
Biblioteki kryptograficzne: C++

Juliusz Kuzyka Rafał Dadura



Język C++ w kontekście cyberbezpieczeństwa

Niskopoziomowy

Wieloplatformowy

Open Source

Biblioteka Crypto++



darmowa i otwartoźródłowa,



implementacje wielu algorytmów kryptograficznych, takich jak AES, DES, RSA, DSA, SHA-256, SHA-512 i wielu innych,



narzędzia do szyfrowania i deszyfrowania danych, a także funkcje do generowania kluczy i bezpiecznej wymiany kluczy,



zaimplementowana zgodnie ze standardami kryptograficznymi, takimi jak FIPS 140-2 i ISO/IEC 18033-2,



łatwa w użyciu i dostarcza wygodny interfejs programistyczny, który umożliwia integrację z różnymi aplikacjami,



zaprojektowana z myślą o bezpieczeństwie i wydajności, a także o minimalnym zużyciu pamięci i czasu procesora,



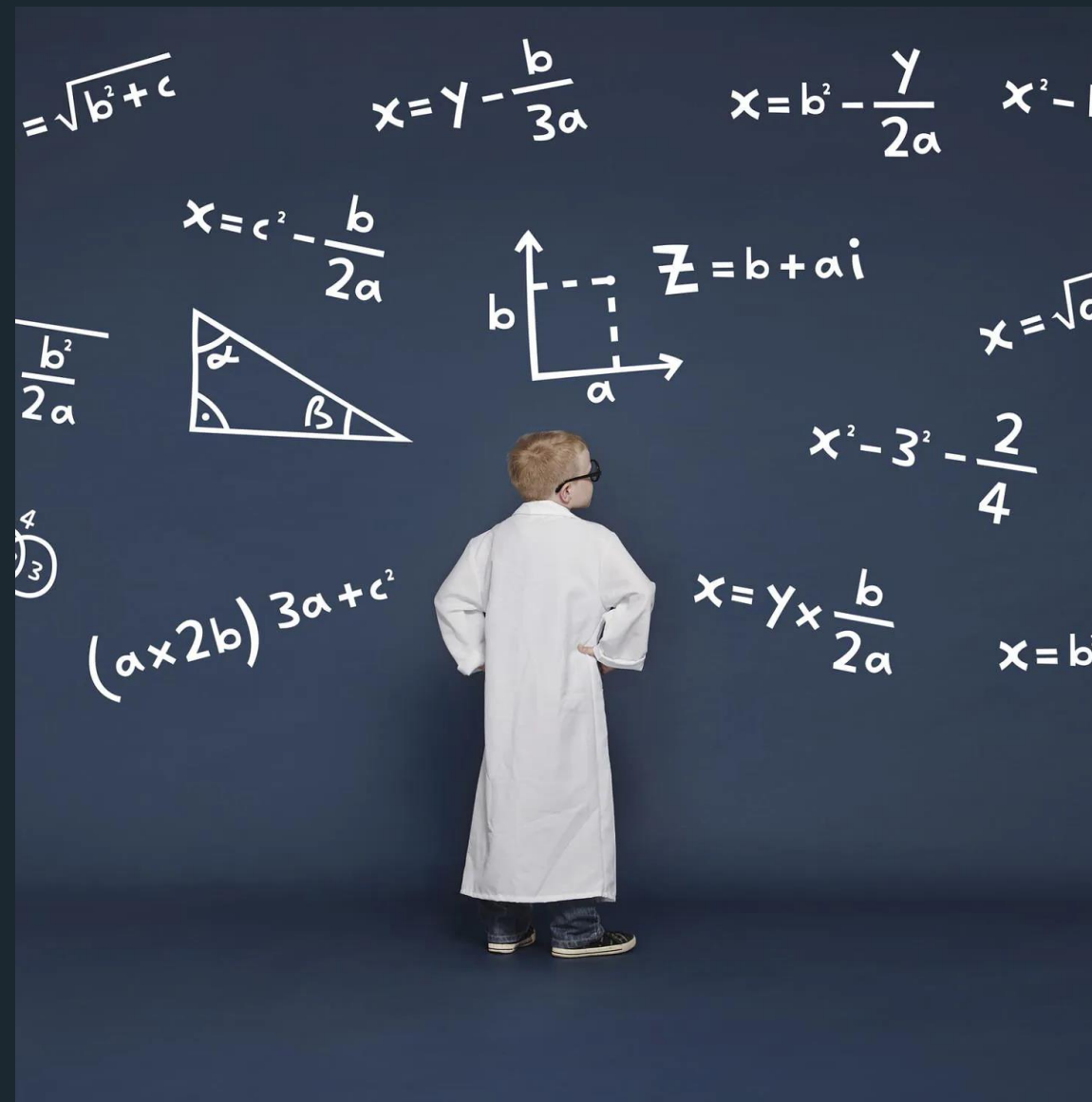
stale rozwijana i aktualizowana

Zadanie 1

(Działania w ciałach skończonych)

