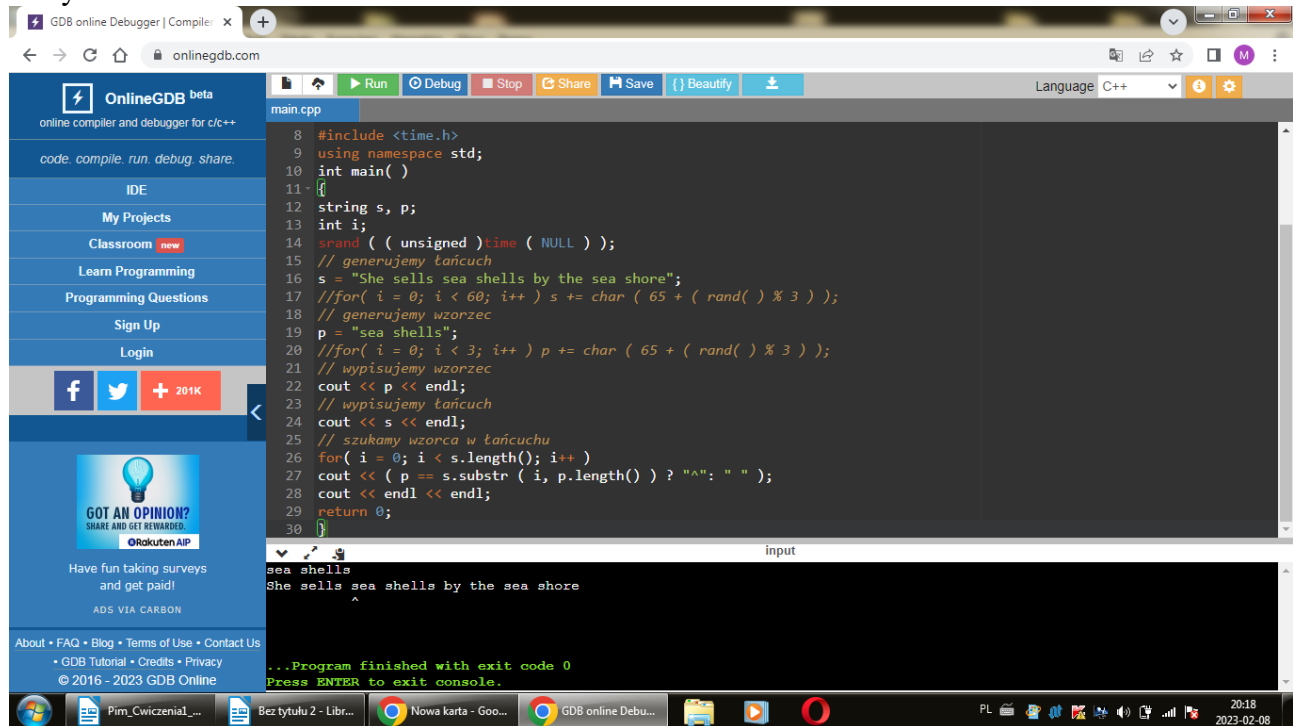


# Rozdział 1

## Zadanie 1.

### 1. Wyszukiwanie naiwne



The screenshot shows the OnlineGDB web IDE interface. The main editor displays a C++ program for a naive string search algorithm. The program generates a random string 's' and a pattern 'p', then searches for 'p' in 's'. The output shows the pattern found at index 0.

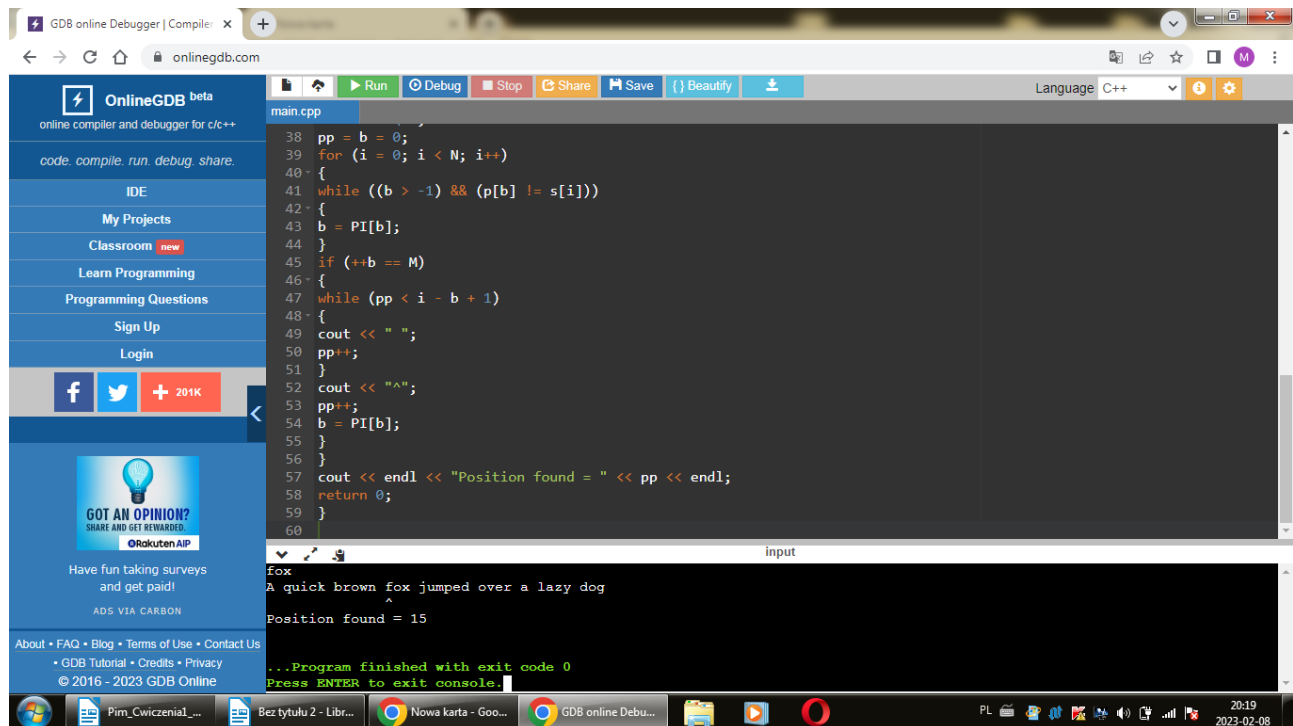
```
8 #include <time.h>
9 using namespace std;
10 int main( )
11 {
12     string s, p;
13     int i;
14     srand ( ( unsigned )time ( NULL ) );
15     // generujemy łańcuch
16     s = "She sells sea shells by the sea shore";
17     //for( i = 0; i < 60; i++ ) s += char ( 65 + ( rand( ) % 3 ) );
18     // generujemy wzorzec
19     p = "sea shells";
20     //for( i = 0; i < 3; i++ ) p += char ( 65 + ( rand( ) % 3 ) );
21     // wypisujemy wzorzec
22     cout << p << endl;
23     // wypisujemy łańcuch
24     cout << s << endl;
25     // szukamy wzorca w łańcuchu
26     for( i = 0; i < s.length(); i++ )
27         cout << ( p == s.substr ( i, p.length() ) ? "A": " " );
28     cout << endl << endl;
29     return 0;
30 }
```

The output console shows the pattern "sea shells" and the string "She sells sea shells by the sea shore". The program finished with exit code 0.

