ФГБОУ ВПО «Чувашский Государственный Университет им. И.Н.Ульянова»

Факультет информатики и вычислительной техники

Кафедра вычислительной техники

**Лабораторная работа №6  
“** **ГЕНЕРАТОР СЛУЧАЙНЫХ ЧИСЕЛ CRYPTO API”**

Выполнил: студент группы ИВТ-21-12  
 Чесноков Александр  
 Проверил: Павлов П.В.

Чебоксары, 2015

**Цель работы**

Целью работы является ознакомление с библиотекой CryptoAPI. Применение этой библиотеки для генерации случайной последовательности.

**Текст функции main:**

void main()

{

setlocale(LC\_ALL, "Russian");

cout << "Введите длину генерируемого блока: ";

int size;

cin >> size;

cout << "\n";

CAPI Rand(0, size);

Rand.Generate();

Rand.Out();

cout << "\n\n";

system("Pause");

}

**Текст модуля, реализующего класс, для генерации случайной последовательности:**

class CAPI

{

public:

HCRYPTPROV hProv; //хэндл криптопровайдера иницализированного вызовом функции CryptAcquireContext

int dwLen; //длина генерируемой последовательности в байтах

byte \*pbBuffer; //буфер для хранения результирующих данных

CAPI(HCRYPTPROV HProv, int length)

{

hProv = HProv;

dwLen = length;

pbBuffer = new byte[dwLen];

}

~CAPI()

{

delete[] pbBuffer;

CryptAcquireContext(&hProv, 0, 0, PROV\_RSA\_FULL, CRYPT\_DELETEKEYSET);

}

void Generate()

{

CryptAcquireContext(&hProv, 0, 0, PROV\_RSA\_FULL, CRYPT\_VERIFYCONTEXT | CRYPT\_NEWKEYSET);

CryptGenRandom(hProv, dwLen, pbBuffer);

}

void Out()

{

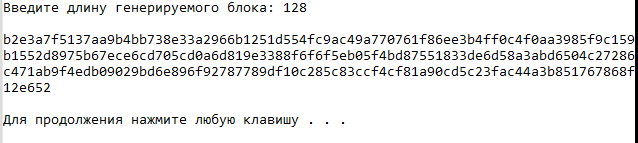
for (DWORD i = 0; i < dwLen; ++i)

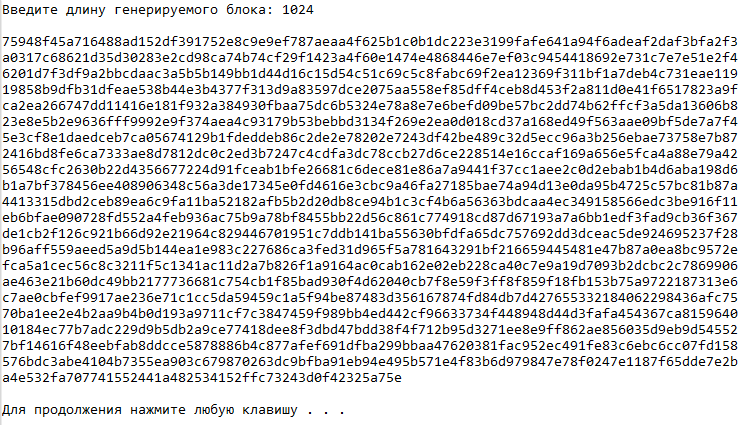
cout << hex << DWORD(pbBuffer[i]);

}

};

**Пример работы программы**





**Выводы**

Мы ознакомились с библиотекой CryptoAPI. Применили ее для генерации случайной последовательности.