Лабораторная работа №1

Разработка программы разграничения полномочий пользователей на основе парольной аутентификации

Содержание задания

1. Программа должна обеспечивать работу в двух режимах: администратора (пользователя с фиксированным именем ADMIN) и обычного пользователя.
2. В режиме администратора программа должна поддерживать следующие функции (при правильном вводе пароля):
   - смена пароля администратора (при правильном вводе старого пароля);
   - просмотр списка имен зарегистрированных пользователей и установленных для них параметров (блокировка учетной записи, включение ограничений на выбираемые пароли) – всего списка целиком в одном окне или по одному элементу списка с возможностью перемещения к его началу или концу;
   - добавление уникального имени нового пользователя к списку с пустым паролем (строкой нулевой длины);
   - блокирование возможности работы пользователя с заданным именем;
   - включение или отключение ограничений на выбираемые пользователем пароли (в соответствии с индивидуальным заданием, определяемым номером варианта);
   - завершение работы с программой.
3. В режиме обычного пользователя программа должна поддерживать только функции смены пароля пользователя (при правильном вводе старого пароля) и завершения работы, а все остальные функции должны быть заблокированы.
4. После своего запуска программа должна запрашивать у пользователя в специальном окне входа ввод его имени и пароля. При вводе пароля его символы всегда должны на экране заменяться символом ‘*’.
5. При отсутствии введенного в окне входа имени пользователя в списке зарегистрированных администратором пользователей программа должна выдавать соответствующее сообщение и предоставлять пользователю возможность повторного ввода имени или завершения работы с программой.
6. При неправильном вводе пароля программа должна выдавать соответствующее сообщение и предоставлять пользователю возможность повторного ввода. При трехкратном вводе неверного пароля работа программы должна завершаться.
7. При первоначальном вводе пароля (обязательном при первом входе администратора или пользователя с зарегистрированным ранее администратором именем) и при дальнейшей замене пароля программа должна просить пользователя подтвердить введенный пароль путем его повторного ввода.
8. Если выбранный пользователем пароль не соответствует требуемым ограничениям (при установке соответствующего параметра учетной записи пользователя), то программа должна выдавать соответствующее сообщение и предоставлять пользователю возможность ввода другого пароля, завершения работы с программой (при первом входе данного пользователя) или отказа от смены пароля.
9. Информация о зарегистрированных пользователях, их паролях, отсутствии блокировки их работы с программой, а также включении или отключении ограничений на выбираемые пароли должна сохраняться в специальном файле. При первом запуске программы этот файл должен создаваться автоматически и содержать информацию только об администраторе, имеющем пустой пароль.
10. Интерфейс с программой должен быть организован на основе меню, обязательной частью которого должно являться подменю «Справка» с командой «О программе». При выборе этой команды должна выдаваться информация об авторе программы и выданном

индивидуальном задании. Интерфейс пользователя программы может также включать панель управления с дублирующими команды меню графическими кнопками и строку состояния.

11. Для реализации указанных в пунктах 2-3 функций в программе должны использоваться специальные диалоговые формы, позволяющие пользователю (администратору) вводить необходимую информацию.
12. Программа для выполнения лабораторной работы составляется на основе выбранного студентом варианта Указаний по выполнению лабораторной работы (с учетом рекомендаций по использованию средств выбранного языка программирования, приведенных в конце этого описания лабораторной работы), после чего в исходный код созданной программы вносятся изменения в соответствии с индивидуальным заданием студента.
13. Все файлы проекта программы, включая файл с исполнимым кодом – exe-файлом – программы помещаются в один архив и отсылаются лектору.

Индивидуальные варианты заданий (ограничения на выбираемые пароли)