## Kod

```
// Algorytm WVN
// Data: 29.05.2008
// (C)2020 mgr Jerzy Wałaszek
//-----
#include <iostream>
#include <string>
#include <cstdlib>
#include <time.h>
using namespace std;
int main( )
{
    string s, p;
    int i;
    srand ( ( unsigned )time ( NULL ) );
    // generujemy łańcuch
    s = "She sells sea shells by the sea shore";
    //for( i = 0; i < 60; i++ ) s += char ( 65 + ( rand( ) % 3 ) );
    // generujemy wzorzec
    p = "sea shells";
    //for( i = 0; i < 3; i++ ) p += char ( 65 + ( rand( ) % 3 ) );
    // wypisujemy wzorzec
    cout << p << endl;
    // wypisujemy łańcuch
    cout << s << endl;
    // szukamy wzorca w łańcuchu
    for( i = 0; i < s.length(); i++ )
        cout << ( p == s.substr ( i, p.length() ) ? "A": " " );
    cout << endl << endl;
    return 0;
}
```

## 2. Wyszukiwanie Morrisa-Pratta



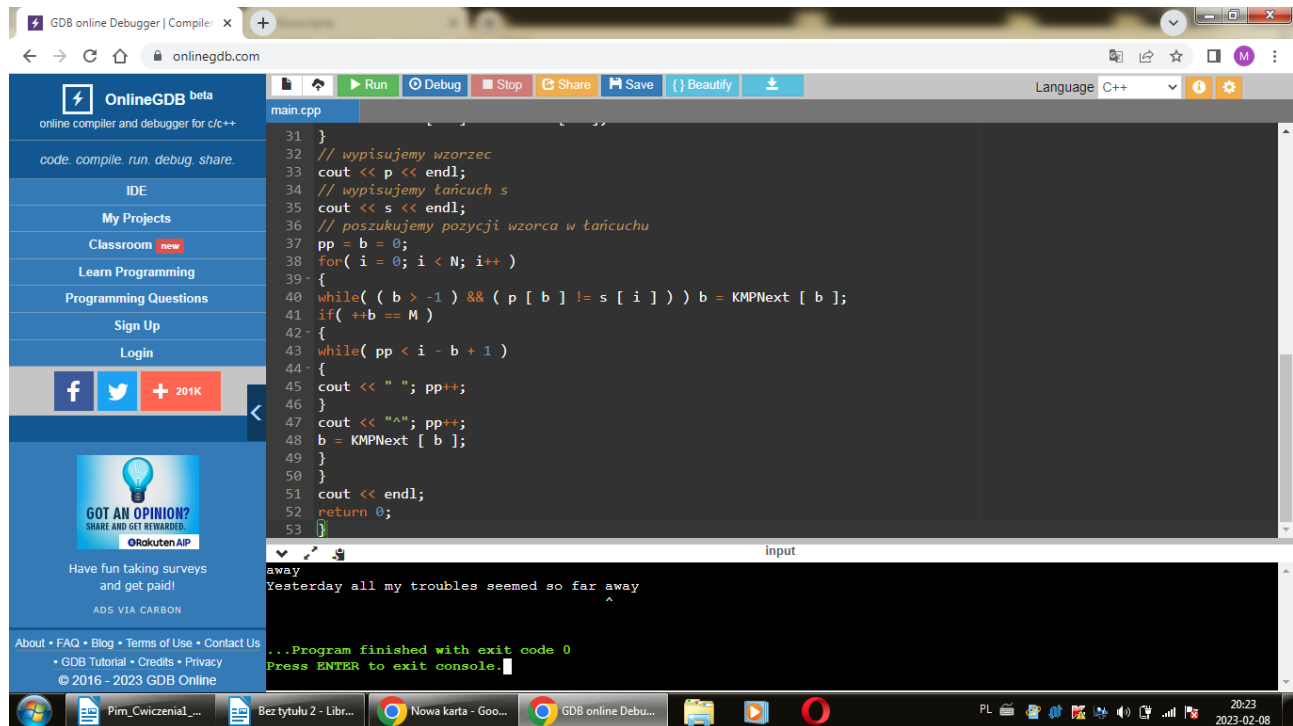
The screenshot shows the OnlineGDB web interface. On the left is a sidebar with navigation links like 'IDE', 'My Projects', 'Classroom', 'Learn Programming', 'Programming Questions', 'Sign Up', and 'Login'. The main area displays a C++ program in a dark-themed editor. The program implements the Morris-Pratt algorithm to find the position of a pattern 'fox' in a string 'A quick brown fox jumped over a lazy dog'. The output in the console shows 'Position found = 15' and '...Program finished with exit code 0'. The browser's address bar shows 'onlinegdb.com'.

```
38 pp = b = 0;
39 for (i = 0; i < N; i++)
40 {
41     while ((b > -1) && (p[b] != s[i]))
42     {
43         b = PI[b];
44     }
45     if (++b == M)
46     {
47         while (pp < i - b + 1)
48         {
49             cout << " ";
50             pp++;
51         }
52         cout << "A";
53         pp++;
54         b = PI[b];
55     }
56     cout << endl << "Position found = " << pp << endl;
57     return 0;
58 }
59 }
```

### Kod

```
// Wyszukiwanie wzorca algorytmem MP
// Data: 3.06.2008
// (C)2020 mgr Jerzy Wałaszek
// -----
#include <iostream>
#include <string>
#include <cstdlib>
#include <time.h>
using namespace std;
int main()
{
    string s, p;
    // generujemy łańcuch s
    s = "A quick brown fox jumped over a lazy dog";
    const int N = s.length();
    // generujemy wzorec
    p = "fox";
    const int M = p.length();
    int i, b, pp;
    int* PI = new int[M + 1];
    srand((unsigned)time(NULL));
    // dla wzorca obliczamy tablicę PI [ ]
    PI[0] = b = -1;
    for (i = 1; i <= M; i++)
    {
        while ((b > -1) && (p[b] != p[i - 1]))
        {
            b = PI[b];
        }
        PI[i] = ++b;
    }
    // wypisujemy wzorec
    cout << p << endl;
    // wypisujemy łańcuch s
    cout << s;
    // poszukujemy pozycji wzorca w łańcuchu
    cout << "n";
    pp = b = 0;
    for (i = 0; i < N; i++)
    {
        while ((b > -1) && (p[b] != s[i]))
        {
            b = PI[b];
        }
        if (++b == M)
        {
            while (pp < i - b + 1)
            {
                cout << " ";
                pp++;
            }
            cout << "A";
            pp++;
            b = PI[b];
        }
        cout << endl << "Position found = " << pp << endl;
        return 0;
    }
}
```

### 3. Knutha-Morrisa-Pratta



The screenshot shows the OnlineGDB web IDE interface. The left sidebar contains navigation links like 'IDE', 'My Projects', 'Classroom', 'Learn Programming', 'Programming Questions', 'Sign Up', and 'Login'. The main editor displays a C++ file named 'main.cpp' with the following code:

