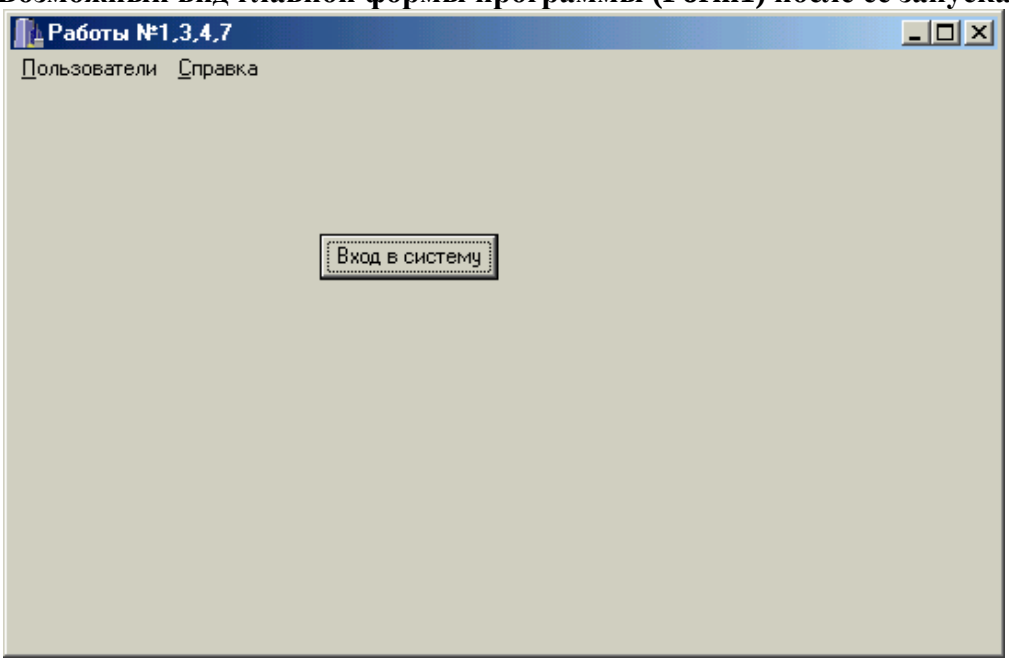
Указания по выполнению лабораторных работ по дисциплине «Защита информации»

Лабораторная работа №1

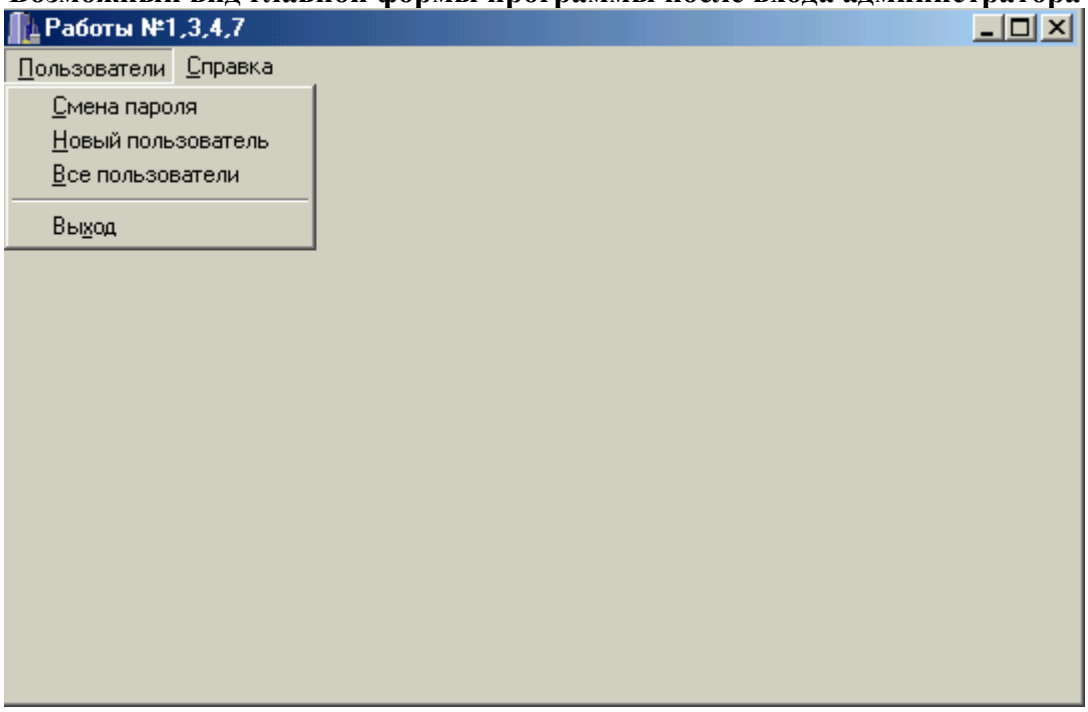
1. После запуска системы программирования Borland C++ Builder автоматически создается новый проект с пустой главной формой. С помощью мыши и палитры компонентов (расположена под главным меню Borland C++ Builder) добавить на главную форму главное меню (компонент MainMenu закладки Standard) и кнопку (компонент Button закладки Standard).