1. Длина не меньше минимальной длины, устанавливаемой администратором и сохраняемой в учетной записи пользователя.
2. Наличие строчных и прописных букв.
3. Наличие букв и цифр.
4. Наличие букв и знаков препинания.
5. Наличие цифр и знаков препинания.
6. Наличие букв и знаков арифметических операций.
7. Наличие цифр и знаков арифметических операций.
8. Наличие латинских букв и символов кириллицы.
9. Наличие букв, цифр и знаков препинания.
10. Наличие латинских букв, символов кириллицы и цифр.
11. Наличие латинских букв, символов кириллицы и знаков препинания.
12. Наличие строчных и прописных букв, а также цифр.
13. Наличие строчных и прописных букв, а также знаков препинания.
14. Наличие строчных и прописных букв, а также знаков арифметических операций.
15. Наличие латинских букв, символов кириллицы и знаков арифметических операций.
16. Наличие букв, цифр и знаков арифметических операций.
17. Наличие букв, знаков препинания и знаков арифметических операций.
18. Наличие цифр, знаков препинания и знаков арифметических операций.
19. Отсутствие повторяющихся символов.
20. Чередование букв, цифр и снова букв.
21. Чередование букв, знаков препинания и снова букв.
22. Чередование цифр, букв и снова цифр.
23. Отсутствие подряд расположенных одинаковых символов.
24. Чередование цифр, знаков препинания и снова цифр.
25. Чередование цифр, знаков арифметических операций и снова цифр.
26. Несовпадение с именем пользователя.
27. Несовпадение с именем пользователя, записанным в обратном порядке.
28. Наличие строчных и прописных латинских букв, цифр и символов кириллицы.
29. Наличие строчных и прописных букв, цифр и знаков арифметических операций.
30. Наличие латинских букв, символов кириллицы, цифр и знаков арифметических операций.
31. Наличие латинских букв, символов кириллицы, цифр и знаков препинания.
32. Наличие строчных и прописных букв, цифр и знаков препинания.
33. Наличие строчных и прописных символов кириллицы, цифр и знаков препинания.

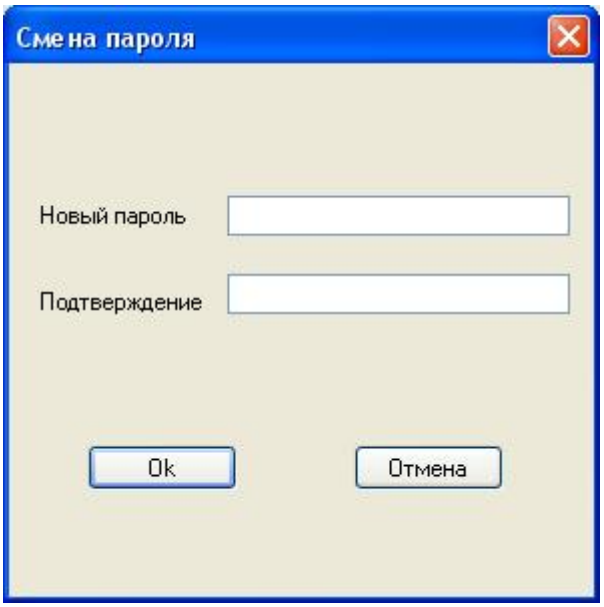
34. Наличие строчных и прописных латинских букв, цифр и знаков арифметических операций.
35. Несовпадение с датой в одном из форматов: дд/мм/гг, дд-мм-гг, дд.мм.гг.

Возможный вид диалоговых форм программы

Окно входа в программу

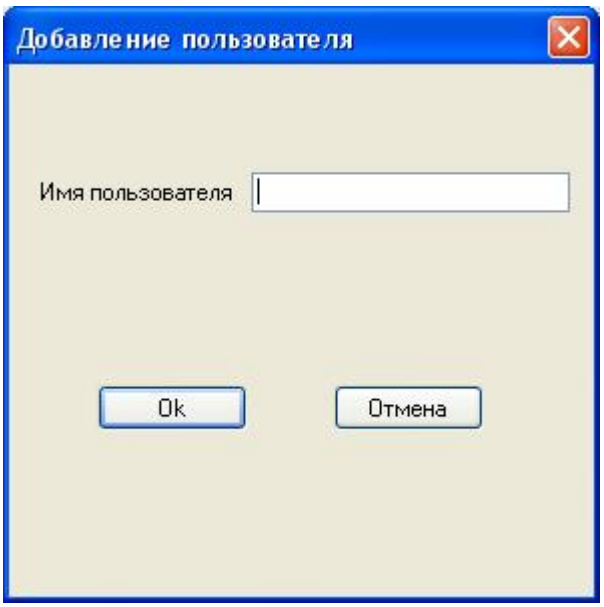
Может быть создано на основе шаблона Password Dialog, выбираемого с помощью команды File | New | Dialogs системы программирования Borland C++ Builder. В указаниях по выполнению лабораторных работ эта форма имеет имя PasswordDlg (C++ в C++ Builder, Object Pascal), Form2 (C#) или IDD_DIALOG1 (C++ в Microsoft Visual Studio).