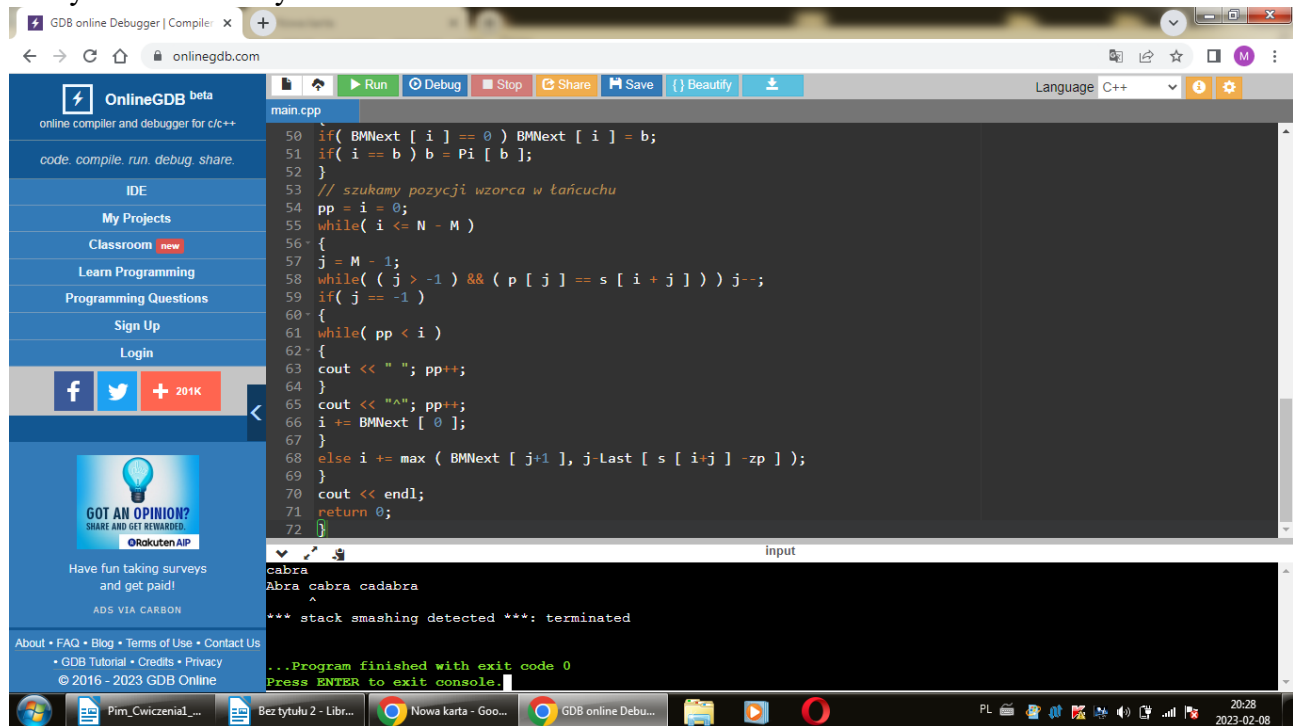
```
31 }
32 // wypisujemy wzorzec
33 cout << p << endl;
34 // wypisujemy łańcuch s
35 cout << s << endl;
36 // poszukujemy pozycji wzorca w łańcuchu
37 pp = b = 0;
38 for( i = 0; i < N; i++ )
39 {
40 while( ( b > -1 ) && ( p [ b ] != s [ i ] ) ) b = KMPNext [ b ];
41 if( ++b == M )
42 {
43 while( pp < i - b + 1 )
44 {
45 cout << " "; pp++;
46 }
47 cout << "^\n"; pp++;
48 b = KMPNext [ b ];
49 }
50 }
51 cout << endl;
52 return 0;
53 }
```

The output window shows the input string 'Yesterday all my troubles seemed so far away' and the message 'Program finished with exit code 0'. The taskbar at the bottom shows various open applications and the system clock indicating 20:23 on 2023-02-08.

### Kod

```
// Wyszukiwanie wzorca algorytmem KMP
// Data: 4.06.2008
// (C)2020 mgr Jerzy Wałaszek
//-----
#include <iostream>
#include <string>
#include <cstdlib>
#include <time.h>
using namespace std;
const int N = 44; // długość łańcucha s
const int M = 4; // długość wzorca p
int main()
{
    string s, p;
    int KMPNext [ M + 1 ], i, b, pp;
    srand ( ( unsigned )time ( NULL ) );
    // generujemy łańcuch s
    s = "Yesterday all my troubles seemed so far away";
    //for( i = 0; i < N; i++ ) s += 65 + rand() % 2;
    // generujemy wzorzec
    p = "away";
    //for( i = 0; i < M; i++ ) p += 65 + rand() % 2;
    // dla wzorca obliczamy tablicę Next [ ]
    KMPNext [ 0 ] = b = -1;
    for( i = 1; i <= M; i++ )
    {
        while( ( b > -1 ) && ( p [ b ] != p [ i - 1 ] ) ) b = KMPNext [ b ];
        ++b;
    }
    if( ( i == M ) || ( p [ i ] != p [ b ] ) ) KMPNext [ i ] = b;
    else KMPNext [ i ] = KMPNext [ b ];
    }
    // wypisujemy wzorzec
    cout << p << endl;
    // wypisujemy łańcuch s
    cout << s << endl;
    // poszukujemy pozycji wzorca w łańcuchu
    pp = b = 0;
    for( i = 0; i < N; i++ )
    {
        while( ( b > -1 ) && ( p [ b ] != s [ i ] ) ) b = KMPNext [ b ];
        if( ++b == M )
        {
            while( pp < i - b + 1 )
            {
                cout << " "; pp++;
            }
            cout << "^\n"; pp++;
            b = KMPNext [ b ];
        }
    }
    cout << endl;
    return 0;
}
```

## 4. Wyszukiwanie Boyera-Moora



The screenshot shows the OnlineGDB web interface. On the left is a sidebar with navigation links: IDE, My Projects, Classroom, Learn Programming, Programming Questions, Sign Up, and Login. Below these are social media icons for Facebook, Twitter, and a +201K badge. A banner for 'GOT AN OPINION?' is also visible. The main area displays a C++ program in main.cpp. The code implements the Boyer-Moore search algorithm, including the calculation of the last occurrence table (Last) and the bad character shift table (BMNext). The program searches for the pattern 'cabra' in the text 'Abra cabra cadabra'. The console output shows the search process, including the detection of a stack smashing error and the termination of the program.

```
main.cpp
50 if( BMNext [ i ] == 0 ) BMNext [ i ] = b;
51 if( i == b ) b = Pi [ b ];
52 }
53 // szukamy pozycji wzorca w łańcuchu
54 pp = i = 0;
55 while( i <= N - M )
56 {
57     j = M - 1;
58     while( ( j > -1 ) && ( p [ j ] == s [ i + j ] ) ) j--;
59     if( j == -1 )
60     {
61         while( pp < i )
62         {
63             cout << " "; pp++;
64         }
65         cout << "A"; pp++;
66         i += BMNext [ 0 ];
67     }
68     else i += max ( BMNext [ j+1 ], j-Last [ s [ i+j ] -zp ] );
69 }
70 cout << endl;
71 return 0;
72 }
```

input

```
cabra
Abra cabra cadabra
^
*** stack smashing detected ***: terminated

...Program finished with exit code 0
Press ENTER to exit console.
```

### Kod

```
// Wyszukiwanie wzorca pełnym algorytmem BM
// Data: 10.06.2008
// (C)2020 mgr Jerzy Wałaszek
//-----
#include <iostream>
#include <string>
#include <cstdlib>
#include <time.h>
using namespace std;
const int N = 18; // długość łańcucha s
const int M = 5; // długość wzorca p
const int zp = 65; // kod pierwszego znaku alfabetu
const int zk = 97; // kod ostatniego znaku alfabetu
int main()
{
    string s, p;
    int Last [ zk - zp + 1 ], BMNext [ M + 1 ], Pi [ M + 1 ], b, i, j, pp;
    srand ( ( unsigned )time ( NULL ) );
    // generujemy łańcuch s
    s = "Abra cabra cadabra";
    //for( i = 0; i < N; i++ )
    // s += zp + rand() % ( zk - zp + 1 );
    // generujemy wzorzec p
    p = "cabra";
    //for( i = 0; i < M; i++ )
    // p += zp + rand() % ( zk - zp + 1 );
    // wypisujemy wzorzec
    cout << p << endl;
    // wypisujemy łańcuch
    cout << s << endl;
    // dla wzorca obliczamy tablicę Last []
    for( i = 0; i <= zk - zp; i++ ) Last [ i ] = -1;
    for( i = 0; i < M; i++ ) Last [ p [ i ] - zp ] = i;
    // Etap I obliczania tablicy BMNext []
    for( i = 0; i <= M; i++ ) BMNext [ i ] = 0;
    i = M; b = M + 1; Pi [ i ] = b;
    while( i > 0 )
    {
        while( ( b <= M ) && ( p [ i - 1 ] != p [ b - 1 ] ) )
        {
            if( BMNext [ b ] == 0 ) BMNext [ b ] = b - i;
            b = Pi [ b ];
        }
        Pi [ --i ] = --b;
    }
    // Etap II obliczania tablicy BMNext []
    b = Pi [ 0 ];
    for( i = 0; i <= M; i++ )
    {
        if( BMNext [ i ] == 0 ) BMNext [ i ] = b;
        if( i == b ) b = Pi [ b ];
    }
    // szukamy pozycji wzorca w łańcuchu
    pp = i = 0;
    while( i <= N - M )
    {
        j = M - 1;
        while( ( j > -1 ) && ( p [ j ] == s [ i + j ] ) ) j--;
        if( j == -1 )
        {
            while( pp < i )
            {
                cout << " "; pp++;
            }
            cout << "A"; pp++;
            i += BMNext [ 0 ];
        }
        else i += max ( BMNext [ j+1 ], j-Last [ s [ i+j ] -zp ] );
    }
    cout << endl;
    return 0;
}
```

