ZAKRES MATERIAŁU

I. SZYFRY AFINICZNE

[Algorytm] (szyfr afiniczny)

Niech $P = C = Z_{26}$ i niech $K = \{(a,b) \in Z_{26} \times Z_{26} : NWD(a,26) = 1\}$

Dla $K = (a,b) \in K$ definiujemy funkcję szyfrującą i deszyfrującą

- $e_K(x) = (ax + b) \bmod 26$
- $d_K(y) = a^{-1}(y - b) \bmod 26$

gdzie $x, y \in Z_{26}$.

Szyfr afiniczny jest szczególnym przypadkiem szyfru podstawieniowego.

W przypadku szyfru afinicznego deszyfrowanie jest możliwe tylko wtedy, gdy
funkcja afiniczna $e_K(x) = (ax + b) \bmod 26$ $(a, b \in Z_{26})$ jest różnowartościowa, tzn. gdy
dla dowolnego $y \in Z_{26}$ kongruencja $ax + b \equiv y \pmod{26}$ ma dokładnie jedno rozwiązanie dla zmiennej x, a wnioskując dalej, gdy kongruencja $ax \equiv b \pmod{26}$ ma dokładnie jedno rozwiązanie $x \in Z_{26}$ dla każdego $b \in Z_{26}$.

[TW] Kongruencja $ax \equiv b \pmod{m}$ ma dokładnie jedno rozwiązanie $x \in Z_m$ dla każdego $b \in Z_m$ wtw $NWD(a,m)=1$.

Stąd w szyfrze afinicznym mamy możliwych 12*26=312 kluczy. Są to pary liczb (a, b), gdzie $a \in \{1, 3, 5, 7, 9, 11, 15, 17, 19, 21, 23, 25\}$, $b \in Z_{26}$.

[Idea] (algorytmu Euklidesa znajdowania NWD)
Mając do policzenia NWD(a,b) sprawdzamy, czy b=0. Jeśli tak jest, to NWD(a,b)=a, w przeciwnym przypadku wywołujemy rekurencyjnie algorytm dla liczb b i (a%b).
Np.: NWD(54,69)=NWD(69,54)=NWD(54,15)=NWD(15,9)=NWD(9,6)=NWD(6,3)=NWD(3,0)=3

[Zadanie 01] (NWD – algorytm Euklidesa)

Zaimplementuj algorytm wyznaczający NWD(a, b), gdzie $a, b \in Z_+ \cup \{0\}$. Skorzystaj z algorytmu Euklidesa.

[TEST] (odp. metoda klasy afiniczny)
Dane:
54 69 // a b
Wynik:
3

(A) ROZSZERZONY ALGORYTM EUKLIDESA ZNAJDOWANIA ELEMENTU ODWROTNEGO DO $a \in Z_m$

[DEF] (Element odwrotny do $a \in Z_m$)

Niech $a \in Z_m$. **Elementem odwrotnym** do a nazywamy element $a^{-1} \in Z_m$, taki że $aa^{-1} \equiv a^{-1}a \equiv 1 \pmod{m}$.

[Algorytm] (Rozszerzony algorytm Euklidesa znajdowania elementu odwrotnego do $a \in Z_m$)

I. Algorytm Euklidesa obliczania $NWD(r_0, r_1)$ $(r_0 > r_1)$

$r_0 = q_1 * r_1 + r_2$ $\quad 0 < r_2 < r_1$
$r_1 = q_2 * r_2 + r_3$ $\quad 0 < r_3 < r_2$
...
$r_{m-2} = q_{m-1} * r_{m-1} + r_m$ $\quad 0 < r_m < r_{m-1}$
$r_{m-1} = q_m * r_m$

Stąd $NWD(r_0, r_1) = r_m$,

II. Czy $a \in Z_m$ ma element odwrotny?

Zdefiniujmy ciąg liczb $w_0, w_1, w_2, ..., w_m$ następująco

$$\begin{cases} w_0 = 0 \\ w_1 = 1 \\ w_i = w_{i-2} - q_{i-1} w_{i-1} \pmod{r_0} \text{ dla } i > 1 \end{cases}$$

[TW] Jeśli $NWD(r_0, r_1)=1$, to $w_m = r_1^{-1} \pmod{r_0}$

[Zadanie 02]
Oblicz $29^{-1} \bmod 75$. Zastosuj rozszerzony algorytm Euklidesa.