Wykorzystując wskazany język programowania wykonaj następujące obliczenia w ciele $GF(p)$, zmieniając rząd wielkości p (p jest liczbą pierwszą), a) $10 < p < 100$; b) p jest dużą liczbą pierwszą, zapisaną na co najmniej 1024 bitach.

- Dodawanie dwóch elementów ciała reszt mod p
- Odejmowanie dwóch elementów ciała reszt mod p
- Pomnożenie dwóch elementów ciała reszt mod p
- Znalezienie elementu odwrotnego do danego elementu ciała reszt mod p



Zadanie 1 - rozwiązanie

- W pierwszej linii kodu zdefiniowana jest liczba p , która jest używana jako ciało $GF(p)$. W przykładzie p jest ustawione na bardzo dużą wartość 1024-bitową liczbę pierwszą.
- W kodzie pokazane są podstawowe operacje na elementach ciała $GF(p)$, w tym dodawanie, odejmowanie i mnożenie.
- W kodzie jest również wykorzystana metoda `InverseMod()`, która służy do znalezienia elementu odwrotnego dla danego elementu w ciele $GF(p)$.

```
#include <iostream>
#include <string>
#include <cryptopp/integer.h>
#include <cryptopp/modarith.h>
#include <cryptopp/osrng.h>

using namespace CryptoPP;

int main()
{
    // Ustawienie parametrów ciała GF(p)
    // p jest wybierane jako liczba pierwsza, większa od 1024 bitów
    Integer p("1129604428034238200456626902470024377103045513327092980570831860428928640361902625505853180754225831580647446925870");

    // Generowanie losowych elementów ciała GF(p)
    AutoSeededRandomPool rng;
    Integer a = Integer(rng, 0, p - 1);
    Integer b = Integer(rng, 0, p - 1);
    std::cout << "a = " << a << std::endl;
    std::cout << "b = " << b << std::endl;

    // Dodawanie dwóch elementów ciała GF(p)
    Integer c = (a + b) % p;
    std::cout << "c = a + b = " << c << std::endl;

    // Odejmowanie dwóch elementów ciała GF(p)
    Integer d = (a - b + p) % p;
    std::cout << "d = a - b = " << d << std::endl;

    // Mnożenie dwóch elementów ciała GF(p)
    Integer e = (a * b) % p;
    std::cout << "e = a * b = " << e << std::endl;

    // Znalezienie elementu odwrotnego do danego elementu ciała GF(p)
    try {
        Integer inv_a = a.InverseMod(p);
        std::cout << "Inverse of a = " << inv_a << std::endl;
    }
    catch (const Exception& e) {
        std::cerr << "Element a nie ma odwrotności w ciele GF(p): " << e.what() << std::endl;
    }

    return 0;
}
```

Zadanie 1 - output

```
Microsoft Visual Studio Debug Console
a = 22692704055171591171035707813756140933684822225354438165949236749677787694998977372430344885507331928258849875918887
882711684925621358676785758061011671603784344130507153733480277366589444751681178923379098905688817827154976141041707870
791666133614152515448361460206609790325181014347349335497657983562579.
b = 15403761769356407700186241852782573936725987103205554509132374534108700959581945989729926462578187531597566724190189
225146192625983362077311476774240611249783562070185836825754753087794925526165179723204341077069373037355802940013274406
674878654770589214913531471552423660997711390697734439616490959330071.
c = a + b = 380964658245279988712219496665387148704108093285599926750816112837864886545809233621602713480855194598564166
001090771078578775516047207540972348352522828535679062006929905592350304543843702778463586465834399827581908645107790810
54982277466544788384741730361892931759033451322892405045083775114148942892650.
d = a - b = 728894228581518347084946596097356699695883512214888365681686221556908673541703138270041842292914439666128315
172869865756549229963799659947428128677106035400078206032131690772552427879451922551599920017475782861944478979917320102
8433464116787478843563300534829988654186129327469623649614895881167024232508.
e = a * b = 416825678025385340011148376960868688491416182377634332735156552544095398449781419308697584606687416706776543
637048083206071474547569529466195764738283021415920932611046707210790396616353102811649620629896229837053738100394507257
86295757440225602173458652361770596126953169470709835041955318040555451955637.
Inverse of a = 39345906838507933149256153250795669160415530528074098886357399575665388242491961296376250558811343074722
057024461100241406400167387417376096221444086233709151872681617684232522467568619554081944007247842870922719130417898212
91375750935715167452732565579125565467431425761940260950007690690645856038121072.

C:\Users\lafar\source\repos\ProjectCryptopp\x64\Debug\ProjectCryptopp.exe (process 19452) exited with code 0.
To automatically close the console when debugging stops, enable Tools->Options->Debugging->Automatically close the conso
le when debugging stops.
Press any key to close this window . . .
```