## Zadanie 2

Przykład 1:

Algorytm: Naiwny

Tekst: She sells sea shells by the sea shore

wzorzec: sea shells

wynik z konsoli:

```
sea shells
She sells sea shells by the sea shore
      ^
```

Przykład 2:

Algorytm: Naiwny

Tekst: A quick brown fox jumped over a lazy dog

wzorzec: fox

wynik z konsoli:

```
fox
a quick brown fox
                ^
```

Przykład 3:

Algorytm: Naiwny

Tekst: Artur ma zielone buty

wzorzec: buty

wynik z konsoli:

```
buty
artur ma zielone buty
                ^
```

Przykład 4:

Algorytm: Naiwny

Tekst: to jest za duzo przykladow

wzorzec: za duzo

wynik z konsoli:

```
za duzo
to jest za duzo przykladow
      ^
```

Przykład 5:

Algorytm: Naiwny

Tekst: jasmína ma jasminowe spodnie

wzorzec: jasminowe

wynik z konsoli:

```
jasminowe
Jasmina ma jasminowe spodnie
      ^
```

Przykład 6:

Algorytm: BoyeraMoora

Tekst: Abra Cabra Cabadra

wzorzec: Cabra

wynik z konsoli:

```
cabra
Abra cabra Cabadra
      ^
```

Przykład 7:

Algorytm: BoyeraMoora

Tekst: To jest krotki tekst

wzorzec: krotki

wynik z konsoli:

```
krotki
To jest krotki tekst
      ^
```

Przykład 8:

Algorytm: BoyeraMoora

Tekst: W Wsb jest 10 kierunkow

wzorzec: jest

wynik z konsoli:

```
jest
W Wsb jest 10 kierunkow
      ^
```

Przykład 9:

Algorytm: BoyeraMoora

Tekst: Choinka nasz ma malo galezi

wzorzec: nasza

wynik z konsoli:

```
nasza
Choinka nasza ma malo galezi
      ^
```

Przykład 10:

Algorytm: BoyeraMoora

Tekst: w styczniu mamy ujemne temperatury

wzorzec: temperatury

wynik z konsoli:

```
temperatury
w styczniu mamy ujemne temperatury
                        ^
```

Przykład 11:  
Algorytm:MorrisaPratta  
Tekst: Mam Trzy Motory  
wzorzec: Trzy  
wynik z konsoli:

```
trzy
Mam trzy motory
  ^
Position found = 5
```

Przykład 12:  
Algorytm:MorrisaPratta  
Tekst: W domu mam tygrysa  
wzorzec: tygrysa  
wynik z konsoli:

```
tygrysa
W domu mam tygrysa
      ^
Position found = 12
```

Przykład 13:  
Algorytm:MorrisaPratta  
Tekst: gdzie masz wycieraczki  
wzorzec:masz  
wynik z konsoli:

```
masz
Gdzie masz Wycieraczki
      ^
Position found = 7
```

Przykład 14:  
Algorytm:MorrisaPratta  
Tekst: To jest projekt nr 1  
wzorzec: To  
wynik z konsoli:

```
To
To jest projekt nr 1
^
Position found = 1
```

Przykład 15:  
Algorytm:MorrisaPratta  
Tekst: jest -10 stopni  
wzorzec: -10  
wynik z konsoli:

```
-10
Jest -10 stopni
    ^
Position found = 6
```

Przykład 16:

Algorytm: KnuthaMorrisaPratta

Tekst: troubles seemed so far away

wzorzec: seemed

wynik z konsoli:

```
seemed
troubles seemed so far away
    ^
```

Przykład 17:

Algorytm: KnuthaMorrisaPratta

Tekst: kinga ma czerwone auto

wzorzec: ma

wynik z konsoli:

```
ma
kinga ma czerwone auto
    ^
```

Przykład 18:

Algorytm: KnuthaMorrisaPratta

Tekst: ide jutro do dentysty

wzorzec: ide

wynik z konsoli:

```
ide
ide jutro do dentysty
^
```

Przykład 19:

Algorytm: KnuthaMorrisaPratta

Tekst: Wizyta prezydenta w Polsce

wzorzec: prezydenta

wynik z konsoli:

```
prezydenta
wizyta prezydenta w Polsce
    ^
```

Przykład 20:

Algorytm: KnuthaMorrisaPratta

Tekst: rosol z kury bez gesi



wzorzec: gesi  
wynik z konsoli:

```
gesi  
rosol z kury bez gesi  
^
```

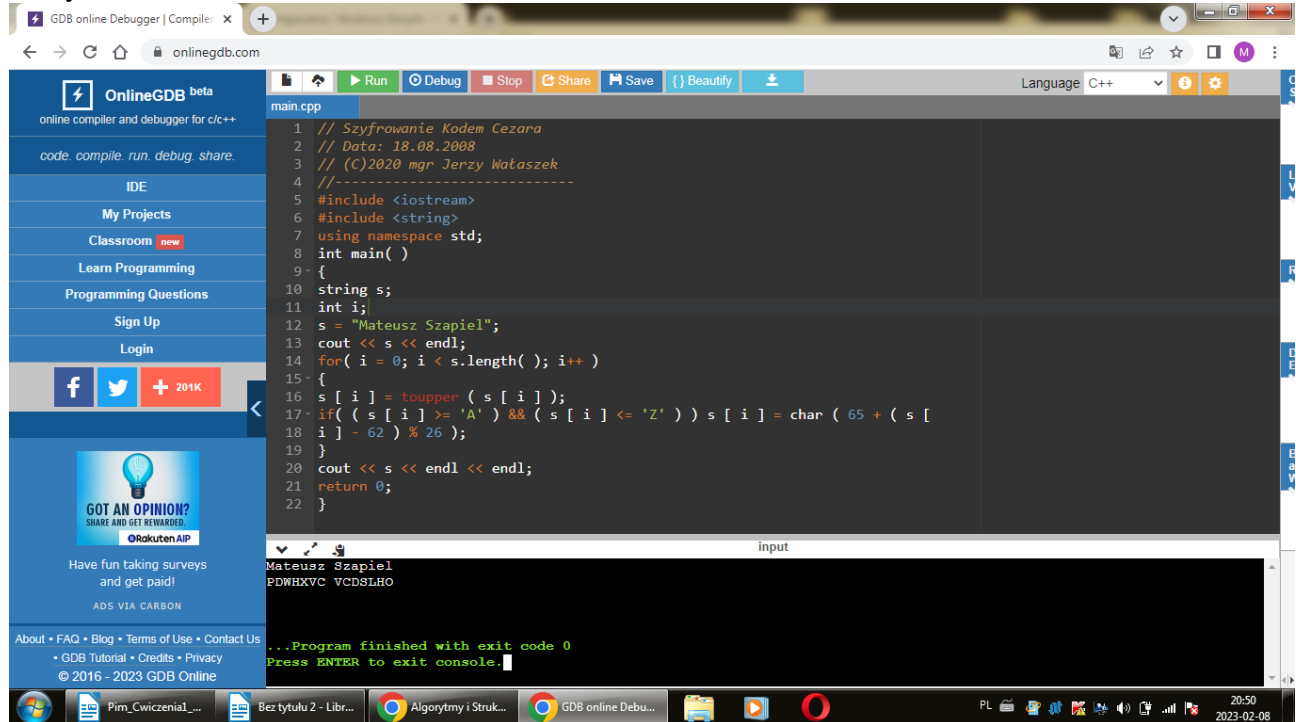
3. Przedstawić krótką dyskusję wybranego algorytmu z punktu widzenia złożoności obliczeniowej, pozytywnych oraz negatywnych cech.