$29^{-1} \bmod 75 = ?$

$75 = 2 * 29 + 17$		$w_0 = 0, \quad w_1 = 1$
$29 = 1 * 17 + 12$	$q_1 = 2$	$w_2 = (0 - 2 * 1) \bmod 75 = 73$
$17 = 1 * 12 + 5$	$q_2 = 1$	$w_3 = (1 - 1 * 73) \bmod 75 = 3$
$12 = 2 * 5 + 2$	$q_3 = 1$	$w_4 = (73 - 1 * 3) \bmod 75 = 70$
$5 = 2 * 2 + 1$	$q_4 = 2$	$w_5 = (3 - 2 * 70) \bmod 75 = 13$
$2 = 2 * 1$	$q_5 = 2$	$w_6 = (70 - 2 * 13) \bmod 75 = 44$

Stąd $NWD(75, 29) = 1$ Stąd $29^{-1} \bmod 75 = 44$, $(29 * 44) \bmod 75 = 1$

[Zadanie 03] (Element odwrotny)

Zaimplementuj algorytm znajdowania elementu odwrotnego do $a \in Z_m$.

[TEST] (odp. metoda klasy int26)
Dane:
13 73 // a m
Realizacja:

```
int26 a = new int26(13,73);
System.out.println(""+a.odwr());
```

Wynik:
45

[Zadanie 04] (Szyfr afiniczny)
Zaimplementuj algorytm realizujący szyfrowanie i deszyfrowanie szyfrem afinicznym. Sprawdź, czy podany przez użytkownika klucz jest prawidłowy.

[TEST] (odp. metody klasy afiniczny)
Dane:

```
"abcdefghijklmnopqrstuvwxyz",3,17,   // alfabet, a, b
kryptografia                         // tekst otwarty
```

Realizacja:

```
afiniczny sys01 = new afiniczny("abcdefghijklmnopqrstuvwxyz",3,17);
System.out.println(sys01);
System.out.println(sys01.szyfr("kryptografia"));
System.out.println(sys01.deszyfr(sys01.szyfr("kryptografia")));
Wyniki:
----------
szyfr afiniczny
alfabet=abcdefghijklmnopqrstuvwxyz
klucz (a,b)=(3,17)
vqlkwhjqrgpr
kryptografia
```

II. SZYFR HILLA

[Algorytm] (Szyfr Hilla)
Niech $n \in Z_+$ będzie ustalone oraz niech $P = C = (Z_{26})^n$ i
niech $K = \{$odwracalne macierze nxn nad $Z_{26}\}$.
Dla klucza K definiujemy funkcję szyfrującą i deszyfrującą

- $e_K(x) = xK$ i
- $d_K(y) = yK^{-1}$

wszystkie działania są wykonywane w Z_{26}.

Szyfr Hilla jest systemem polialfabetycznym.

(A) KONSTRUKCJA MACIERZY ODWROTNEJ DO MACIERZY $A \in Z_m^{nxn}$

[DEF] (wyznacznik macierzy – wyznaczanie za pomocą rozwinięcia Laplace'a)
Wyznacznik macierzy kwadratowej – funkcja, która każdej macierzy rzeczywistej (zespolonej) A przypisuje liczbę rzeczywistą (zespoloną) $\det A$. Funkcja ta jest określona rekurencyjnie

$$\det A = \begin{cases} a_{11} & \text{dla } n = 1 \\ \sum_{i=1}^{n} a_{ki} A_{ki} & \text{dla } n > 1 \end{cases} \quad \text{dla dowolnego } k \in \{1,\ldots,n\}$$

gdzie

- A_{ki} - **dopełnienie algebraiczne** elementu a_{ki} w macierzy A $\qquad A_{ki} = (-1)^{k+i} M_{ki}$
- M_{ki} - **minor elementu** a_{ki} **macierzy A** (tj. wyznacznik n-1-szego stopnia, powstały przez skreślenie k-tego wiersza i i-tej kolumny w macierz kwadratowej A)

[DEF] (Macierz dopełnień algebraicznych macierzy $A \in R^{nxn}$ (ozn. adjA)) $\qquad adjA = [A_{ij}]_{i,j=1,\ldots,n}$

[DEF] (Macierz dołączona)
Macierz adjA transponowana (ozn. $(adjA)^T$).

[DEF] (Macierz odwrotna do macierzy $A \in R^{nxn}$ **)**
Macierz kwadratowa (ozn. A^{-1}) spełniająca warunek $AA^{-1} = A^{-1}A = I_n$. Wówczas A nazywamy **macierzą odwracalną**.

$$A^{-1} = \begin{cases} (a_{11})^{-1} & \text{dla } n = 1 \\ (\det A)^{-1}(adjA)^T & \text{dla } n > 1 \end{cases}$$

Macierz kwadratowa $A \in R^{nxn}$ jest odwracalna wtw jest macierzą **nieosobliwą** (tj. gdy $\det A \neq 0$).

[DEF] (macierz osobliwa)

- macierz kwadratowa spełniająca warunek $\det A = 0$.
- **macierz nieosobliwa** – macierz, która nie jest osobliwa.