Окно смены пароля



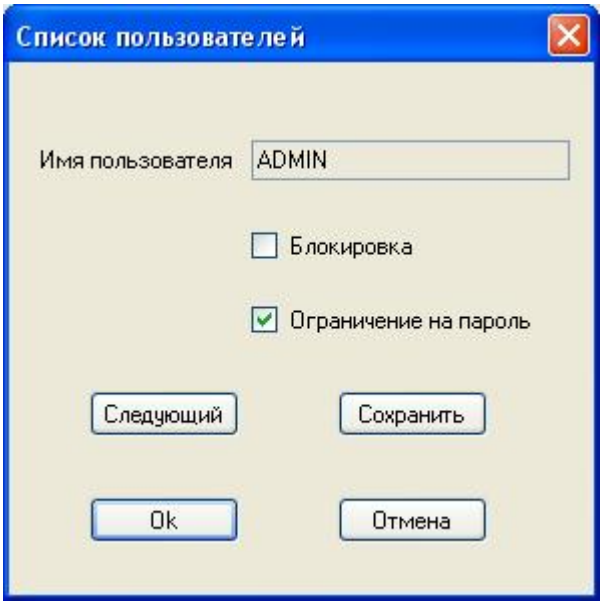
Возможно добавление на форму надписи «Старый пароль» и редактируемой строки для ввода действующего пароля. В указаниях по выполнению лабораторных работ эта форма имеет имя Form5 (C++ в C++ Builder, C#), Form3 (Object Pascal) или IDD_DIALOG2 (C++ в Microsoft Visual Studio).

Окно добавление нового пользователя



Возможно добавление на форму элементов управления для отображения и изменения значений параметров, устанавливаемых администратором для новой учетной записи (блокировка, ограничение на выбираемые пароли). В указаниях по выполнению лабораторных работ эта форма имеет имя Form4 (C++ в C++ Builder, C#, Object Pascal) или IDD_DIALOG3 (C++ в Microsoft Visual Studio).

Окно просмотра (редактирования) учетных записей



Возможно добавление кнопки «Предыдущий» для перехода к предыдущей учетной записи или отображение списка учетных записей пользователей и их параметров в одном окне с помощью компонента StringGrid (группа Additional) системы программирования Borland C++ Builder. В указаниях по выполнению лабораторных работ эта форма имеет имя Form3 (C++ в C++ Builder, C#), Form5 (Object Pascal) или IDD_DIALOG4 (C++ в Microsoft Visual Studio).

Рекомендуемые для разработки программы средства языка C#

1. Тип данных для представления информации об учетной записи пользователя программы:

public struct *Имя_типа*
{
    public byte[] *имя*;
    public byte[] *пароль*;

public int *длина пароля*;
public bool *признак блокировки учетной записи администратором*;
public bool *признак включения ограничений на выбираемые пароли*;

}

2. Объект класса FileStream файлового потокового ввода-вывода (определен в пространстве имен System.IO) для представления файла учетных записей пользователей программы.
3. Методы и свойства класса FileStream для работы с файлом учетных записей:

FileStream(string path, FileMode mode); /* конструктор класса для открытия файла path в режимах Create при создании нового файла и Open при чтении и записи существующего файла */
int Read(byte[] array, int offset, int count); /* чтение из файла count байт в буфер array со смещением offset с возвращением количества фактически прочитанных байт */
void Write(byte[] array, int offset, int count); /* запись в файл count байт из буфера array со смещением offset */
long Seek(long offset, SeekOrigin origin); /* перемещение текущей позиции файла на offset байт относительно его начала (Begin), конца (End) или текущей позиции (Current) */
void Close(): // закрытие файла
long Length; // длина файла
long Position; // текущая позиция в файле

4. Статический метод класса File (определен в пространстве имен System.IO)^

bool Exists(string path); // проверка существования файла path

5. Свойство и методы класса Encoding для преобразования строковых данных и массивов байт:

Encoding Unicode; /* статическое свойство для представления символов в кодировке Unicode */
byte[] GetBytes(string s); // получение массива байт для строки s
string GetString(byte[] bytes, int index, int count); /* преобразование в строку count байт из массива bytes, начиная с позиции index */

6. Свойства и метод класса массив байт:

int Length; // количество элементов массива
void CopyTo(Array array, int index); /* копирование текущего массива в массив array, начиная с позиции index *\

7. Статические методы класса BitConverter для преобразования данных:

byte[] GetBytes(int value); // из целого числа в массив байт
byte[] GetBytes(bool value); // из логического значения в массив байт
int ToInt32(byte[] value, int startIndex); /* из массива байт value, начиная с позиции startIndex, в целое числоо */
bool ToBoolean(byte[] value, int startIndex); /* из массива байт value, начиная с позиции startIndex, в логическое значение */