Algorytm N– naiwny – ustawia okno o długości wzorca  $p$  na pierwszej pozycji w łańcuchu  $s$ . Następnie sprawdza, czy zawartość tego okna jest równa wzorcowi  $p$ . Jeśli tak, pozycja okna jest zwracana jako wynik, po czym okno przesuwa się o jedną pozycję w prawo i cała procedura powtarza się. Algorytm kończymy, gdy okno wyjdzie poza koniec łańcucha. Klasa pesymistycznej złożoności obliczeniowej algorytmu N jest równa  $O(n \times m)$ , gdzie  $n$  oznacza liczbę znaków tekstu, a  $m$  liczbę znaków wzorca. Jednakże w typowych warunkach algorytm pracuje w czasie  $O(n)$ , ponieważ zwykle wystarczy porównanie kilku początkowych znaków okna z wzorcem, aby stwierdzić, iż są one niezgodne. Z strony użytkownika ten algorytm według mnie jest najlepszy, ponieważ wymaga on najmniej informacji poprzez najłatwiej przyswoić o co w nim chodzi, a następnie możemy przejść do kolejnych algorytmów, które są nieco złożone

## Rozdział 2

Zadanie 2.1 i 2.2 jednocześnie zrobione Po 2 Przykłady na Algorytm i każdy zastosowany.

### 1.Szyfr Cezara



```
1 // Szyfrowanie Kodem Cezara
2 // Data: 18.08.2008
3 // (C)2020 mgr Jerzy Wataszek
4 //-----
5 #include <iostream>
6 #include <string>
7 using namespace std;
8 int main( )
9 {
10 string s;
11 int i;
12 s = "Mateusz Szapiel";
13 cout << s << endl;
14 for( i = 0; i < s.length( ); i++ )
15 {
16 s [ i ] = toupper ( s [ i ] );
17 if( ( s [ i ] >= 'A' ) && ( s [ i ] <= 'Z' ) ) s [ i ] = char ( 65 + ( s [
18 i ] - 62 ) % 26 );
19 }
20 cout << s << endl << endl;
21 return 0;
22 }
```

input

Mateusz Szapiel  
PDWHXVC VCDSLHO

...Program finished with exit code 0  
Press ENTER to exit console.

### Kod

```
// Szyfrowanie Kodem Cezara
// Data: 18.08.2008
// (C)2020 mgr Jerzy Wataszek
//-----
#include <iostream>
#include <string>
using namespace std;
int main( )
{
string s;
int i;
s = "Mateusz Szapiel";
cout << s << endl;
for( i = 0; i < s.length( ); i++ )
{
s [ i ] = toupper ( s [ i ] );
if( ( s [ i ] >= 'A' ) && ( s [ i ] <= 'Z' ) ) s [ i ] = char ( 65 + ( s [
i ] - 62 ) % 26 );
}
cout << s << endl << endl;
return 0;
}
```

### Przykład 1

Algorytm: Szyfr Cezara

Dane Wejsciowe: Mateusz Szapiel

Wyniki z Konsoli: Mateusz Szapiel

PDWHXVC VCDSLHO

### Przykład 2

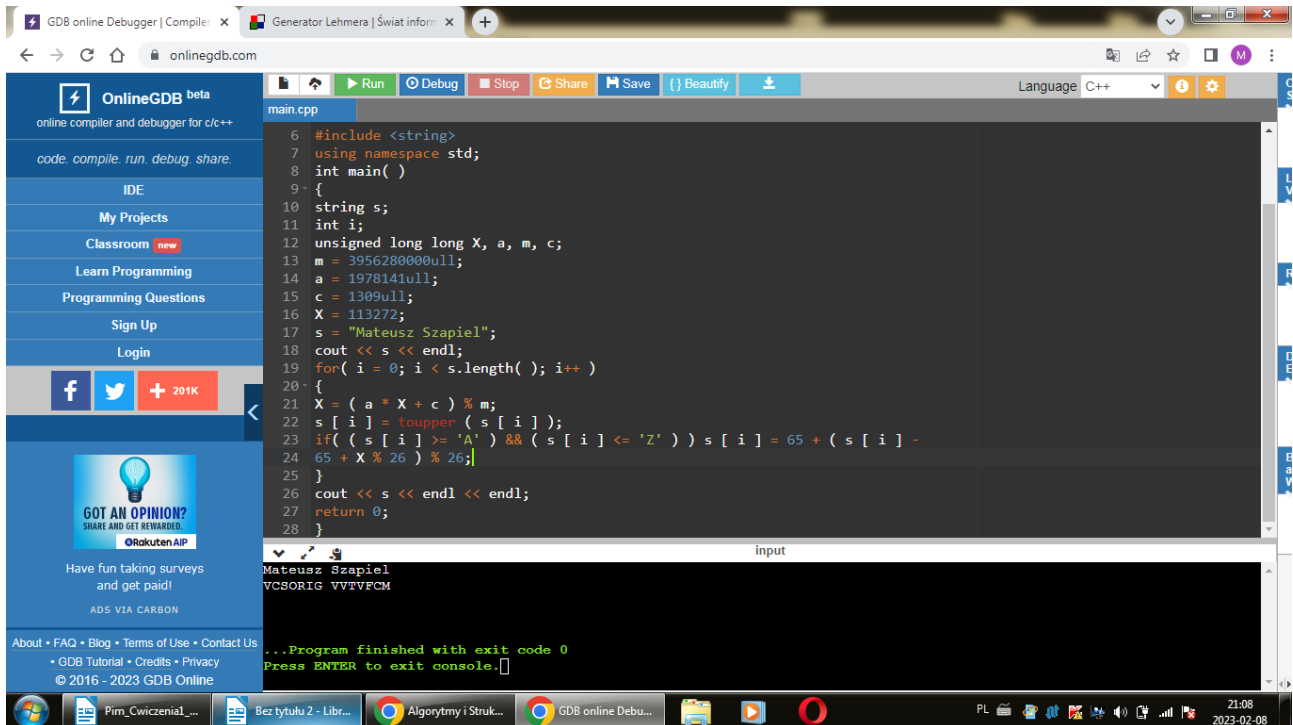
Algorytm: Szyfr Cezara

Dane Wejsciowe: słońce zabija mamuty w Australii

Wyniki z Konsoli: słońce zabija mamuty w Australii

VORQFH CDELM DPXWB Z DXVWUDOLL

## 2.Szyfr Z Pseudo Losowym Dostepem



### Kod

```
#include <string>
using namespace std;
int main()
{
    string s;
    int i;
    unsigned long long X, a, m, c;
    m = 3956280000ull;
    a = 1978141ull;
    c = 1309ull;
    X = 113272;
    s = "Mateusz Szapiel";
    cout << s << endl;
    for( i = 0; i < s.length(); i++)
    {
        X = (a * X + c) % m;
        s[i] = toupper(s[i]);
        if( (s[i] >= 'A') && (s[i] <= 'Z')) s[i] = 65 + (s[i] -
        65 + X % 26) % 26;
    }
    cout << s << endl << endl;
    return 0;
}
```