Zadanie 2

(Generator liczb pseudolosowych)

Zaprezentować działanie generatorów liczb w danym języku np. PRNG



Zadanie 2 - rozwiązanie

- W funkcji **main()** obiekt **AutoSeededRandomPool** jest tworzony, co inicjuje generator liczb losowych.
- Następnie w pętli **for**, generator losowy jest używany, aby wygenerować 10 liczb pseudolosowych.
- Na końcu funkcji **main()**, zwracana jest wartość **0**, co oznacza, że program zakończył się bez problemów.

```
#include <iostream>
#include <cryptopp/osrng.h>
#include <cryptopp/cryptlib.h>

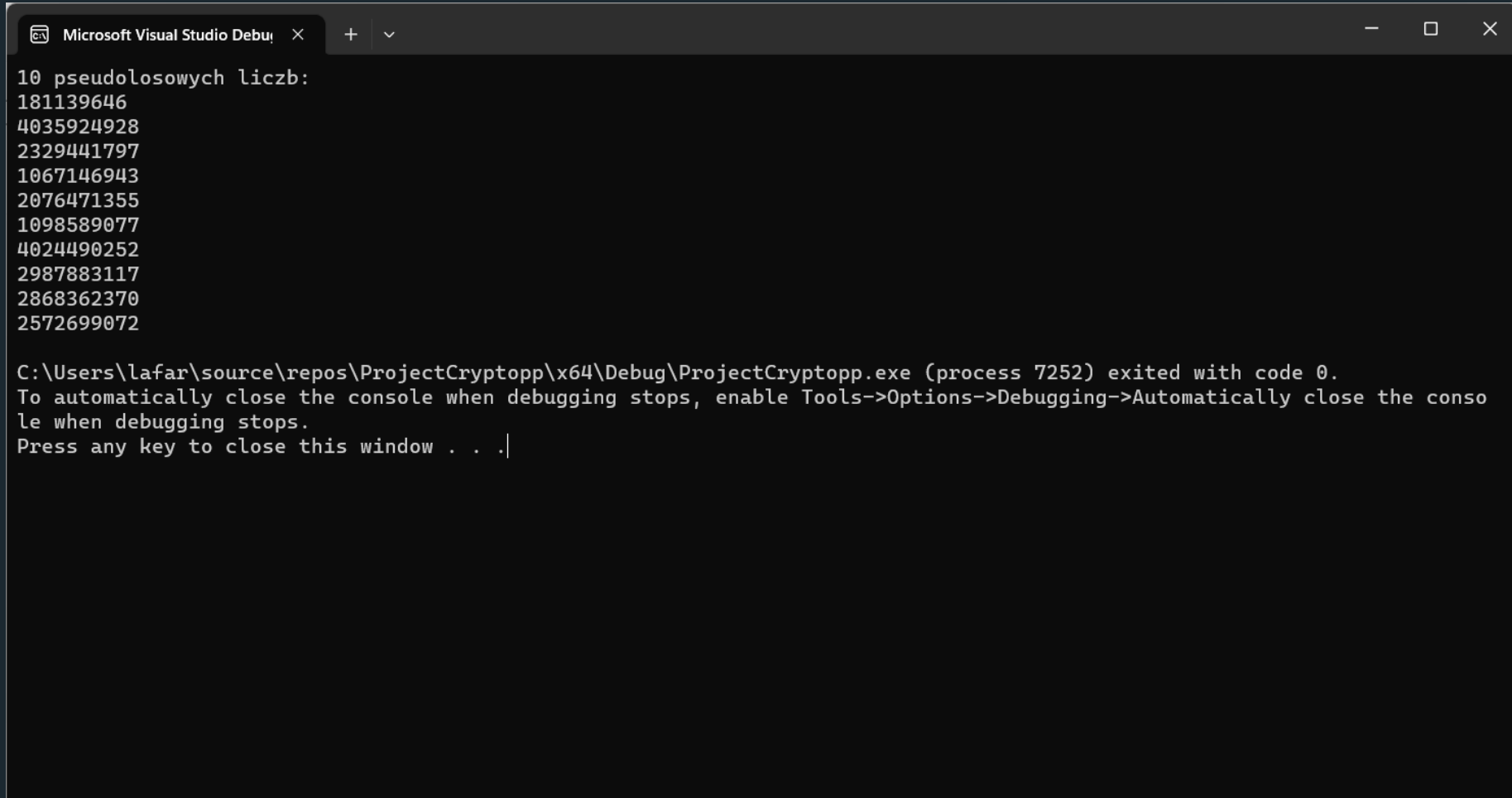
using namespace std;
using namespace CryptoPP;

int main()
{
    AutoSeededRandomPool prng;

    cout << "10 pseudolosowych liczb:" << endl;
    for (int i = 0; i < 10; i++) {
        byte buffer[4];
        prng.GenerateBlock(buffer, sizeof(buffer));
        unsigned int randomNum = 0;
        for (int j = 0; j < sizeof(buffer); j++) {
            randomNum |= static_cast<unsigned int>(buffer[j]) << (j * 8);
        }
        cout << randomNum << endl;
    }

    return 0;
}
```