8. Средства проверки установленных ограничений на выбираемые пароли (статические методы класса Char):

bool IsDigit(char c); // проверка, является ли *c* цифрой
bool IsLower(char c); // проверка, является ли *c* строчной буквой
bool IsUpper(char c); // проверка, является ли *c* прописной буквой
bool IsSymbol(char c); // проверка, является ли *c* математическим символом
bool IsPunctuation(char c); // проверка, является ли *c* знаком препинания
bool IsLetter(char c); // проверка, является ли *c* буквой
char ToUpper(char c); // преобразование буквы *c* в прописную

9. Свойство класса string:

int Length; // длина строки

10. Замена на экране символом ‘*’ символов вводимого пароля:

Свойство PasswordChar компонента TextBox в Microsoft Visual C# (текстовый редактор)=’*’.

Рекомендуемые для разработки программы средства языка Си++

1. Тип данных для представления информации об учетной записи пользователя программы:

Struct {
//имя пользователя – строка в стиле Си (массив символов)
//длина пароля
//пароль пользователя – массив символов
//признак блокировки учетной записи
//признак включения ограничений на выбираемые пароли
} *имя_структуры*;

2. Объект класса fstream файлового потокового ввода-вывода, открытый в двоичном режиме и состоящий из структур приведенного выше типа (определен в заголовочном файле fstream.h) для представления файла учетных записей пользователей программы:

fstream *имя_файловой_переменной*;

3. Методы класса fstream для работы с файлом учетных записей:

/* открытие существующего файла под именем FileName для чтения и записи в двоичном режиме */
void open(const char *FileName, ios::in|ios::out|ios::binary);
// создание нового файла
void open(const char *FileName, ios::out|ios::binary);
! – перегруженная операция, возвращающая true, если последняя операция ввода или вывода завершилась с ошибкой
// проверка существования файла с именем FileName
bool FileExists(const AnsiString& FileName); // функция Borland C++ Builder
BOOL Open(LPCTSTR *lpszFileName*, UINT *nOpenFlags*, CFileException* *pError* = NULL); // метод класса CFile библиотеки MFC Microsoft Visual Studio
/* сброс флага ошибки для потока ввода или вывода (необходим для продолжения работы в программе с этим потоком) */
void clear(int=0);
// закрытие файла
void close();
/* перемещение указателя текущей позиции файла на off байт относительно dir (возможные значения ios::beg, ios::cur, ios::end) */
ostream& seekp(long off, seek_dir dir);
/* чтение данных из файла в буфер buf длины n, равной длине структуры приведенного выше типа */
istream& read(char *buf, int n);
/* запись данных из буфера buf длины n, содержащего структуру приведенного выше типа, в файл */
ostream& write(const char *buf, int n);
// проверка достижения конца файла
bool eof();

4. Средства проверки установленных ограничений на выбираемые пароли (прототипы функций определены в заголовочных файлах string.h и stdlib.h):

/* преобразование объекта класса AnsiString в Borland C++ Builder (значения свойства Text