### Przykład 1

Algorytm: Szyfr Z Pseudolosowym Dostepem

Dane Wejsciove: Mateusz Szapiel

Wyniki z Konsoli: Mateusz Szapiel

VCSORIG VVTVFCM

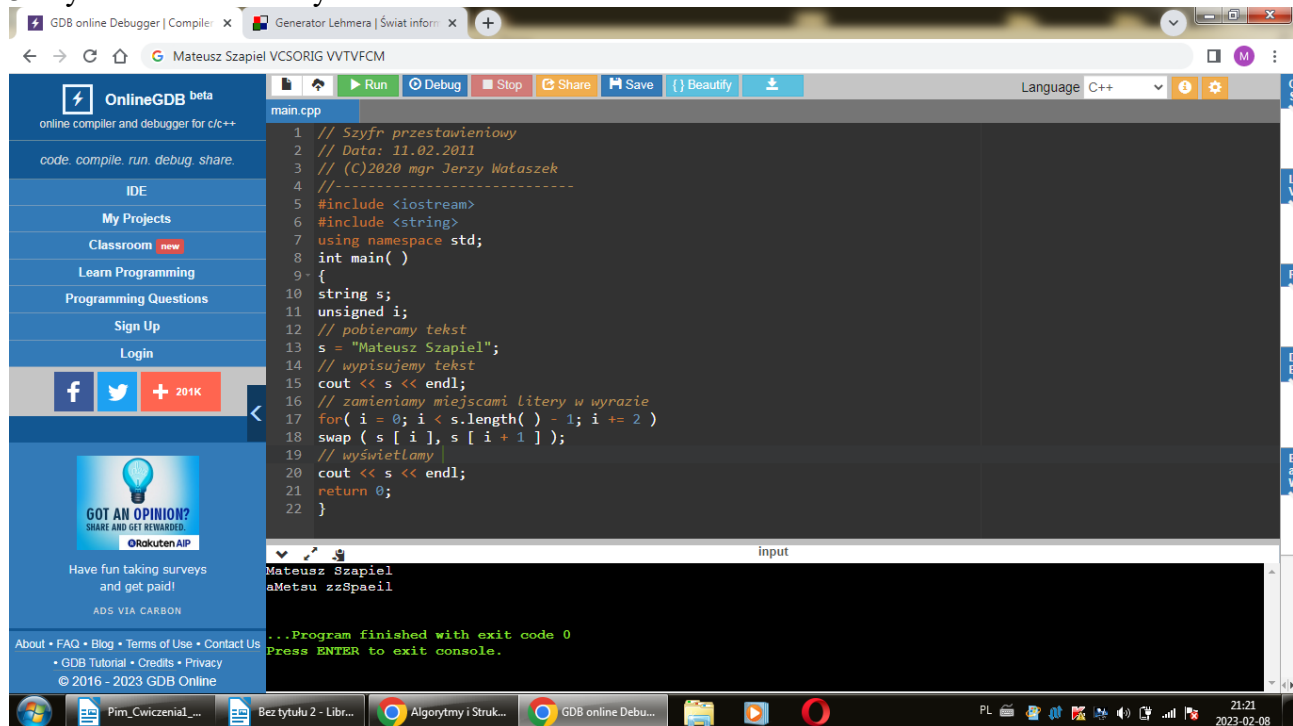
### Przykład 2

Algorytm: Szyfr Z pseudolosowym Dostepem

Dane Wejsciove: slonce zabija mamuty w Australii

Wyniki z Konsoli: słońce zabija mamuty w Australii  
BNNXZU DDXBPX NQPMCY O UXGGVTDPA

### 3.Szyfr Przestawieniowy



```
1 // Szyfr przestawieniowy
2 // Data: 11.02.2011
3 // (C)2020 mgr Jerzy Wałaszek
4 //-----
5 #include <iostream>
6 #include <string>
7 using namespace std;
8 int main( )
9 {
10 string s;
11 unsigned i;
12 // pobieramy tekst
13 s = "Mateusz Szapiel";
14 // wypisujemy tekst
15 cout << s << endl;
16 // zamieniamy miejscami litery w wyrazie
17 for( i = 0; i < s.length( ) - 1; i += 2 )
18 swap ( s [ i ], s [ i + 1 ] );
19 // wyświetlamy
20 cout << s << endl;
21 return 0;
22 }
```

input

Mateusz Szapiel  
aMetsu zzSpaeil

...Program finished with exit code 0  
Press ENTER to exit console.

#### Kod

```
// Szyfr przestawieniowy
// Data: 11.02.2011
// (C)2020 mgr Jerzy Wałaszek
//-----
#include <iostream>
#include <string>
using namespace std;
int main( )
{
string s;
unsigned i;
// pobieramy tekst
s = "Mateusz Szapiel";
// wypisujemy tekst
cout << s << endl;
// zamieniamy miejscami litery w wyrazie
for( i = 0; i < s.length( ) - 1; i += 2 )
swap ( s [ i ], s [ i + 1 ] );
// wyświetlamy
cout << s << endl;
return 0;
}
```

#### Przykład 1

Algorytm: Szyfr Przestawieniowy

Dane Wejściowe: Mateusz Szapiel

Wyniki z Konsoli: Mateusz Szapiel  
aMetsu zzSpaeil

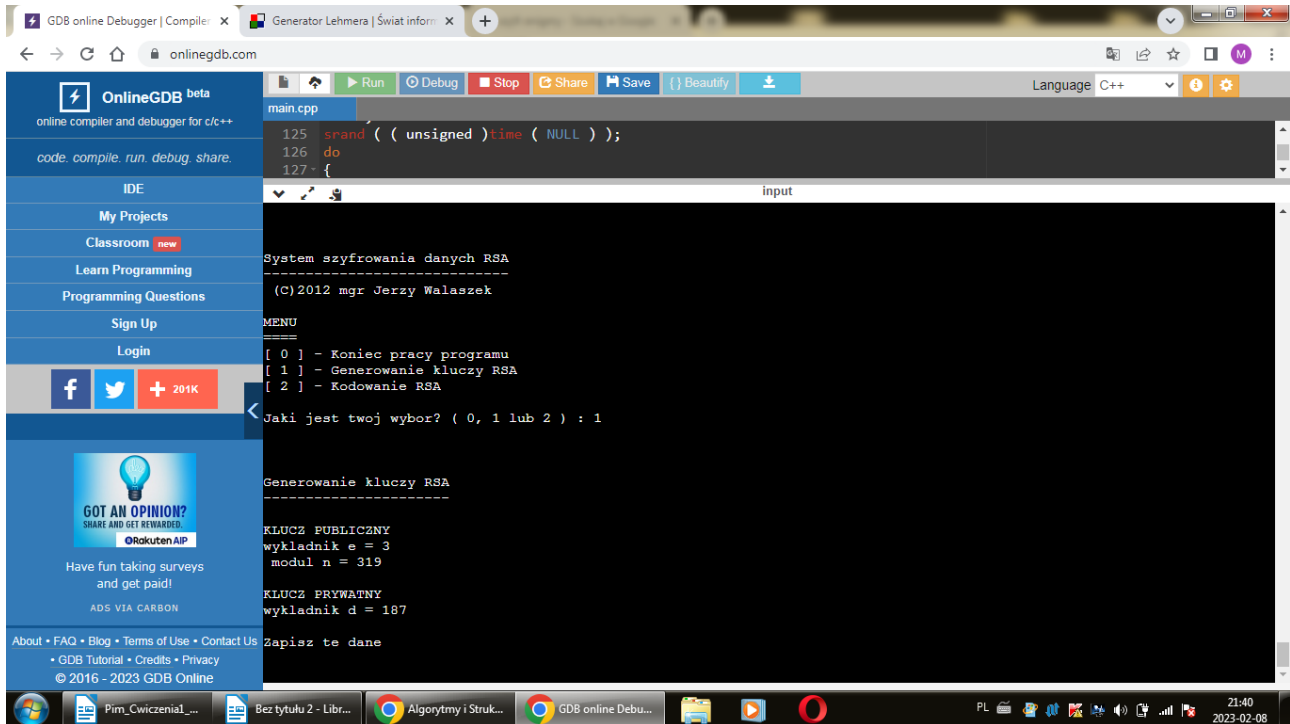
#### Przykład 2

Algorytm: Szyfr Przestawieniowy

Dane Wejściowe: słońce zabija mamuty w Australii

Wyniki z Konsoli: słońce zabija mamuty w Australii  
lsnoecz baji aamumytw A surtlaii

## 4.Szyfr RSA



The screenshot shows the OnlineGDB web interface. The left sidebar contains navigation links like 'IDE', 'My Projects', 'Classroom', 'Learn Programming', 'Programming Questions', 'Sign Up', and 'Login'. The main area displays a C++ program for RSA encryption. The code includes a menu with options: 0 - Koniec pracy programu, 1 - Generowanie kluczy RSA, and 2 - Kodowanie RSA. The console output shows the program's execution, including the generation of public and private keys. The public key is (e=3, n=319) and the private key is (d=187). The program prompts the user for a choice, and the user has entered '1'.

