tmp->exam[i] = sbuffer.c\_str()[i]; tmp->exam[i + 1] = '\0';  input >> tmp->day;  input >> tmp->month;  input >> tmp->year;  tmp->next = NULL;  if (top != NULL) tmp->next = top;  top = tmp;  getline(input, sbuffer);  } while (getline(input, sbuffer));  input.close();  }  void Stack::fwrite(Stack\* a)  {  system("cls");  ofstream output("C:/Users/nitro/Desktop/проекты вижлы/Projects/семестр2/лаба8/лаба8/2.txt", ios\_base::out);  Person \*tmp = top;  if (top == NULL) cout << "Стек пуст" << endl;  while (tmp != NULL)  {  int i = 0;  while (tmp->name[i] != 'Н') { output << tmp->name[i]; i++; } i = 0;  output << '\n';  while (tmp->exam[i] != 'Н') { output << tmp->exam[i]; i++; } i = 0;  output << '\n';  output << '\n';  output << tmp->day;  output << '\n';  output << tmp->month;  output << '\n';  output << tmp->year;  output << '\n';  output << '\n';  tmp = tmp->next;  }  output.close();  } |

|  |
| --- |
| Заголовочные файлы |
| #pragma once  #include <Windows.h>  #include <iostream>  #include "Stack.h"  #include <string>  using namespace std;  struct Person  {  char name[20];  char exam[20];  int day; int month; int year;  Person \*next;  };  struct Stack  {  Person \*top;  Stack()  {  top = NULL;  }  void Stack::push(Stack\* a);  void Stack::pop(Stack\* a);  void Stack::print(Stack\* a);  void Stack::deletefirst(Stack\* a);  void Stack::clear(Stack\* a);  void Stack::fwrite(Stack\* a);  void Stack::fread(Stack\* a);  }; |