Zadanie 2 - output



The screenshot shows a 'Microsoft Visual Studio Debug Console' window. The title bar includes the Visual Studio icon, the text 'Microsoft Visual Studio Debug Console', and standard window controls (close, maximize, minimize). The console content displays the output of a program, starting with a prompt and ten lines of pseudorandom numbers. It then shows a message about the program exiting with code 0 and instructions on how to automatically close the console. The text ends with a prompt to press any key to close the window.

```
Microsoft Visual Studio Debug Console
10 pseudolosowych liczb:
181139646
4035924928
2329441797
1067146943
2076471355
1098589077
4024490252
2987883117
2868362370
2572699072

C:\Users\lafar\source\repos\ProjectCryptopp\x64\Debug\ProjectCryptopp.exe (process 7252) exited with code 0.
To automatically close the console when debugging stops, enable Tools->Options->Debugging->Automatically close the console when debugging stops.
Press any key to close this window . . .|
```

Zadanie 3

(Szyfrowanie i odszyfrowanie) Utwórz plik o nazwie zawierającej swoje Imię_Nazwisko z rozszerzeniem txt (np. Jan_Kowalski.txt). W pliku wpisz tekst: To wiadomość do zaszyfrowania . Następnie wygeneruj klucz algorytmu AES-128 i kolejno wykonaj szyfrowanie i odszyfrowanie pliku, gdy:

- A) Klucz i plik są poprawne
- B) Wystąpił błąd w zaszyfrowanym pliku
- C) Wystąpił błąd w kluczu przy odszyfrowaniu



Zadanie 3 - rozwiązanie a)

- Ten kod implementuje algorytm szyfrowania CBC (Cipher Block Chaining) z użyciem algorytmu szyfrowania AES.
- Generowany jest klucz i wektor inicjalizacyjny za pomocą klasy `AutoSeededRandomPool`.
- Klasa `CBC_Mode<AES>::Encryption` używana jest do szyfrowania tekstu jawnej (plaintext) przy użyciu klucza i wektora inicjalizacyjnego.
- W ostatniej części programu odszyfrowywany jest zaszyfrowany tekst przy użyciu klucza i wektora inicjalizacyjnego.

```
AutoSeededRandomPool prng;
SecByteBlock key(AES::DEFAULT_KEYLENGTH);
prng.GenerateBlock(key, key.size());
SecByteBlock iv(AES::BLOCKSIZE);
prng.GenerateBlock(iv, iv.size());

// Szyfrowanie z IV
try {
    CBC_Mode<AES>::Encryption encryptor(key, key.size(), iv);
    StringSource(plaintext, true,
        new StreamTransformationFilter(encryptor,
            new FileSink("encrypted.txt")
        )
    );
    std::cout << "Szyfrowanie zakończone sukcesem" << std::endl;
    std::cout << "Zaszyfrowana wiadomosc: " << std::endl;
    std::cout << "-----" << std::endl;
    std::cout << std::hex;
    for (int i = 0; i < plaintext.size(); i++) {
        std::cout << (int)plaintext[i] << " ";
    }
    std::cout << std::dec << std::endl;
}
catch (const Exception& e) {
    std::cerr << "Bład szyfrowania: " << e.what() << std::endl;
    std::cerr << "Nie udało się zaszyfrować pliku " << filename << std::endl;
    return 1;
}

// Odszyfrowywanie z IV
//key[0] ^= 0x01;
try {
    CBC_Mode<AES>::Decryption decryptor(key, key.size(), iv);
    std::string decryptedtext;
    FileSource("encrypted.txt", true,
        new StreamTransformationFilter(decryptor,
            new StringSink(decryptedtext)
        )
    );
    /*FileSource("modified_encrypted.txt", true,
        new StreamTransformationFilter(decryptor,
            new StringSink(decryptedtext)
        )
    );*/
    std::cout << "Odszyfrowywanie zakończone sukcesem" << std::endl;
    std::cout << "Odszyfrowana wiadomosc: " << std::endl;
    std::cout << "-----" << std::endl;
    std::cout << decryptedtext << std::endl;
}
catch (const Exception& e) {
    std::cerr << "Bład odszyfrowywania: " << e.what() << std::endl;
    return 1;
}

return 0;
```