## Kod

```
/* ***** Przykładowa aplikacja obrazująca sposób działania ***** asymetrycznego systemu kodowania informacji RSA. ***** -----
(C)2020 mgr Jerzy Walaszek ***** I Liceum Ogólnokształcące ***** im. Kazimierza Brodzińskiego ***** w Tamowie *****
***** */ #include #include #include #include using namespace std; // Funkcja czeka na dowolny klawisz i czyści ekran //-----
void czekaj ( void ) { char c [ 1 ]; cout << "\nZapisz te dane\n\n"; cin.getline ( c, 1 ); cin.getline ( c, 1 ); for( int i = 1; i < 500; i++) cout << endl; } // Funkcja obliczająca NWD
dla dwóch liczb //----- int nwd ( int a, int b ) { int t; while ( b != 0 ) { t = b; b = a % b; a = t; } return a; } // Funkcja obliczania odwrotności modulo n //-----
int odwrotnosc_mod ( int a, int n ) { int a0, n0, p0, p1, q, r, t; p0 = 0; p1 = 1; a0 = a; n0 = n; q = n0 / a0; r = n0 % a0; while ( r > 0 ) { t = p0 - q * p1; if ( t >= 0 ) t = t % n; else t = n - ( -t ) % n;
p0 = p1; p1 = t; n0 = a0; a0 = r; q = n0 / a0; r = n0 % a0; } return p1; } // Procedura generowania kluczy RSA //----- void klucze_RSA ( ) { const int tp [ 10 ] = { 11, 13, 17, 19,
23, 29, 31, 37, 41, 43 }; int p, q, phi, n, e, d; cout << "Generowanie kluczy RSA\n\n"; // generujemy dwie różne, losowe liczby pierwsze do { p = tp [ rand ( ) % 10 ]; q = tp
[ rand ( ) % 10 ]; } while ( p == q ); phi = ( p - 1 ) * ( q - 1 ); // wyznaczamy wykładniki e i d for e = 3; nwd ( e, phi ) != 1; e += 2; d = odwrotnosc_mod ( e, phi ); // gotowe, wypisujemy klucze
cout << "KLUCZ PUBLICZNY\n" "wykładnik e = " << e << "\n" "modul n = " << n << "\n" "KLUCZ PRYWATNY\n" "wykładnik d = " << d << endl; czekaj ( ); } // Funkcja oblicza modulo potęgę
podanej liczby //----- int pot_mod ( int a, int w, int n ) { int pot, wyn, q; // wykładnik w rozkładamy na sumę potęg 2 // przy pomocy algorytmu Hornera. Dla reszty //
niezerowych tworzymy iloczyn potęg a modulo n. pot = a; wyn = 1; for( q = w; q > 0; q /= 2 ) { if ( q % 2 ) wyn = ( wyn * pot ) % n; pot = ( pot * pot ) % n; // kolejna potęga } return wyn; } //
Procedura kodowania danych RSA //----- void kodowanie_RSA ( ) { int e, n, t; cout << "Kodowanie danych RSA\n\n"; // Podaj wykładnik = "; cin >> e; cout
<< " Podaj modul = "; cin >> n; cout << "\n\n"; // Podaj kod RSA = "; cin >> t; cout << " Wynik kodowania = " << pot_mod ( t, e, n ) << endl; czekaj ( ); } //
***** Program główny ***** int main ( ) { int w; srand ( ( unsigned ) time ( NULL ) ); do { cout << "System szyfrowania danych RSA\n\n";
while ( w != 0 ) { return 0; } // Wynik: System
szyfrowania danych RSA -----
\n\n" " (C)2012 mgr Jerzy Walaszek\n\n" "MENU\n\n" " [ 0 ] - Koniec pracy programu\n" " [ 1 ] - Generowanie kluczy RSA\n" " [ 2 ] - Kodowanie RSA\n\n" " Jaki jest twój wybór?
( 0, 1 lub 2 ) : "; cin >> w; cout << "\n\n\n"; switch ( w ) { case 1 : klucze_RSA ( ); break; case 2 : kodowanie_RSA ( ); break; } cout << "\n\n\n"; } while ( w != 0 ); return 0; }
```

## Przykład 1

## Algorytm: Rsa

## Wyniki z Konsoli:

The screenshot shows the OnlineGDB interface with a C++ program for RSA key generation. The code in `main.cpp` includes a loop that prompts the user to choose between ending the program, generating RSA keys, or encoding data. The console output shows the program running and the user selecting option 1 to generate keys. The generated public key has `e = 3` and `n = 319`, while the private key has `d = 187`.

```
main.cpp
125 srand( ( unsigned )time ( NULL ) );
126 do
127 {
    System szyfrowania danych RSA
    -----
    (C)2012 mgr Jerzy Walaszek
    =====
    MENU
    =====
    [ 0 ] - Koniec pracy programu
    [ 1 ] - Generowanie kluczy RSA
    [ 2 ] - Kodowanie RSA

    Jaki jest twój wybór? ( 0, 1 lub 2 ) : 1

    Generowanie kluczy RSA
    -----
    KLUCZ PUBLICZNY
    wykładnik e = 3
    modul n = 319
    KLUCZ PRYWATNY
    wykładnik d = 187

    Zapisz te dane
```

Przykład 2:  
Algorytm:RSA  
wynik konsoli:

This screenshot shows the same OnlineGDB interface, but with different parameters for the RSA key generation. The console output shows the user selecting option 1 to generate keys. The generated public key has `e = 7` and `n = 533`, while the private key has `d = 343`.