Возможный вид главной формы программы (Form1) после ее запуска



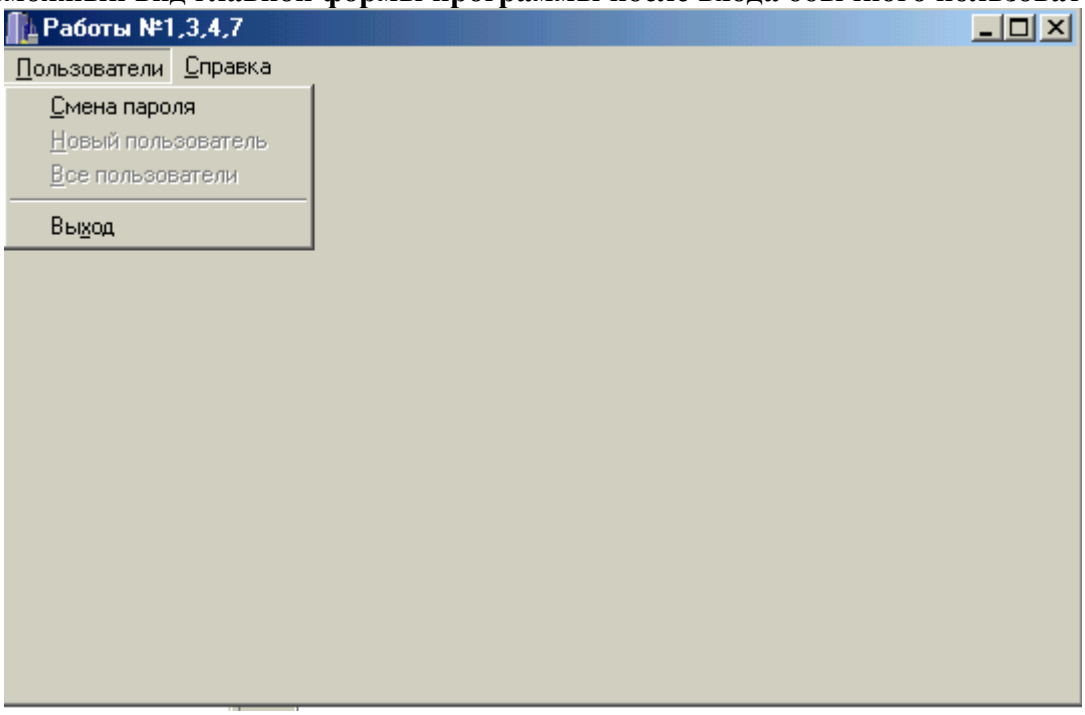
2. Двойным щелчком на значке главного меню открыть редактор меню и добавить в меню два подменю, а в каждое подменю – команды.

Возможный вид главной формы программы после входа администратора



3. Команды «Новый пользователь» и «Все пользователи» заблокировать с помощью инспектора объектов в левом нижнем углу окна Borland C++ Builder (свойство Enabled команд меню установить в false).