Zadanie 3 a) - output

```
Microsoft Visual Studio Debug Console
Szyfrowanie zakonczzone sukcesem
Zaszyfrowana wiadomosc:
-----
54 6f 20 77 69 61 64 6f 6d 6f 73 63 20 64 6f 20 7a 61 73 7a 79 66 72 6f 77 61 6e 69 61
Odszyfrowywanie zakonczzone sukcesem
Odszyfrowana wiadomosc:
-----
To wiadomosc do zaszyfrowania

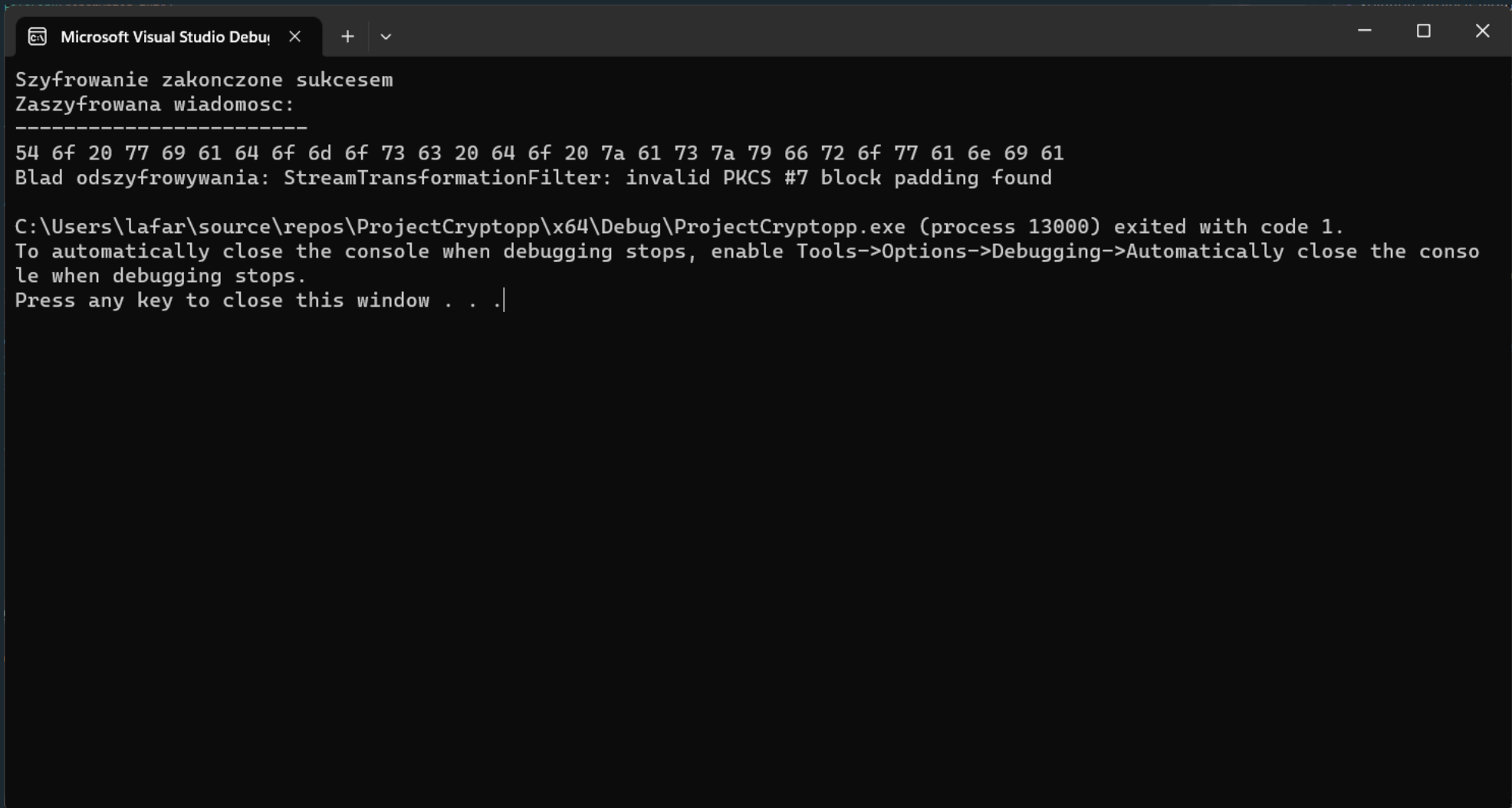
C:\Users\lafar\source\repos\ProjectCryptopp\x64\Debug\ProjectCryptopp.exe (process 17604) ex
To automatically close the console when debugging stops, enable Tools->Options->Debugging->A
le when debugging stops.
Press any key to close this window . . .|
```