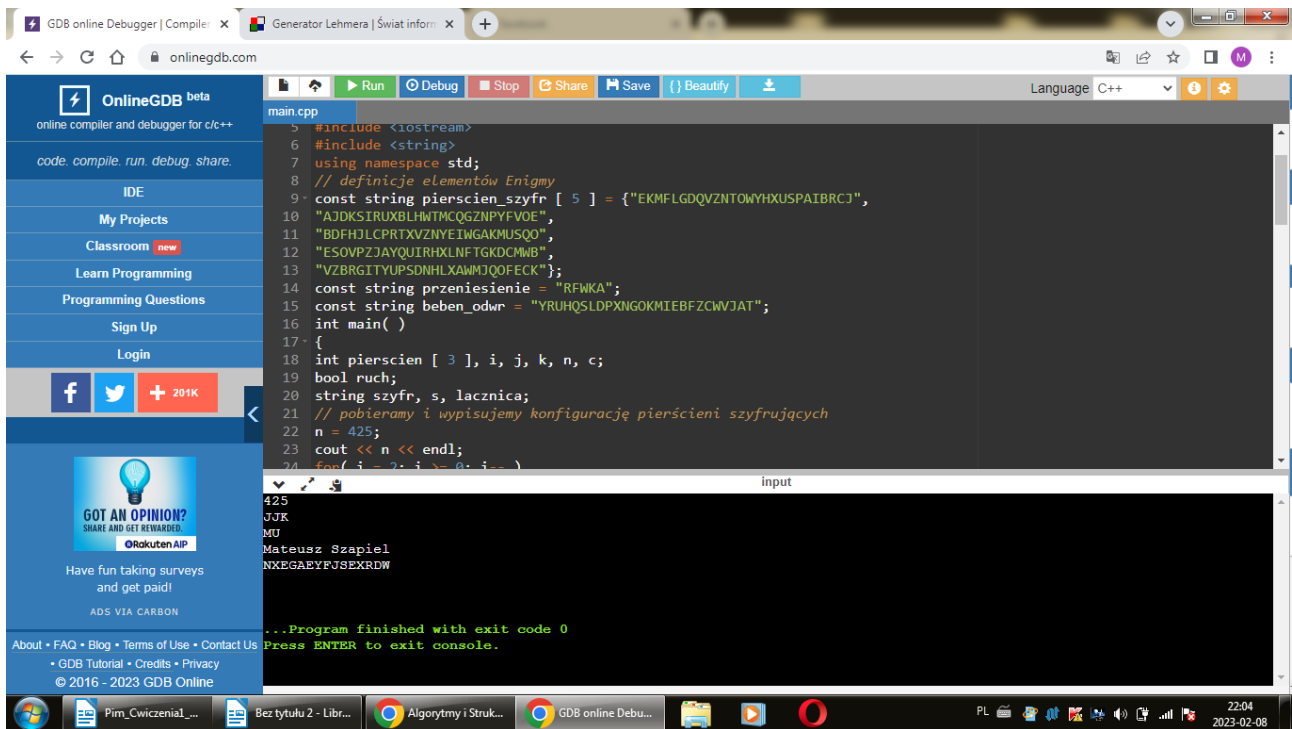
```
main.cpp
125 srand( ( unsigned )time ( NULL ) );
126 do
127 {
    System szyfrowania danych RSA
    -----
    (C)2012 mgr Jerzy Walaszek
    =====
    MENU
    =====
    [ 0 ] - Koniec pracy programu
    [ 1 ] - Generowanie kluczy RSA
    [ 2 ] - Kodowanie RSA

    Jaki jest twój wybór? ( 0, 1 lub 2 ) : 1

    Generowanie kluczy RSA
    -----
    KLUCZ PUBLICZNY
    wykładnik e = 7
    modul n = 533
    KLUCZ PRYWATNY
    wykładnik d = 343

    Zapisz te dane
```

## 5.Szyfr Enigmy



## Kod:

```
// Symulator Enigmy
// Data: 22.08.2008
// (C)2020 mgr Jerzy Wałaszek
//-----
#include <iostream>
#include <string>
using namespace std;
// definicje elementów Enigmy
const string pierścien_szyfr [ 5 ] = {"EKMFLGDQVZNTOWYHXUSPAIBRCJ",
"AJDKSIRUXBLHWTCQGZNPYFVOE",
"BDFHJLCPRTXVZNYEIWGAKMUSQO",
"ESOVJPZJAYQUIRHXNLFTGKDCMWB",
"VZBRGITYUPSDNHLXAWMJQOFECK"};
const string przeniesienie = "RFWKA";
const string beben_odwr = "YRUHQSLDPXNGOKMIEBFZCWVJAT";
int main()
{
    int pierścien [ 3 ], i, j, k, n, c;
    bool ruch;
    string szyfr, s, łącznica;
    // pobieramy i wypisujemy konfigurację pierścieni szyfrujących
    n = 235;
    cout << n << endl;
    for( i = 2; i >= 0; i-- )
    {
        pierścien [ i ] = ( n % 10 ) - 1; // numer pierścienia na i-tej pozycji
        n /= 10;
    }
    // pobieramy i wypisujemy położenia początkowe pierścieni
    szyfr = "KJJ";
    cout << szyfr << endl;
    for( i = 0; i < szyfr.length(); i++ ) szyfr [ i ] = toupper ( szyfr [ i ] );
    // pobieramy i wypisujemy stan łącznicy wtyczkowej
    s = "UM";
    cout << s << endl;
    for( i = 0; i < s.length(); i++ ) s [ i ] = toupper ( s [ i ] );
    łącznica = "ABCDEFGHIJKLMNOPQRSTUVWXYZ";
    for( i = 0; i < s.length() - 1; i += 2 )
    {
        łącznica [ s [ i ] - 65 ] = s [ i + 1 ];
        łącznica [ s [ i + 1 ] - 65 ] = s [ i ];
    }
    // pobieramy i wypisujemy szyfrogram
    s = "Mateusz Szapiel";
    cout << s << endl;
    for( i = 0; i < s.length(); i++ ) s [ i ] = toupper ( s [ i ] );
    // szyfrujemy/rozszyfrowujemy szyfrogram
    for( i = 0; i < s.length(); i++ )
    {
        // najpierw ruch pierścieni szyfrujących
        for( ruch = true, j = 2; ruch && ( j >= 0 ); j-- )
        {
            ruch = ( szyfr [ j ] == przeniesienie [ pierścien [ j ] ] );
            szyfr [ j ] = 65 + ( szyfr [ j ] - 64 ) % 26;
        }
        // pobieramy znak szyfrogramu
        c = s [ i ];
        // przechodzimy przez łącznicę wtyczkową
        c = łącznica [ c - 65 ];
        // przechodzimy przez pierścienie w kierunku do bębna odwracającego
        for( j = 2; j >= 0; j-- )
```

```

{
k = szyfr [ j ] - 65;
c = pierscien_szyfr [ pierscien [ j ] ] [ ( c - 65 + k ) % 26 ];
c = 65 + ( c - 39 - k ) % 26;
}
// przechodzimy przez bęben odwracający
c = beben_odwr [ c - 65 ];
// wracamy ścieżką powrotną
for( j = 0; j < 3; j++ )
{
k = szyfr [ j ] - 65;
c = 65 + ( c - 65 + k ) % 26;
for( n = 0; pierscien_szyfr [ pierscien [ j ] ] [ n ] != c; n++ );
c = 65 + ( 26 + n - k ) % 26;
}
// przechodzimy przez łącznicę wtyczkową
c = lacznica [ c - 65 ];
// uaktualniamy szyfrogram
s [ i ] = c;
}
// wyświetlamy szyfrogram
cout << s << endl << endl;
return 0;
}

```

## Przykład 1

Algorytm: Szyfr Enigmy

Dane Wejsciowe: Mateusz Szapiel

Wyniki z Konsoli:

235

JJK

MU

Mateusz Szapiel

NXEGAEYFJSEXRDW

## Przykład 2

Algorytm: Szyfr Enigmy

Dane Wejsciowe: słońce zabija mamuty w Australii

Wyniki z Konsoli:

434

DDU

JJ

słońce zabija mamuty w Australii

ADSKJDAKSDJJDJJ



### Zadanie 3

2.3 Przedstawić krótką dyskusję wybranego algorytmu z punktu widzenia złożoności obliczeniowej, pozytywnych oraz negatywnych cech.

Wybieram szyfr cezara ponieważ jest on łatwy w podstawieniu. Jego kod jest bardzo krótki, przejrzysty i ciężko się w nim pogubić osobie która ma swoje początki z algorytmami szyfrowania. Szyfr polega na zastępowaniu liter alfabetu A...Z literami leżącymi o trzy pozycje dalej w alfabecie. Ostatnie trzy znaki X, Y i Z nie posiadają następników w alfabecie przesuniętych o trzy pozycje. Dlatego umawiamy się, iż alfabet "zawija się" i za literką Z następuje znów litera A. Deszyfrowanie tekstu zaszyfrowanego kodem Cezara polega na wykonywaniu operacji odwrotnych. Każdą literę kodu zamieniamy na literę leżącą o trzy pozycje wcześniej w alfabecie. Plusami tego szyfru jest prosta konstrukcja oraz, że klucz jest łatwy do zapamiętania. Minusem tego szyfru jest fakt iż można go rozszyfrować w bardzo prosty sposób