Возможный вид главной формы программы после входа обычного пользователя



4. С помощью команды File | New | Form добавить к проекту дополнительные формы, образцы которых приведены в описании лабораторной работы (форма входа в систему создается с помощью команды File | New | Other | Dialogs и шаблона Password Dialog). Добавить на созданные формы необходимые элементы с помощью закладки Standard палитры компонентов: текстовые редакторы (компонент Edit), выключатели (CheckBox), кнопки (Button), надписи (Label).
5. Задать указанные ниже имена элементам созданных форм и самим формам (с помощью инспектора объектов).

Имена (значения свойства Name) элементов форм, используемые в расположенных ниже фрагментах программного кода

Тип и текст элемента	Имя элемента в программе
Главная форма (Form1)	
Команда меню Смена пароля	Change
Команда меню Новый пользователь	New
Команда меню Все пользователи	All
Кнопка Вход в систему	Button1
Форма входа в программу (PasswordDlg)	
Редактор для ввода имени	Login
Редактор для ввода пароля	Password
Форма просмотра (редактирования) списка пользователей (Form3)	
Редактор для отображения имени пользователя	UserName
Выключатель Блокировка	CheckBox1
Выключатель Парольное ограничение	CheckBox2
Кнопка Следующий	Next
Кнопка Ok	Button1
Форма добавления нового пользователя (Form4)	
Редактор для ввода имени пользователя	UserName
Форма смены пароля (Form5)	

Редактор для ввода пароля	Edit1
Редактор для подтверждения ввода пароля	Edit2

6. С помощью команды File | Save Project As сохранить созданные проект и все его формы в своей индивидуальной папке на несистемном диске.
7. Для использования в одной форме элементов другой формы предназначена команда File | Include Unit Hdr.
8. С помощью буфера обмена добавить определения в заголовочный (.h) файл модуля с главной формой (Unit1) перед определением класса (оператором class):

```
// максимальная длина имени учетной записи
#define MAXNAME 20
// максимальная длина пароля
#define MAXPASS 10
// структурный тип для хранения учетной записи
struct AccountType
{
char UserName[MAXNAME]; // имя
int PassLen; // длина пароля
char UserPass[MAXPASS]; // пароль
bool Block; // признак блокировки учетной записи администратором
/* признак включения администратором ограничений на выбираемые пользователями
пароли */
bool Restrict;
};
extern AccountType UserAcc; // структура для хранения одной учетной записи
// файловая переменная для чтения и записи в файл с учетными записями
extern fstream AccFile;
extern unsigned RecCount; // номер текущей учетной записи
```

9. Добавить определения глобальных констант и переменных модуля с главной формой (.cpp) после определения переменной Form1:

```
// имя файла с учетными записями пользователей
#define SECFILE "security.db"
// структура для хранения одной учетной записи
AccountType UserAcc;
// файловая переменная для чтения и записи в файл с учетными записями
fstream AccFile;
// номер текущей учетной записи
unsigned RecCount;
```

10. С помощью закладки Events инспектора объектов добавить программный код для обработки создания главной формы программы Form1 (события OnCreate):

```
/* если файл с учетными записями пользователей не существует (первый запуск
программы) */
  if(!FileExists(SECFILE))
  { AccFile.open(SECFILE,ios::out|ios::binary); // создание нового файла
     // подготовка учетной записи администратора
    strcpy(UserAcc.UserName,"ADMIN");
    strcpy(UserAcc.UserPass,"");
    UserAcc.PassLen=0;
    UserAcc.Block=false;
    UserAcc.Restrict=true;
    // запись в файл
    AccFile.write((const char*)&UserAcc,sizeof(UserAcc));
    // закрытие файла
```

```
AccFile.close(); }
```