Zadanie 3 b) - rozwiązanie

→ Klucz został przesunięty
o jeden bit

```
std::cout << std::dec << std::endl;
}
catch (const Exception& e) {
    std::cerr << "Bład szyfrowania: " << e.what() << std::endl;
    std::cerr << "Nie udało sie zaszyfrowac pliku " << fileName;
    return 1;
}
// Odszyfrowywanie z IV
key[0] ^= 0x01;
try {
    CBC_Mode<AES>::Decryption decryptor(key, key.size(), iv)
    std::string decryptedtext;
    FileSource("encrypted.txt", true,
        new StreamTransformationFilter(decryptor,
            new StringSink(decryptedtext)
        )
    );
    /*FileSource("modified_encrypted.txt", true,
```

Zadanie 3 b) – output



```
Microsoft Visual Studio Debug Console
Szyfrowanie zakończone sukcesem
Zaszyfrowana wiadomosc:
-----
54 6f 20 77 69 61 64 6f 6d 6f 73 63 20 64 6f 20 7a 61 73 7a 79 66 72 6f 77 61 6e 69 61
Bład odszyfrowywania: StreamTransformationFilter: invalid PKCS #7 block padding found

C:\Users\lafar\source\repos\ProjectCryptopp\x64\Debug\ProjectCryptopp.exe (process 13000) exited with code 1.
To automatically close the console when debugging stops, enable Tools->Options->Debugging->Automatically close the console when debugging stops.
Press any key to close this window . . .|
```


Zadanie 3 c) - rozwiązanie

- Ta część kodu modyfikuje pierwszą literę oryginalnej wiadomości odczytanej z pliku i następnie szyfruje ją z wykorzystaniem klucza i wektora inicjalizacyjnego wygenerowanych wcześniej.

```
);  
std::cout << "Szyfrowanie zakonczone sukcesem" << std::endl;  
std::cout << "Zaszyfrowana wiadomosc: " << std::endl;  
std::cout << "-----" << std::endl;  
std::cout << std::hex;  
for (int i = 0; i < plaintext.size(); i++) {  
    std::cout << (int)plaintext[i] << " ";  
}  
std::cout << std::dec << std::endl;  
// Modyfikacja pierwszej litery  
std::string modified_plaintext = plaintext;  
modified_plaintext[0] = 'z';  
// Szyfrowanie zmodyfikowanego plaintextu  
StringSource(modified_plaintext, true,  
    new StreamTransformationFilter(encryptor,  
        new FileSink("modified_encrypted.txt")  
    )  
);  
std::cout << "Szyfrowanie zmodyfikowanego plaintextu zakonczone sukcesem"  
std::cout << "Zmodyfikowana zaszyfrowana wiadomosc: " << std::endl;  
std::cout << "-----" << std::endl;  
std::cout << std::hex;  
for (int i = 0; i < modified_plaintext.size(); i++) {  
    std::cout << (int)modified_plaintext[i] << " ";  
}  
std::cout << std::dec << std::endl;  
}  
catch (const Exception& e) {  
    std::cerr << "Blad szyfrowania: " << e.what() << std::endl;  
    std::cerr << "Nie udalo sie zaszyfrowac pliku " << filename << std::endl;  
    return 1;  
}  
// Odszyfrowywanie z IV  
//key[0] ^= 0x01;  
try {  
    CBC_Mode<AES>::Decryption decryptor(key, key.size(), iv);  
    std::string decryptedtext;  
    StringSource("modified_encrypted.txt", true,  
        new StreamTransformationFilter(decryptor,  
            new FileSink("modified_decrypted.txt")  
        )  
    );  
}
```


Zadanie 3 c) - output

```

Szyfrowanie zakonczone sukcesem
Zaszyfrowana wiadomosc:
-----
54 6f 20 77 69 61 64 6f 6d 6f 73 63 20 64 6f 20 7a 61 73 7a 79 66 72 6f 77 61 6e 69 61
Szyfrowanie zmodyfikowanego plaintextu zakonczone sukcesem
Zmodyfikowana zaszyfrowana wiadomosc:
-----
7a 6f 20 77 69 61 64 6f 6d 6f 73 63 20 64 6f 20 7a 61 73 7a 79 66 72 6f 77 61 6e 69 61
Odszyfrowywanie zakonczone sukcesem
Odszyfrowana wiadomosc:
-----
İ!▼ù■y=0%±=ei¥±izaszyfrowania

C:\Users\lafar\source\repos\ProjectCryptopp\x64\Debug\ProjectCryptopp.exe (process 22440) exited with code 0.
To automatically close the console when debugging stops, enable Tools->Options->Debugging->Automatically close the console when debugging stops.
Press any key to close this window . . .|

```

Zadanie 4

→ (Funkcja skrótu) Wykorzystując plik z poprzedniego ćwiczenia oblicz z niego skrót. Następnie zmień treść pliku na: Ta wiadomość do zaszyfrowania i oblicz skrót takiego pliku. Utwórz też skrót dla pliku z poprzedniego ćwiczenia. Porównaj wartości skrótów.