[Zadanie 05] Dla podanej macierzy A wyznacz
(a) wyznacznik macierzy detA,
(b) macierz dopełnień algebraicznych
(c) macierz dołączoną
(d) macierz odwrotną do A

$$A = \begin{bmatrix} 1 & 2 & 1 \\ 3 & 0 & 1 \\ -1 & -1 & 4 \end{bmatrix}$$

(a) $detA=3*(-M_{21})+0*M_{22}+1*(-M_{23})=3*(-9)+0*M_{22}+1*(-1)=-27-1=-28$,
(b) macierz dopełnień algebraicznych adjA
$A_{11}=(-1)^{1+1}*M_{11}=1$
$A_{12}=(-1)^{1+2}*M_{12}=-13$
$A_{13}=(-1)^{1+3}*M_{13}=-3$
$A_{21}=(-1)^{2+1}*M_{21}=-9$
$A_{22}=(-1)^{2+2}*M_{22}=5$
$A_{23}=(-1)^{2+3}*M_{23}=-1$
$A_{31}=(-1)^{3+1}*M_{31}=2$
$A_{32}=(-1)^{3+2}*M_{32}=-(-2)$
$A_{33}=(-1)^{3+3}*M_{33}=-6$

$$adjA = \begin{bmatrix} 1 & -13 & -3 \\ -9 & 5 & -1 \\ 2 & 2 & -6 \end{bmatrix}$$

(c) macierz dołączona $(adjA)^T$

$$(adjA)^T = \begin{bmatrix} 1 & -9 & 2 \\ -13 & 5 & 2 \\ -3 & -1 & -6 \end{bmatrix}$$

(d) macierz odwrotna A^{-1}

$$A^{-1} = (\det A)^{-1} * (adjA)^T = (-28)^{-1} * \begin{bmatrix} 1 & -9 & 2 \\ -13 & 5 & 2 \\ -3 & -1 & -6 \end{bmatrix} = -0.035714285 * \begin{bmatrix} 1 & -9 & 2 \\ -13 & 5 & 2 \\ -3 & -1 & -6 \end{bmatrix} =$$

KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI
Projekt współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego
UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

```
 |-0.035714, 0.321429,-0.071429 |
=| 0.464286,-0.178571,-0.071429 |
 | 0.107143, 0.035714, 0.214286 |
```

[!!!] Powyższą metodę wyznaczania macierzy odwrotnej można zastosować dla macierzy $A \in Z_m^{nxn}$.

Zadanie 06] Dla podanej macierzy A wyznacz w arytmetyce mod 41
(a) wyznacznik macierzy detA,
(b) macierz dopełnień algebraicznych,
(c) macierz dołączoną,
(d) macierz odwrotną do A.

$$A = \begin{bmatrix} 1 & 4 & 5 & 6 \\ 3 & 2 & 3 & 4 \\ 3 & 6 & 4 & 2 \\ 1 & 2 & 1 & 5 \end{bmatrix}$$

(a) detA=26,
$detA=1*(-M_{41})+2*M_{42}+1*(-M_{43})+5*M_{44}=(1*(-0)+2*38+1*(-35)+5*38) \bmod 41=26$

(b) $$adjA = \begin{bmatrix} 32 & 30 & 11 & 4 \\ 29 & 29 & 26 & 2 \\ 10 & 3 & 23 & 25 \\ 0 & 38 & 6 & 38 \end{bmatrix}$$

(c) $$(adjA)^T = \begin{bmatrix} 32 & 29 & 10 & 0 \\ 30 & 29 & 3 & 38 \\ 11 & 26 & 23 & 6 \\ 4 & 2 & 25 & 38 \end{bmatrix}$$

$A_{11}=(-1)^{11}*M_{11}=32$
$A_{12}=(-1)^{12}*M_{12}=-(-30)=30$
$A_{13}=(-1)^{13}*M_{13}=11$
$A_{14}=(-1)^{14}*M_{14}=-(-4)=4$
$A_{21}=(-1)^{21}*M_{21}=-(-29)=29$
$A_{22}=(-1)^{22}*M_{22}=29$
$A_{23}=(-1)^{23}*M_{23}=-(-26)=26$
$A_{24}=(-1)^{24}*M_{24}=2$
$A_{31}=(-1)^{31}*M_{31}=10$
$A_{32}=(-1)^{32}*M_{32}=-(-3)=3$
$A_{33}=(-1)^{33}*M_{33}=23$
$A_{34}=(-1)^{34}*M_{34}=-(-25)=25$
$A_{41}=(-1)^{41}*M_{41}=-0$
$A_{42}=(-1)^{42}*M_{42}=38$
$A_{43}=(-1)^{43}*M_{43}=-(-6)=6$
$A_{44}=(-1)^{44}*M_{44}=38$