11. Добавить программный код для обработки нажатия кнопки «Вход» на главной форме программы (события OnClick):

```
// запрос и проверка имени учетной записи и пароля
static unsigned EnterCount=0; // счетчик попыток входа в программу
// отображение формы для ввода имени и пароля
if(PasswordDlg->ShowModal()==mrOk)
{// если повторная попытка входа, то сброс признака конца файла и его закрытие
 if(AccFile.is_open())
 { AccFile.clear();
  AccFile.close(); }
 // открытие файла для чтения и записи
 AccFile.open(SECFILE,ios::in|ios::out|ios::binary);
 // сброс номера текущей учетной записи
 RecCount=0;
// чтение учетных записей и сравнение имен из них с введенным пользователем именем
 while(!AccFile.eof())
 { AccFile.read((char*)&UserAcc,sizeof(UserAcc));
   RecCount++;
  // прекращение чтения из файла, если обнаружено совпадение имен
  if(PasswordDlg->Login->Text==UserAcc.UserName) break; }
 // если совпадения не найдено (достигнут конец файла)
 if(AccFile.eof())
   // генерация исключительной ситуации (пользователь не зарегистрирован)
   throw Exception("Вы не зарегистрированы!");
 // если пароль отсутствует (первый вход пользователя в программу)
 else if(!UserAcc.PassLen)
  // отображение формы для ввода (смены) пользователем пароля
  if(Form5->ShowModal()==mrOk)
  {// смещение к началу текущей учетной записи в файле
   AccFile.seekp((RecCount-1)*sizeof(UserAcc),ios::beg);
   // добавление введенного пароля и его длины в учетную запись
   strcpy(UserAcc.UserPass,Form5->Edit1->Text.c_str());
   UserAcc.PassLen=Form5->Edit1->Text.Length();
    // запись в файл
   AccFile.write((const char*)&UserAcc,sizeof(UserAcc)); }
  // если пользователь не ввел пароль, то выход из функции
  else return;
 // если пользователь уже имел пароль
 else
 { // сравнение пароля из учетной записи и введенного пароля
  if(strcmp(UserAcc.UserPass,PasswordDlg->Password->Text.c_str()))
   // если пароли не совпадают и число попыток превысило 2
   if(++EnterCount>2)
   {// скрытие кнопки «Вход»
    Button1->Visible=false;
    // генерация исключительной ситуации (вход в программу невозможен)
    throw Exception("Вход в программу невозможен!"); }
   // если пароли не совпадают и число попыток не превысило 2
   else
     // генерация исключительной ситуации (пользователь ввел неверный пароль)
     throw Exception("Неверный пароль!");
```

```
          // если пароли совпадают, то продолжение работы
          else; }
        // если учетная запись заблокирована администратором
        if(UserAcc.Block)
          // генерация исключительной ситуации (пользователь заблокирован)
          throw Exception("Вы заблокированы!");
        // проверка полномочий пользователя
        // если пользователь является администратором
        if(!strcmp(UserAcc.UserName,"ADMIN"))
        {// снятие блокировки с команд меню «Все пользователи» и «Новый пользователь»
          All->Enabled=true;
          New->Enabled=true;
          // закрытие файла с учетными записями
          AccFile.close(); }
      // снятие блокировки с команды меню «Смена пароля» (для всех пользователей)
        Change->Enabled=true;
        // скрытие кнопки «Вход»
        Button1->Visible=false; }
```

12. Добавить программный код для обработки отображения окна входа в программу PasswordDlg (события OnShow):

```
      // очистка редакторов для ввода имени учетной записи и пароля
      Login->Text="";
      Password->Text="";
      // установка фокуса ввода на редактор для ввода имени учетной записи
      ActiveControl=Login;
```

13. Добавить программный код для обработки возможности закрытия окна входа в программу (события OnCloseQuery):

```
// если нажата кнопка “Ok”, то окно закрывается только, если введено имя учетной записи
if(ModalResult==mrOk) CanClose=Login->Text!="";
```

14. Определения на глобальном уровне модуля окна ввода (смены) пароля Unit5:

```
// подключение заголовочного файла модуля главной формы
#include "Unit1.h"
// пример функции для проверки ограничений на введенный пользователем пароль (Pass)
// (проверяется наличие в пароле строчных и прописных букв, цифр и знаков препинания)
bool CheckPassword(const AnsiString& Pass)
{
 // признаки наличия в пароле требуемых групп символов
bool High=false,Low=false,Digit=false,Punct=false;
  for(int i=1;i<=Pass.Length();i++)
  {
    /* проверка принадлежности очередного символа пароля одной из требуемых групп и
изменение соответствующего признака */
    High|=IsCharUpper(Pass[i]);
    Low|=IsCharLower(Pass[i]);
    Digit|=isdigit(Pass[i]);
    Punct|=ispunct(Pass[i]);
  }
  // формирование результата выполнения функции проверки
  return High && Low && Digit && Punct;
}
```

15. Обработка отображения окна смены пароля Form5 (события OnShow):

```
      // очистка редакторов для ввода и подтверждения пароля
```

```
      Edit1->Text="";
      Edit2->Text="";
       // установка фокуса ввода на редактор для ввода пароля
      ActiveControl=Edit1;
```

16. Обработка возможности закрытия окна смены пароля (события OnCloseQuery):

```
    // если нажата кнопка “Ok”
    if(ModalResult==mrOk)
      // если новый пароль не совпадает с его подтверждением
      if(Edit1->Text!=Edit2->Text)
      {// вывод сообщения об ошибке
        ShowMessage("Пароли должны совпадать!");
         // установка фокуса ввода на редактор для ввода пароля
        ActiveControl=Edit1;
        // запрет закрытия окна
        CanClose=false; }
  // если в введенном пароле не соблюдены установленные администратором ограничения
      else if(UserAcc.Restrict && !CheckPassword(Edit1->Text))
          {// вывод сообщения об ошибке
           ShowMessage("Пароль не соответствует ограничениям!");
           // установка фокуса ввода на редактор для ввода пароля
           ActiveControl=Edit1;
           // запрет закрытия окна
           CanClose=false; }
     // если ошибок нет, то окно может быть закрыто
     else CanClose=true;
```

17. Обработка команды «Смена пароля» меню главной формы (события OnClick):

```
 /* если программа в режиме администратора (команда «Все пользователи» разблокирована) */
  if(All->Enabled)
    {// открытие файла с учетными записями для чтения и записи
     AccFile.open(SECFILE,ios::in|ios::out|ios::binary);
     // сброс номера текущей учетной записи
     RecCount=0;
     // чтение учетной записи администратора (первой учетной записи)
     AccFile.read((char*)&UserAcc,sizeof(UserAcc));
     RecCount++; }
  // отображение формы для смены пароля
  if(Form5->ShowModal()==mrOk)
  {// смещение к началу текущей учетной записи в файле
   AccFile.seekp((RecCount-1)*sizeof(UserAcc),ios::beg);
   // помещение в учетную запись нового пароля и его длины
   strcpy(UserAcc.UserPass,Form5->Edit1->Text.c_str());
   UserAcc.PassLen=Form5->Edit1->Text.Length();
   // запись в файл
   AccFile.write((const char*)&UserAcc,sizeof(UserAcc)); }
  // если программа в режиме администратора, то закрытие файла
  if(All->Enabled) AccFile.close();
```

18. Обработка команды «Новый пользователь» меню главной формы (события OnClick):

```
  // открытие файла с учетными записями для чтения и записи
  AccFile.open(SECFILE,ios::in|ios::out|ios::binary);
  // отображение окна добавления нового пользователя
```

```
 if(Form4->ShowModal()==mrOk)
 {// смещение к концу файла
  AccFile.seekp(0,ios::end);
  // сохранение в учетной записи введенного имени пользователя
  strcpy(UserAcc.UserName,Form4->UserName->Text.c_str());
  // сохранение в новой учетной записи пустого пароля
  strcpy(UserAcc.UserPass,"");
  UserAcc.PassLen=0;
  // установка признака отсутствия блокирования новой учетной записи
  UserAcc.Block=false;
  // установка признака ограничений на выбираемые пароли
  UserAcc.Restrict=true;
  // запись в файл
  AccFile.write((const char*)&UserAcc,sizeof(UserAcc)); }
 // закрытие файла
 AccFile.close();
```

19. Определения на глобальном уровне модуля окна добавления нового пользователя Unit4:

```
// подключение заголовочного файла модуля главной формы
#include "Unit1.h"
```

20. Обработка отображения окна добавления нового пользователя Form4 (события OnShow):

```
 // установка фокуса ввода на редактор для ввода имени пользователя
 ActiveControl=UserName;
 // очистка редактора для ввода имени нового пользователя
 UserName->Text="";
```

21. Обработка проверки возможности закрытия окна добавления нового пользователя (события OnCloseQuery):

```
// если нажата кнопка “Ok”
if(ModalResult==mrOk)
 // если имя пользователя не введено
 if(UserName->Text=="")
 {// запрет закрытия окна
  CanClose=false;
  // установка фокуса ввода на редактор для ввода имени пользователя
  ActiveControl=UserName; }
 else
 {// смещение к началу файла с учетными записями
  AccFile.seekg(0,ios::beg);
  // чтение учетных записей из файла для проверки уникальности введенного имени
  while(!AccFile.eof())
  { AccFile.read((char*)&UserAcc,sizeof(UserAcc));
    // если учетная запись с введенным именем уже существует, то прекращение чтения
    if(UserName->Text==UserAcc.UserName) break; }
  // если учетная запись с введенным именем уже существует (не достигнут конец файла)
  if(!AccFile.eof())
  {// вывод сообщения об ошибке
    ShowMessage(AnsiString("Пользователь ")+Form4->UserName->Text+
       "\nуже зарегистрирован!");
    // запрет закрытия окна
    CanClose=false;
    // установка фокуса ввода на редактор для ввода имени
```

```
    ActiveControl=UserName; }
  // если ошибок нет, то сброс состояния «конец файла»
  else AccFile.clear(); }
```

22. Обработка команды «Все пользователи» меню главной формы (события OnClick):

```
char ch; // вспомогательная символьная переменная
// открытие файла с учетными записями для чтения и записи
AccFile.open(SECFILE,ios::in|ios::out|ios::binary);
// сброс номера текущей учетной записи
RecCount=0;
// чтение первой учетной записи
AccFile.read((char*)&UserAcc,sizeof(UserAcc));
RecCount++;
// отображение имени учетной записи
Form3->UserName->Text=UserAcc.UserName;
// отображение признака блокировки учетной записи
Form3->CheckBox1->Checked=UserAcc.Block;
// отображение признака установленных ограничений на выбираемые пароли
Form3->CheckBox2->Checked=UserAcc.Restrict;
// попытка чтения следующей учетной записи
AccFile.read(&ch,1);
/* если следующей учетной записи нет, то блокирование кнопки «Следующий» в окне
просмотра (редактирования) учетных записей */
Form3->Next->Enabled=!AccFile.eof();
// если достигнут конец файла с учетными записями
if(AccFile.eof())
{ // сброс состояния «конец файла»
  AccFile.clear();
  // смещение к началу первой учетной записи
  AccFile.seekg(0,ios::beg); }
// если конец файла не достигнут, то смещение к началу следующей учетной записи
else AccFile.seekg(sizeof(UserAcc),ios::beg);
// отображение окна просмотра (редактирования) учетных записей
Form3->ShowModal();
// закрытие файла
AccFile.close();
```

23. Определения на глобальном уровне модуля окна просмотра (редактирования) учетных записей Unit3:

```
// подключение заголовочного файла модуля главной формы
#include "Unit1.h"
```

24. Обработка отображения окна просмотра (редактирования) учетных записей Form3 (события OnShow):

```
// установка фокуса ввода на кнопку «Ok»
ActiveControl=Button1;
```

25. Обработка нажатия кнопки «Сохранить» окна просмотра (редактирования) учетных записей (события OnClick):

```
// сохранение в учетной записи сделанных администратором изменений
UserAcc.Block=CheckBox1->Checked;
UserAcc.Restrict=CheckBox2->Checked;
// смещение к началу редактируемой учетной записи в файле
AccFile.seekp((RecCount-1)*sizeof(UserAcc),ios::beg);
// запись в файл
AccFile.write((const char*)&UserAcc,sizeof(UserAcc));
```

```
// смещение для чтения следующей учетной записи
AccFile.seekg(RecCount*sizeof(UserAcc),ios::beg);
```