Zadanie 4 - rozwiązanie

- Funkcja `calculateHash` pobiera nazwę pliku jako argument, otwiera ten plik w trybie binarnym i odczytuje jego zawartość do ciągu `message`. Następnie, korzystając z obiektu SHA256, oblicza skrót haszując zawartość pliku i zwraca wynik jako ciąg znaków w formacie szesnastkowym.
- Funkcja `changeFileContent` służy do zmiany zawartości pliku o podanej nazwie.
- W funkcji `main` obliczany jest skrót pliku "Rafal_Dadura.txt" za pomocą funkcji `calculateHash`. Wynik jest wyświetlany na ekranie.
- Następnie zawartość pliku jest zmieniana na nową wiadomość i obliczany jest skrót nowej zawartości pliku.

```
#include <iostream>
#include <fstream>
#include <iomanip>
#include <cryptopp/sha.h>
#include <cryptopp/filters.h>
#include <cryptopp/hex.h>
#include <cryptopp/files.h>

using namespace std;
using namespace CryptoPP;

string calculateHash(const string& filename) {
    SHA256 hash;
    string digest;
    ifstream file(filename.c_str(), ios::binary);
    if (file) {
        string message((istreambuf_iterator<char>(file)), istreambuf_iterator<char>());
        cout << "Message before hashing: " << message << endl;
        HashFilter filter(hash, new HexEncoder(new StringSink(digest)));
        file.clear(); // resetujemy flagi pliku
        file.seekg(0, ios::beg); // ustawiamy wskaźnik na początek pliku
        FileSource(file, true /* pump all */, new Redirector(filter));
    }
    file.close();
    return digest;
}

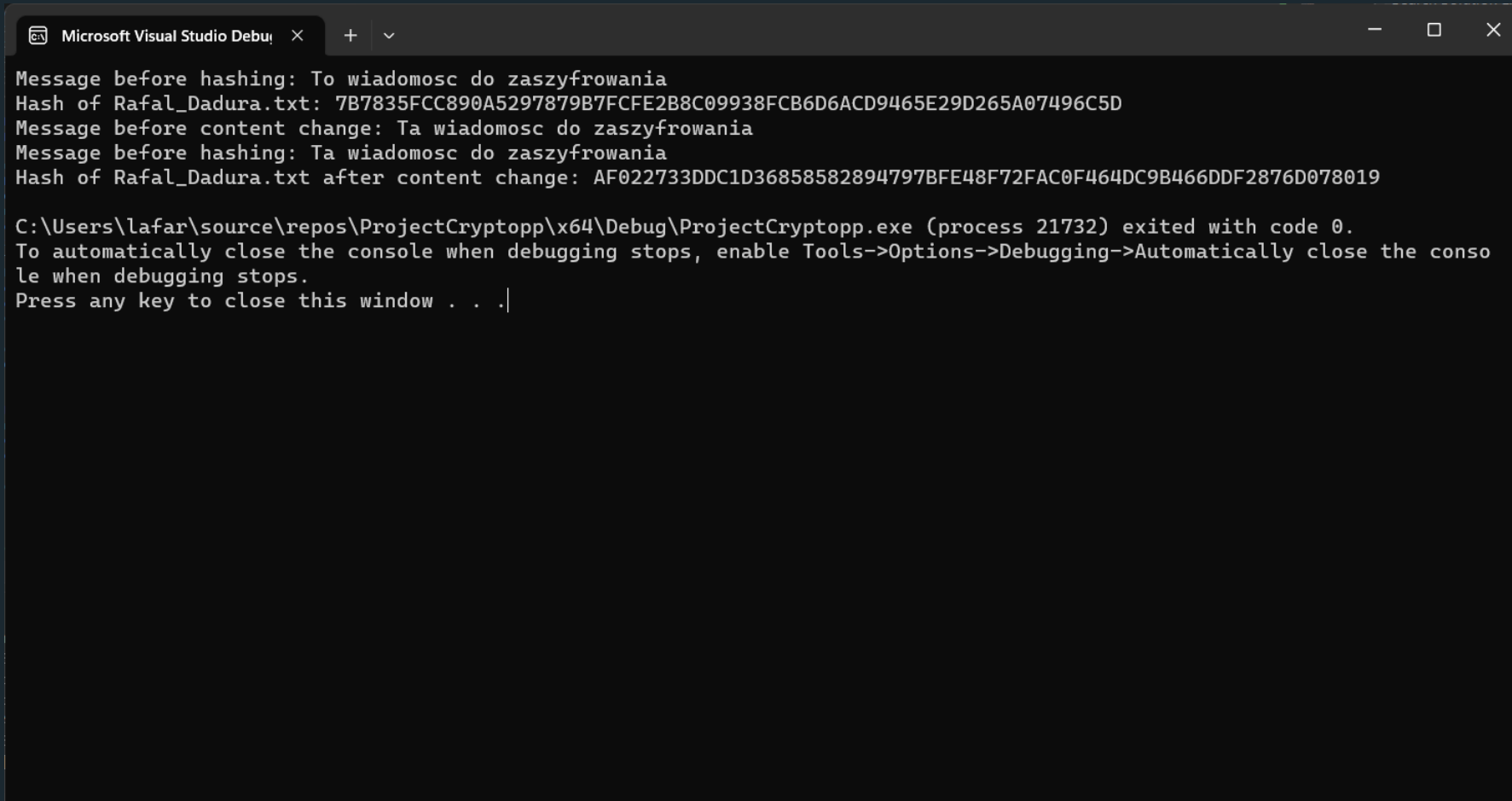
void changeFileContent(const string& filename, const string& newContent) {
    ofstream file(filename.c_str(), ios::binary);
    if (file) {
        file << newContent;
    }
    file.close();
}

int main() {
    // Obliczenie skrótu pliku Rafal_Dadura.txt
    string filename = "Rafal_Dadura.txt";
    string hash = calculateHash(filename);
    cout << "Hash of " << filename << ": " << hash << endl;

    // Zmiana zawartości pliku i obliczenie skrótu nowej zawartości
    string newContent = "Ta wiadomosc do zaszyfrowania";
    cout << "Message before content change: " << newContent << endl;
    changeFileContent(filename, newContent);
    string newHash = calculateHash(filename);
    cout << "Hash of " << filename << " after content change: " << newHash << endl;

    return 0;
}
```

Zadanie 4 - output



```
Microsoft Visual Studio Debug Console
Message before hashing: To wiadomosc do zaszyfrowania
Hash of Rafal_Dadura.txt: 7B7835FCC890A5297879B7FCFE2B8C09938FCB6D6ACD9465E29D265A07496C5D
Message before content change: Ta wiadomosc do zaszyfrowania
Message before hashing: Ta wiadomosc do zaszyfrowania
Hash of Rafal_Dadura.txt after content change: AF022733DDC1D36858582894797BFE48F72FAC0F464DC9B466DDF2876D078019

C:\Users\lafar\source\repos\ProjectCryptopp\x64\Debug\ProjectCryptopp.exe (process 21732) exited with code 0.
To automatically close the console when debugging stops, enable Tools->Options->Debugging->Automatically close the console when debugging stops.
Press any key to close this window . . .|
```

Wnioski – Crypto++



Bibliografia

- **Crypto++[®] Library 8.7**, <https://www.cryptopp.com/>
- **Compiling Crypto++ in Microsoft Visual Studio 2019/2017 (with Cryptopp-PEM)** <https://www.youtube.com/watch?v=5XE4zEN-WKg>
- The C++ Programming Language, Special Edition - The C++ manual from the original designer of C++, Bjarne Stroustrup. This edition is up to date with the ISO/ANSI C++ Standard.