Øving K14

Oppgave 1

Regn ut 232 + 22 · 77 – 18^2 (mod 8)

In [159]:
```
(232 + 22 * 77 - 18^2) % 12
```

Out[159]: 2

In [160]:
```
n = 232 % 8 + (22 % 8) * (77 % 8) - (18 % 12)^2
print(f'{n} % 12 ≡ {n % 12}')
```

```
26 % 12 ≡ 2
```

Oppgave 2

a) Skriv ut multiplikasjonstbellen Z_{12}, uten å ta med 0 (mod 12)

In [161]:
```
import numpy as np

n = 12
A = np.zeros((n-1, n-1), dtype=int)

for i in range(n - 1):
    for j in range(n - 1):
        A[i][j] = ((i + 1) * (j + 1)) % 12

print(A)
```

```
[[ 1  2  3  4  5  6  7  8  9 10 11]
 [ 2  4  6  8 10  0  2  4  6  8 10]
 [ 3  6  9  0  3  6  9  0  3  6  9]
 [ 4  8  0  4  8  0  4  8  0  4  8]
 [ 5 10  3  8  1  6 11  4  9  2  7]
 [ 6  0  6  0  6  0  6  0  6  0  6]
 [ 7  2  9  4 11  6  1  8  3 10  5]
 [ 8  4  0  8  4  0  8  4  0  8  4]
 [ 9  6  3  0  9  6  3  0  9  6  3]
 [10  8  6  4  2  0 10  8  6  4  2]
 [11 10  9  8  7  6  5  4  3  2  1]]
```

b) Hvilke tall har multiplikative invers modulo 12?

In [162]:
```
for i in np.where(A == 1)[0]: print(i+1)
```

```
1
5
7
11
```

c) Forklar hvorfor en ikke kan ha 0 og 1 i samme rad eller kolonne i tabellen, eller, sagt på en annen måte, hvis a ikke har multiplikativ invers, så finnes det en b som ikke er null mod 12, slik at ab ≡ 0 (mod 12)

Ser på de to utfallene isolert sett:

1. Verdien 1 forekommer i rad/kolonne a. Da er $gcd(a, n) = 1$. Tallet a er relativt primisk med n.
2. Verdien 0 forekommer i rad/kolonne a. Da er $gcd(a, n) > 1$. Tallet a er ikke relativt primisk med n.

Dette er to gjensidig utelukkende utfall. Enten er a relativt primisk med n, eller så er a ikke relativt primisk med n. Det kan derfor ikke forekomme både et 0 og et 1 tall i samme rad/kolonne.

Oppgave 3

In [163]:
```
A = np.array([[2, -1], [5, 8]]);
detA = int(np.linalg.det(A))

print(f'{A}, det = {detA}')
```

```
[[ 2 -1]
 [ 5  8]], det = 21
```

a) Finn den inverse matrisen til A over Z_{10}

In [164]:
```
if np.gcd(detA, 10) == 1:
    print((np.linalg.inv(A) * detA) % 10)
else:
    print(f'gcd({detA}, 10) = {np.gcd(detA, 10)} ≠ 1 => ingen invers')
```

```
[[8. 1.]
 [5. 2.]]
```

b) Finn den inverse matrisen til A over Z_9

In [165]:
```
if np.gcd(detA, 9) == 1:
    print((np.linalg.inv(A) * detA) % 9)
else:
    print(f'gcd({detA}, 9) = {np.gcd(detA, 9)} ≠ 1 => ingen invers')
```

```
gcd(21, 9) = 3 ≠ 1 => ingen invers
```

Oppgave 4

a) Hvor mange forskjellige nøkler kan et (enkelt) substitusjonschiffer ha når vi opererer med et alfabet med 29 tegn?

Da kan man ha $29! \approx 10^{30}$ forskjellige nøkler

b) Et slikt substitusjonschiffer er ikke særlig trygt. Hvilke enkel grep kan Alice og Bob bruke for å gjøre det vanskeligere for Eva å dekode meldingene?

- Unnlate å bruke mellomrom. Da er det vanskeligere å se oppbyggingen i meldingen. Kan ikke se antall ord og lengde på ordene.
- Unngå å bruke gjentagene faser. (ref. Enigma m/ 'Heil Hitler', 'Værrapport')

c) Hvis vi lager en substitusjonchiffer for blokker med n tegn, hvor mange nøkler finnes da?

Da finnes det $n!$ forskjellige nøkler

Oppgave 5

Du har snappet om følgende melding:

YÆVFB VBVFR ÅVBV

Du vet at Alice og Bob bruker et k-skift-chiffer. Finn krypteringsnøkkelen og klarteksten! (Husk at mellomrom ikke er tatt med i teksten.)

In [166]:
```
ALPHABET = 'ABCDEFGHIJKLMNOPQRSTUVWXYZÆØÅ'

def shift(p, K):
    c = ''

    for char in p:
        index = (ALPHABET.index(char) + K) % len(ALPHABET)
        c += ALPHABET[index]

    return c
```

In [167]:
```
for k in range(len(ALPHABET)):
    shifted = shift('YÆVFBVBVFRÅVBV', k)
    print(f'{k}\t{shifted.lower()}')
```

```
0       yævfbvbvfråvbv
1       zøwgcwcwgsawcw
2       ææxhdxdxhtbxdx
3       øayieyeyiucyey
4       åbzjfzfzjvdzfz
5       acækgægækweægæ
6       bdølhøhølxføhø
7       ceåmiåiåmygåiå
8       dfanjajanzhaja
9       egbokbkboæibkb
10      fhcplclcpøjclc
11      gidqmdmdqåkdmd
12      hjerneneralene
13      ikfsofofsbmfof
14      jlgtpgpgtcngpg
15      kmhuqhqhudohqh
16      lnivririvepiri
17      mojwsjsjwfqjsj
18      npkxtktkxgrktk
19      oqlyululyhslul
20      prmzvmvmzitmvm
21      qsnæwnwnæjunwn
22      rtoøxoxoøkvoxo
23      supåypypålwpyp
24      tvqazqzqamxqzq
25      uwrbærærbnyrær
26      vxscøsøscozsøs
27      wytdåtåtdpætåt
28      xzueauaueqøuau
```

Nøkkel 12 gir dekryptert streng "Hjernen er alene"

Oppgave 6

Definer et blokk-chiffer med blokklengde b, og et alfabete med N tegn, som bruker samme prinsipp som skift-chifret.
a) Skriv opp en formell definisjon.

La $P = C = K = \{x \mid 0 \leq x < N\}$, hvor N er antall tegn. La b være blokklengden.

$x = x_1 x_2 \dots x_b$

$e_k(x) = (x + k)(mod N)$
$d_k(y) = (x - k)(mod N)$

b) Hvor mange forskjellige nøkler har chifret?

Det vil finnes N forskjellige nøkler.

Oppgave 7

a) Krypter teksten 'Nå er det snart helg' med nøkkelordet 'torsk'

In [169]:
```
ALPHABET = 'ABCDEFGHIJKLMNOPQRSTUVWXYZÆØÅ'

def encryptCharacter(char, keyChar):
    charIndex = ALPHABET.index(char)
    keyIndex  = ALPHABET.index(keyChar)

    return shift(ALPHABET, keyIndex)[charIndex]

def encrypt(p, key):
    c = ''

    p = p.replace(' ', '').upper()
    key = key.replace(' ', '').upper()

    for i in range(len(p)):
        c += encryptCharacter(p[i], key[i % len(key)])

    return c.upper()
```

In [170]:
```
p = 'Nå er det snart helg'
K = 'torsk'
c = encrypt(p, K)

print(f'p = {p}, K: {K} => c = {c}')
```

```
p = Nå er det snart helg, K: torsk => c = DNVGNXEGCKHEYWVZ
```

b) Dekrypter 'QZQOBVCAFFKSDC' med nøkkelordet 'brus'

In [171]:
```
ALPHABET = 'ABCDEFGHIJKLMNOPQRSTUVWXYZÆØÅ'

def decryptCharacter(char, keyChar):
    charIndex = ALPHABET.index(char)
    keyIndex  = ALPHABET.index(keyChar)

    return shift(ALPHABET, -keyIndex)[charIndex]

def decrypt(c, key):
    p = ''

    c = c.replace(' ', '').upper()
    key = key.replace(' ', '').upper()

    for i in range(len(c)):
        p += decryptCharacter(c[i], key[i % len(key)])

    return p.lower()
```

In [172]:
```
c = 'QZQOBVCAFFKSDC'
K = 'brus'
p = decrypt(c, K)

print(f'c = {c}, K = {K} => p = {p}')
```

```
c = QZQOBVCAFFKSDC, K = brus => p = pizzaellertaco
```

c) Hvis m = 5 (se definisjonen), hvor mange nøkler finnes?

Har $N = 29$ tegn i alfabetet. Da blir antall nøkler 29^5

Oppgave 8

a) Finn K^{-1} over Z_{29}

In [173]:
```
K = np.array([[11, 8], [3, 7]])
detK = np.linalg.det(K)
invK = (np.linalg.inv(K) * detK) % len(ALPHABET)

print(invK)
```

```
[[ 7. 21.]
 [26. 11.]]
```

b) Krypter teksten “prim” med K som nøkkel i Hill-chifret.

In [174]:
```
# Converts string to matrix
def stringToMatrix(string):
    matrix = []

    for char in string:
        matrix.append(ALPHABET.index(char))

    return np.array([ matrix ])

# Converts matrix to string
def matrixToString(matrix):
    string = ''

    for num in matrix[0]:
        string += ALPHABET[int(num)]

    return string
```

In [175]:
```
ALPHABET = 'ABCDEFGHIJKLMNOPQRSTUVWXYZÆØÅ'

def encrypt(p, K):
    c = ''

    # Splits message into blocks of two
    p = [p.upper()[i:i+2] for i in range(0, len(p), 2)]

    for char in p:
        x = stringToMatrix(char)
        matrix = np.matmul(x, K) % len(ALPHABET)
        c += matrixToString(matrix)

    return c.upper()
```

In [176]:
```
encrypt('prim', K)
```

Out[176]: 'NHID'

b) Dekrypter meldingen TOYYSN

In [177]:
```
def modInverse(a, m) :
    a = a % m;
    for x in range(1, m) :
        if ((a * x) % m == 1) :
            return x
    return 1
```

In [178]:
```
ALPHABET = 'ABCDEFGHIJKLMNOPQRSTUVWXYZÆØÅ'

def decrypt(c, invK):
    p = ''

    # Splits message into blocks of two
    c = [c.upper()[i:i+2] for i in range(0, len(c), 2)]

    for char in c:
        y = stringToMatrix(char)
        inv = modInverse(int(detK), 29)
        matrix = np.matmul(y, (invK * inv) % 29) % len(ALPHABET)
        p += matrixToString(matrix)

    return p.lower()
```

In [179]:
```
decrypt('TOYYSN', invK)
```

Out[179]: 'fredag'

d) For en annen nøkkel med m = 2, så er meldingen EASY kryptert til IØÅY. Finn nøkkelen ut fra bare kjennskap til denne ene meldingen og dens kryptering. (Dette er et eksempel på kjent klartekst-angrep)

In [180]:
```
def findKey(p, c, N):
    for i in range(N):
        for j in range(N):
            for k in range(N):
                for l in range(N):
                    K = np.array([[i, j], [k, l]])
                    if encrypt(p, K) == c:
                        return K
    return 1
```

In [181]:
```
p = 'easy'
c = 'IØÅY'

N = 20

K = findKey(p, c, N)
print(K)
```

```
[[ 2 14]
 [19  5]]
```