26. Обработка нажатия кнопки «Следующий» окна просмотра (редактирования) учетных записей (события OnClick):

```
char ch; // вспомогательная символьная переменная
// чтение учетной записи из файла
AccFile.read((char*)&UserAcc,sizeof(UserAcc));
// увеличение номера текущей учетной записи
RecCount++;
// отображение имени учетной записи
UserName->Text=UserAcc.UserName;
// отображение признака блокировки учетной записи
CheckBox1->Checked=UserAcc.Block;
// отображение признака установленных ограничений на выбираемые пароли
CheckBox2->Checked=UserAcc.Restrict;
// попытка чтения следующей учетной записи
AccFile.read(&ch,1);
// если текущая учетная запись является последней
if(AccFile.eof())
{   // очистка состояния файла от признака конца файла
   AccFile.clear();
   // блокировка кнопки «Следующий»
   Next->Enabled=false; }
// если конец файла не достигнут, то смещение к началу следующей учетной записи
else AccFile.seekg(RecCount*sizeof(UserAcc),ios::beg);
```

27. Обработка закрытия главной формы программы (события OnClose):

```
// если файл учетных записей открыт, то сброс признака конца файла и его закрытие
if(AccFile.is_open())
{ AccFile.clear();
  AccFile.close(); }
```

Лабораторная работа №2

1. С помощью команды File | New | Form добавить к проекту дополнительную форму, образец которой приведен в описании лабораторной работы. Разместить на ней необходимые элементы (надпись и редактор).
2. Задать указанные ниже имена элементу созданной формы и самой форме (с помощью инспектора объектов).

Имена (значения свойства Name) элементов форм, используемые в расположенных ниже фрагментах программного кода

Тип и текст элемента	Имя элемента в программе
Форма ввода пароля для расшифрования базы учетных записей (Form6)	
Редактор для ввода пароля	Edit1

3. Дополнительные глобальные константы и переменные модуля с главной формой Unit1:

```
// имя временного (расшифрованного) файла с учетными записями
#define TMPFILE "temp.db"
// вспомогательные файловые переменные
fstream TmpFile1,TmpFile2;
// буфер для чтения и записи в файл
BYTE Buf[128];
// дескриптор криптопровайдера
HCRYPTPROV hProv=0;
```

```
// дескриптор ключа шифрования (расшифрования)
HCRYPTKEY hKey;
// дескриптор хеш-значения
HCRYPTHASH hHash;
```

4. Дополнительные действия при обработке создания главной формы Form1 (события OnCreate):

```
 DWORD Param=CRYPT_MODE_CBC, // режим шифрования (расшифрования)
       Len; // длина блока данных
 // создание формы для ввода парольной фразы
 TForm6* Form6=new TForm6(Application);
 // блок с возможной генерацией исключительных ситуаций
 try
 {// получение дескриптора криптопровайдера
  if(!CryptAcquireContext(&hProv,NULL,NULL,PROV_RSA_FULL,0))
/* если пользователь еще не зарегистрирован в криптопровайдере, то создание для него
контейнера ключей */
    if((unsigned)GetLastError()==NTE_BAD_KEYSET)
     CryptAcquireContext(&hProv,NULL,NULL,PROV_RSA_FULL,CRYPT_NEWKEYSET);
   // если доступ к криптопровайдеру невозможен, то генерация исключительной ситуации
    else throw Exception("Ошибка при доступе к CryptoAPI!");
   // отображение формы для ввода парольной фразы для расшифрования файла Form6
   if(Form6->ShowModal()!=mrOk)
    // если парольная фраза не введена, то генерация исключительной ситуации
    throw Exception("Работа программы невозможна!");
    // создание пустого хеш-значения
   CryptCreateHash(hProv,CALG_SHA,0,0,&hHash);
    // хеширование введенной пользователем парольной фразы
   CryptHashData(hHash,Form6->Edit1->Text.c_str(),
     Form6->Edit1->Text.Length(),0);
    // уничтожение формы для ввода парольной фразы
   delete Form6;
    // генерация ключа расшифрования из хеш-значения парольной фразы
   CryptDeriveKey(hProv,CALG_RC2,hHash,CRYPT_EXPORTABLE,&hKey);
    // установка режима расшифрования
   CryptSetKeyParam(hKey,KP_MODE,(BYTE*)&Param,0);
    // разрушение хеш-значения
   CryptDestroyHash(hHash); }
 // обработка исключительных ситуаций
 catch(Exception& E)
 { // уничтожение формы для ввода парольной фразы
    delete Form6;
    // освобождение дескриптора криптопровайдера
    if(hProv) CryptReleaseContext(hProv,0);
    // вывод сообщения об ошибке
   Application->ShowException(&E);
    // завершение работы программы
   Application->Terminate();
   return; }
/* далее в программе при указании имени файла с учетными записями пользователей
следует использовать константу TMPFILE вместо константы SECFILE */
. . .
```

```
/* если файл с учетными записями существует (второй и последующие запуски программы), то он должен быть расшифрован */
  else
  { // открытие зашифрованного файла для чтения
    TmpFile1.open(SECFILE,ios::in|ios::binary);
    // создание временного файла
    TmpFile2.open(TMPFILE,ios::out|ios::binary);
    // цикл расшифрования данных и записи их во временный файл
    do
    { // чтение порции зашифрованных данных из файла
      TmpFile1.read((char*)Buf,sizeof(Buf));
      // получение фактической длины прочитанных данных
      Len=TmpFile1.gcount();
      // расшифрование прочитанных данных
      CryptDecrypt(hKey,0,TmpFile1.eof(),0,Buf,&Len);
      // запись во временный файл
      TmpFile2.write((const char*)Buf,Len); }
    while(!TmpFile1.eof());
    // сброс признака конца файла
    TmpFile1.clear();
    // закрытие файлов
    TmpFile1.close();
    TmpFile2.close();
    // проверка правильности расшифрования файла с учетными записями
    // открытие временного файла для чтения
    TmpFile2.open(TMPFILE,ios::in|ios::binary);
    // чтение первой учетной записи (администратора)
    TmpFile2.read((char*)&UserAcc,sizeof(UserAcc));
    // закрытие временного файла
    TmpFile2.close();
    // если имя первой учетной записи не совпадает с ADMIN
    if(strcmp(UserAcc.UserName,"ADMIN"))
    { // разрушение ключа расшифрования
      CryptDestroyKey(hKey);
       // освобождение дескриптора криптопровайдера
      CryptReleaseContext(hProv,0);
      // вывод сообщения об ошибке
      ShowMessage("Неверный ключ расшифрования!");
      // удаление временного файла
      remove(TMPFILE);
      // завершение работы программы
      Application->Terminate();
      return; } }
```

5. Дополнительные действия при обработке закрытия главной формы программы (события OnClose):

```
  DWORD Len; // длина блока данных
  // открытие временного файла для чтения
  if(AccFile.is_open()) AccFile.close();
  TmpFile2.open(TMPFILE,ios::in|ios::binary);
  // создание нового зашифрованного файла с учетными записями
  TmpFile1.open(SECFILE,ios::out|ios::binary);
  // цикл шифрования данных и записи их в файл
```

```
do
{// чтение порции данных из временного файла
  TmpFile2.read((char*)Buf,sizeof(Buf));
  // получение фактической длины прочитанных данных
  Len=TmpFile2.gcount();
  // шифрование данных
  CryptEncrypt(hKey,0,TmpFile2.eof(),0,Buf,&Len,sizeof(Buf));
  // запись зашифрованных данных во вновь созданный файл
  TmpFile1.write((const char*)Buf,Len); }
while(!TmpFile2.eof());
// закрытие файлов
TmpFile1.close();
TmpFile2.close();
// удаление временного файла
remove(TMPFILE);
// разрушение ключа шифрования
CryptDestroyKey(hKey);
// освобождение дескриптора криптопровайдера
CryptReleaseContext(hProv,0);
```