объекта класса CEdit, соответствующего компоненту диалоговой формы - однострочному редактору) в строку-массив символов, метод класса AnsiString */
char* c_str();
// получение текущей длины строки S
unsigned strlen(const char *S);
int lstrlen(LPCTSTR lpString); //для строки из двухбайтовых символов
int Length() const; // метод класса AnsiString в Borland C++ Builder
int GetLength( ) const; // метод класса CString в Microsoft Visual Studio
/* получение указателя на символ в строке S, с которого начинается первое вхождение подстроки Substr, или NULL, если Substr не входит в S */
char* strstr(const char *S, const char *Substr);
const wchar_t *wcsstr(const wchar_t *str, const wchar_t *strSearch);
// преобразование строки S в целое число
int atoi(const char *S);
int _wtoi(const wchar_t *str);
/* получение указателя на первый символ строки s1, совпавший с одним из символов строки s2, или NULL */
char *strpbrk(char *s1, const char *s2);
const wchar_t *wcspbrk(const wchar_t *str, const wchar_t *strCharSet);
/* получение длины начального сегмента s1, состоящего только из символов, входящих в s2, или 0 */
unsigned strspn(const char *s1, const char *s2);
size_t wcsspn(const wchar_t *str, const wchar_t *strCharSet);
/* изменение порядка следования символов строки на обратный (последний становится первым и т.д. */
char *strrev(char *s);
wchar_t *_wcsrev(wchar_t *str);
// получение дубликата строки
char *strdup(const char *s);
wchar_t *_wcsdup(const wchar_t *strSource);
// проверка символа ch
BOOL IsCharAlpha(TCHAR ch); // TRUE, если ch – буква
BOOL IsCharUpper(TCHAR ch); // TRUE, если ch – прописная буква
BOOL IsCharLower(TCHAR ch); // TRUE, если ch – строчная буква
int isalpha(int ch); // true, если ch – латинская буква
int iswalpha(wint_t c); // вариант предыдущей функции для двухбайтовых символов
int isdigit(int ch); // true, если ch –арабская цифра
int iswdigit(wint_t c); // вариант предыдущей функции для двухбайтовых символов
int isupper(int ch); // true, если ch – прописная латинская буква
int iswupper(wint_t c); // вариант предыдущей функции для двухбайтовых символов
int islower(int ch); // true, если ch – строчная латинская буква
int iswlower(wint_t c); // вариант предыдущей функции для двухбайтовых символов
int ispunct(int ch); /* true, если ch – печатаемый символ, не являющийся латинской буквой, цифрой или пробелом */
int iswpunct(wint_t c); // вариант предыдущей функции для двухбайтовых символов
‘+’ ‘-‘ ‘*’ ‘/’ ‘%’ – символы знаков арифметических операций

5. Замена на экране символом ‘*’ символов вводимого пароля:

Свойство PasswordChar компонента Edit в Borland C++ Builder (редактируемая строка)=’*’.
Значение True свойства Password элемента управления Edit Control (текстовый редактор) в Microsoft Visual Studio.

Рекомендуемые для разработки программы средства языка Object Pascal

1. Тип данных для представления информации об учетной записи пользователя программы:

Type З*апись_для_информации_о_пользователе* = Record
//имя – строка в стиле Паскаля ограниченной длины
//пароль – строка в стиле Паскаля ограниченной длины
//признак блокировки учетной записи
//признак включения ограничений на выбираемые пароли
end;

2. Типизированный файл из записей приведенного выше типа для представления файла учетных записей:

Var *Имя_файловой_переменной*:File of З*апись_для_информации_о_пользователе*;

3. Стандартные подпрограммы для работы с файлом учетных записей:

procedure AssignFile(var F:File; FileName: string); { «связывание» файловой переменной F с файлом под именем FileName }
procedure Reset(var F: File); // открытие существующего файла для чтения и записи
function IOResult: Integer; { код ошибки последней операции ввода или вывода (при компиляции с режимом $I-) }
function FileExists(const FileName: string): Boolean; { проверка существования файла с именем FileName }
procedure Rewrite(var F: File); // создание нового файла
procedure CloseFile(var F:File); // закрытие файла
function FileSize(var F:File): Integer; // размер файла в записях
procedure Seek(var F:File; N: Longint); { перемещение указателя текущей позиции файла на запись с номером N (нумерация от 0) }
procedure Read(F:File; V); // чтение записи V из файла F
procedure Write(F:File; V); // запись данных из записи V в файл F
function Eof(F:File):Boolean; // проверка достижения конца файла

4. Средства проверки выполнения установленных ограничений на выбираемые пароли:

function Length(S): Integer; // текущая длина строки S
function Pos(Substr: string; S: string): Integer; { позиция символа в строке S, с которого начинается первое вхождение подстроки Substr, или 0, если Substr не входит в S }
function StrToInt(const S: string): Integer; // преобразование строки S в целое число
function IsCharAlpha(ch:Char):Bool; // TRUE, если ch – буква
function IsCharUpper(ch:Char):Bool; // TRUE, если ch – прописная буква
function IsCharLower(ch:Char):Bool; // TRUE, если ch – строчная буква
['А’..'Я’] – множество прописных букв кириллицы
['а’..'я’] – множество строчных букв кириллицы
['A’..'Z’] – множество прописных латинских букв
['a’..'z’] – множество строчных латинских букв
[‘0’..’9’] – множество цифр
[‘.’,’,’,’-‘,’:’,’;’,’!’,’?’,’(‘,’)’,’”’] – множество знаков препинания
[‘+’,‘-‘,‘*’,’/’,’%’] – множество знаков арифметических операций
in – операция проверки вхождения элемента в множество

5. Замена на экране символом ‘*’ символов вводимого пароля:

Свойство PasswordChar компонента Edit (редактируемая строка):=’*’.