(d) $$A^{-1} = (\det A)^{-1} * (adjA)^T = (26)^{-1} * \begin{bmatrix} 32 & 29 & 10 & 0 \\ 30 & 29 & 3 & 38 \\ 11 & 26 & 23 & 6 \\ 4 & 2 & 25 & 38 \end{bmatrix} = 30 * \begin{bmatrix} 32 & 29 & 10 & 0 \\ 30 & 29 & 3 & 38 \\ 11 & 26 & 23 & 6 \\ 4 & 2 & 25 & 38 \end{bmatrix} = \begin{bmatrix} 17 & 9 & 13 & 0 \\ 39 & 9 & 8 & 33 \\ 2 & 1 & 34 & 16 \\ 38 & 19 & 12 & 33 \end{bmatrix}$$

[Zadanie 07] (odwracanie macierzy)

Zaimplementuj algorytm wyznaczający macierz odwrotną do macierzy $A \in Z_m^{nxn}$. Skorzystaj z algorytmu wykorzystującego rozwinięcie Laplace'a.

[TEST] (odp. metody klasy mac26)
Dane:
26 // moduł
1, 3, 2, 3
2, 6, 3, 11
3, 4, 8, 10
1, 21, 12, 8
Realizacja

```
mac26 M26 = new mac26(4,26);
M26.read(M);        // wczytanie macierzy
System.out.println(M26);
System.out.println("det="+M26.det26().get_x());
System.out.println(M26.adj26());
System.out.println(M26.adj26().transp26());
System.out.println(M26.inverse26());
System.out.println(M26.inverse26().iloczM26(M26));
```

Wyniki:

```
1,3,2,3,            // macierz A
2,6,3,11,
3,4,8,20,
1,21,12,8,

det=23

10,4,5,23,          // adjA
8,4,15,18,
17,3,14,8,
24,5,1,21,

10,8,17,24,         // adjA transponowana
4,4,3,5,
5,15,14,1,
23,18,8,21,

14,6,3,18,          // macierz odwrotna do A
16,16,25,7,
7,21,4,17,
1,20,6,19,

1,0,0,0,            // A*A-1
0,1,0,0,
0,0,1,0,
0,0,0,1,
```

[Zadanie 08] (szyfr Hilla)

Zaimplementuj algorytm szyfrujący i deszyfrujący szyfrem Hilla. Sprawdź, czy podany przez użytkownika klucz jest prawidłowy.

[TEST] (odp. metody klasy Hill)
Dane:
"abcdefghijklmnopqrstuvwxyz" // alfabet
1 3 2 3 // klucz
2 6 3 11
3 4 8 20
1 21 12 8
Realizacja

```
mac26 M26 = new mac26(4,26);
M26.read(M);                        // wczytanie macierzy
Hill sys02 = new Hill("abcdefghijklmnopqrstuvwxyz",M26);
```


KAPITAŁ LUDZKI NARODOWA STRATEGIA SPÓJNOŚCI
Projekt współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego
UNIA EUROPEJSKA EUROPEJSKI FUNDUSZ SPOŁECZNY

```
System.out.println(sys02);
System.out.println(sys02.szyfr("kryptografia"));
System.out.println(sys02.deszyfr(sys02.szyfr("kryptografia")));
```

Wyniki:

```
----------
szyfr Hilla
alfabet=abcdefghijklmnopqrstuvwxyz
klucz K=
1,3,2,3,
2,6,3,11,
3,4,8,20,
1,21,12,8,

macierz odwrotna
14,6,3,18,
16,16,25,7,
7,21,4,17,
1,20,6,19,

bxblecuzikbh
kryptografia
```

[Zadanie 09] Zaimplementuj klasę

(a) macierz kwadratowa mac (z podstawowymi operacjami macierzowymi)

(b) macierz kwadratowa mac26 (z podstawowymi operacjami macierzowymi w arytmetyce modulo m)

```
class mac{//klasa wybranych operacji dla macierzy kwadratowych
public double[][] A;
private int n;

public mac(int n){…}
public void read(double [][]b){…}
public String toString(){…}
//zwraca kopię macierzy this.A
public mac copy(){…}
//zwraca wyznacznik macierzy this.A
public double det(){…}
//metoda pomocnicza do det()
private double _det(int n1, int w, int[] WK){…}
//zwraca macierz bez i1-tego wiersza i j1-tej kolumny
public mac Aij(int i1, int j1){…}
//zwraca macierz dopełnień algebraicznych dla this.A
public mac adj(){…}
//zwraca transpozycję macierzy this.A
public mac transp(){…}
//zwraca iloczyn macierzy this.A przez liczbę k
public mac iloczk(double k){…}
//zwraca iloczyn macierzy this.A i M
public mac iloczM(mac M){…}
//zwraca iloczyn wektora W przez macierz this.A
public double[] iloczW(double[] W){…}
//zwraca macierz odwrotną do this.A
public mac diverse(){…}
}
//-------------------------------------------------------------
```


KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI
UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY
Projekt współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego