Classical Ciphers

Part 1

Content

Content
Caesar and Rot13 Ciphers
Affine Cipher
Permutation Cipher

Caesar and Rot13 Ciphers

- Encryption: shift the letters by k letters to the **right**.
- Decryption: shift the letters by k letters to the **left**.

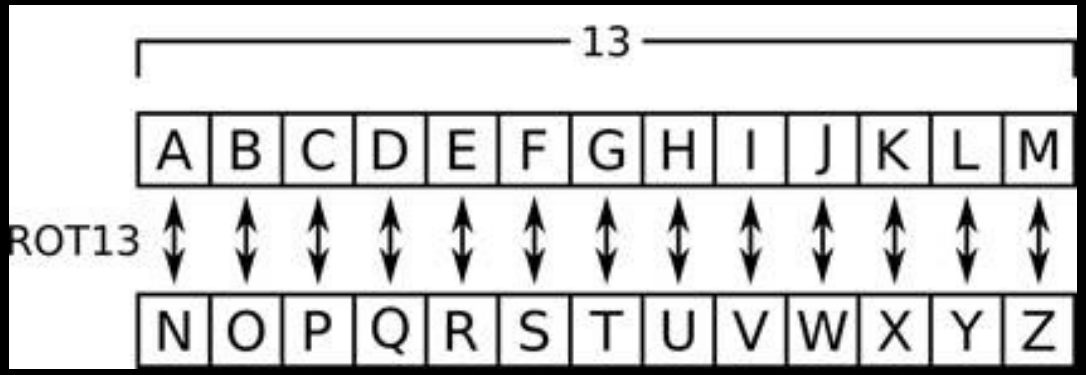
- For example, if $k = 5$

plaintext:	`abcdefghijklmnopqrstuvwxyz`
ciphertext:	`FGHIJKLMNOPQRSTUVWXYZABCDE`

Caesar and Rot13 Ciphers

- Caeser cipher using modular arithmetic:
  - Encryption: $c_i = (p_i + k)\%26$
  - Decryption: $p_i = (c_i - k)\%26$

- If the shift amount $k = 13$, the cipher is called **Rot13** cipher.
  - This shift has a special name because 13 is half of 26.



Caesar and Rot13 Ciphers

TASK

- Implement a custom modulo function to map a value x in the range n_1 and n_2 such that $n_1 < n_2$

Algorithm 38: Custom Modulo Algorithm

Input: $x, n1, n2$
Output: $result$

$difference \leftarrow x - n1;$
$modulus\_result \leftarrow difference \mod (n2 - n1 + 1);$
$result \leftarrow modulus\_result + n1;$
return $result;$

Caesar and Rot13 Ciphers

TASK

- Implement the Caesar cipher using the custom modulo function
- Implement the Rot13 function using the Caesar cipher as a helper function
- Brute force the following Caesar cipher ciphertext
  “WXF HXD LJW LAJLT VN!”

Algorithm 39: Caesar Cipher Algorithm

Input: $text, shift$
Output: $ctxt$

```
ctxt ← "";
foreach s in text do
    if s ∈ [A : Z] then
        c ← s + shift;
        c ← custom_mod(c, A, Z);
        ctxt ← ctxt + c;
    end
    else
        ctxt ← ctxt + s;
    end
end
return ctxt;
```

Content

Content
Caesar and Rot13 Ciphers
Affine Cipher
Permutation Cipher

Affine Cipher

- It extends the Caesar cipher to include the use of modular multiplication.
  - Encryption: $c_i = (a \cdot p_i + b) \,\%\, 26$
  - Decryption: $p_i = a^{-1}(c_i - b) \,\%\, 26$

- The key is two values: the multiplier a and the shift b.
- Note that a must invertible, so $\gcd(a, 26)$ must be equal to one.
  - i.e., no common factors between a and 26
  - Ex: $\gcd(15, 26) = 1$
  - Ex: $\gcd(12, 26) = 2$

Affine Cipher

- How to find the inverse of a?

- We need to find a value x such that:

$$ax \equiv 1 \ (mod\ 26)$$

  - i.e., find a number x such that If you multiply x and a and reduce the result (mod 26), you will get the answer 1.

- Extended GCD computes the multiplicative inverse

Affine Cipher

- How decryption works?

$$
\begin{aligned}
D(E(x)) &= a^{-1}(E(x) - b) \bmod m \\
&= a^{-1}(((ax + b) \bmod m) - b) \bmod m \\
&= a^{-1}(ax + b - b) \bmod m \\
&= a^{-1}ax \bmod m \\
&= x \bmod m
\end{aligned}
$$

Affine Cipher

- GCD algorithm

Algorithm 35: Euclidean Algorithm for GCD

Input: a, b
Output: GCD of a and b
while $b \neq 0$ **do**
    $r \leftarrow a \mod b$;
    $a \leftarrow b$;
    $b \leftarrow r$;
end
return a

Affine Cipher

- Extended GCD algorithm

Algorithm 37: Iterative Extended Euclidean Algorithm

Input: a, b

Output: GCD of a and b, and coefficients x and y such that $ax + by = \gcd(a, b)$

Function `IterEGCD(`a, b`)`**:**

- $x_0 \leftarrow 1$, $y_0 \leftarrow 0$;
- $x_1 \leftarrow 0$, $y_1 \leftarrow 1$;
- **while** $b \neq 0$ **do**
  - $q \leftarrow \lfloor \frac{a}{b} \rfloor$;
  - $r \leftarrow a \mod b$;
  - $a \leftarrow b$;
  - $b \leftarrow r$;
  - $x \leftarrow x_0 - q \times x_1$;
  - $y \leftarrow y_0 - q \times y_1$;
  - $x_0 \leftarrow x_1$, $y_0 \leftarrow y_1$;
  - $x_1 \leftarrow x$, $y_1 \leftarrow y$;
- **end**
- **Return** a, x_0, y_0

Return `IterEGCD(`a, b`)`

Affine Cipher

- **Hands-on example**: encrypt the message “HELLO” using affine cipher. Assume $a = 5$ and $b = 4$.
- **Encryption steps:**
  - First check if a is invertible → $\gcd(a, 26) = 1$
  - Represent the plaintext as integer values
  - For each plaintext value, compute $c_i = (a \cdot p_i + b)\ \%\ 26$
- **Decryption steps:**
  - Compute a^{-1} → $egcd(a, 26)$.
  - Represent the ciphertext as integer values
  - For each ciphertext value, compute $p_i = a^{-1}(c_i\ - b)\ \%\ 26$

Affine Cipher

Encryption
$a = 5$ is invertible?
Encode the plaintext
Encrypt each value

Affine Cipher

Encryption
$a = 5$ is invertible?
Encode the plaintext
Encrypt each value

$\gcd(26, 5)$:
1) $a = 26, \quad b = 5$
2) $r := 26 \ \% \ 5 = 1$
3) $a = 5$
4) $b = 1$

---

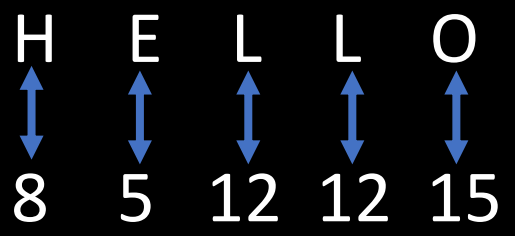
1) $r = 5 \ \% \ 1 = 0$
2) $a = 1$
3) $b = 0$

Since $\mathrm{b} = 0$, the algorithm stops
The result is $a = 1$

Since $\gcd(26, 5) = 1$.
$\therefore$ 5 is invertible.

Affine Cipher

Encryption
$a = 5$ is invertible?
Encode the plaintext
Encrypt each value



Affine Cipher

plaintext = 8 5 12 12 15

Encryption
$a = 5$ is invertible?
Encode the plaintext
Encrypt each value

$c_i = (a \cdot p_i + b)\ \%\ 26$:

$c_0 = (5 * 8 + 4)\%26 = 18$

$c_1 = (5 * 5 + 4)\%26 = 3$

$c_2 = (5 * 12 + 4)\%26 = 12$

$c_3 = (5 * 12 + 4)\%26 = 12$

$c_4 = (5 * 15 + 4)\%26 = 1$

Affine Cipher

Decryption
Compute a^{-1}
Encode the ciphertext
decrypt each value

Affine Cipher

Decryption
Compute a^{-1}
Encode the ciphertext
decrypt each value

$q := \left\lfloor \frac{a}{b} \right\rfloor$	$r := a \,\%\, b$	$a := b$	$b := r$	$x := x_0 - q \cdot x_1$	$y := y_0 - q \cdot y_1$	$x_0 := x_1$	$y_0 := y_1$	$x_1 := x$	$y_1 := y$

Affine Cipher

Decryption
Compute a^{-1}
Encode the ciphertext
decrypt each value

Initial values

$q := \left\lfloor \frac{a}{b} \right\rfloor$	$r := a \,\%\, b$	$a := b$	$b := r$	$x := x_0 - q \cdot x_1$	$y := y_0 - q \cdot y_1$	$x_0 := x_1$	$y_0 := y_1$	$x_1 := x$	$y_1 := y$
-	-	5	26	-	-	1	0	0	1

Affine Cipher

Decryption
Compute a^{-1}
Encode the ciphertext
decrypt each value

$q := \left\lfloor \frac{a}{b} \right\rfloor$	$r := a \,\%\, b$	$a := b$	$b := r$	$x := x_0 - q \cdot x_1$	$y := y_0 - q \cdot y_1$	$x_0 := x_1$	$y_0 := y_1$	$x_1 := x$	$y_1 := y$
-	-	5	26	-	-	1	0	0	1
0	5	26	5	1	0	0	1	1	0

Affine Cipher

Decryption
Compute a^{-1}
Encode the ciphertext
decrypt each value

$q := \left\lfloor \frac{a}{b} \right\rfloor$	$r := a \,\%\, b$	$a := b$	$b := r$	$x := x_0 - q \cdot x_1$	$y := y_0 - q \cdot y_1$	$x_0 := x_1$	$y_0 := y_1$	$x_1 := x$	$y_1 := y$
-	-	5	26	-	-	1	0	0	1
0	5	26	5	1	0	0	1	1	0
5	1	5	1	-5	1	1	0	-5	1

Affine Cipher

Decryption
Compute a^{-1}
Encode the ciphertext
decrypt each value

$q := \left\lfloor \frac{a}{b} \right\rfloor$	$r := a \,\%\, b$	$a := b$	$b := r$	$x := x_0 - q \cdot x_1$	$y := y_0 - q \cdot y_1$	$x_0 := x_1$	$y_0 := y_1$	$x_1 := x$	$y_1 := y$
-	-	5	26	-	-	1	0	0	1
0	5	26	5	1	0	0	1	1	0
5	1	5	1	-5	1	1	0	-5	1
5	0	1	0	26	-5	-5	1	26	-5

Affine Cipher

Decryption
Compute a^{-1}
Encode the ciphertext
decrypt each value

Stop when $b = 0$

$q := \left\lfloor \frac{a}{b} \right\rfloor$	$r := a \,\%\, b$	$a := b$	$b := r$	$x := x_0 - q \cdot x_1$	$y := y_0 - q \cdot y_1$	$x_0 := x_1$	$y_0 := y_1$	$x_1 := x$	$y_1 := y$
-	-	5	26	-	-	1	0	0	1
0	5	26	5	1	0	0	1	1	0
5	1	5	1	-5	1	1	0	-5	1
5	0	1	0	26	-5	-5	1	26	-5

Affine Cipher

Decryption
Compute a^{-1}
Encode the ciphertext
decrypt each value

Modular inverse is $a^{-1} = x_0 = -5$

$q := \left\lfloor \frac{a}{b} \right\rfloor$	$r := a \,\%\, b$	$a := b$	$b := r$	$x := x_0 - q \cdot x_1$	$y := y_0 - q \cdot y_1$	$x_0 := x_1$	$y_0 := y_1$	$x_1 := x$	$y_1 := y$
-	-	5	26	-	-	1	0	0	1
0	5	26	5	1	0	0	1	1	0
5	1	5	1	-5	1	1	0	-5	1
5	0	1	0	26	-5	-5	1	26	-5

gcd

Coeff. x, a^{-1}

Coeff. y

Affine Cipher

Decryption
Compute a^{-1}
Encode the ciphertext
decrypt each value

18 3 12 12 1

Affine Cipher

Decryption
Compute a^{-1}
Encode the ciphertext
decrypt each value

18 3 12 12 1

$p_i = a^{-1}(c_i - b)\ \%\ 26$:

$p_0 = -5\ (18 - 4)\%\ 26 = 8 \rightarrow$ H

$p_1 = -5\ (3 - 4)\%\ 26 = 5 \rightarrow$ E

$p_2 = -5\ (12 - 4)\%\ 26 = 12 \rightarrow$ L

$p_3 = -5\ (12 - 4)\%\ 26 = 12 \rightarrow$ L

$p_4 = -5\ (1 - 4)\%\ 26 = 15 \rightarrow$ O

Affine Cipher

TASK

- Write the $\gcd(a, b)$ function

Algorithm 35: Euclidean Algorithm for GCD

Input: a, b
Output: GCD of a and b
while $b \neq 0$ **do**
    $r \leftarrow a \mod b$;
    $a \leftarrow b$;
    $b \leftarrow r$;
end
return a

Affine Cipher

TASK

- Write the egcd(a, b) function

Algorithm 37: Iterative Extended Euclidean Algorithm

Input: a, b

Output: GCD of a and b, and coefficients x and y such that $ax + by = \gcd(a, b)$

Function `IterEGCD(`a, b`)`**:**

- $x_0 \leftarrow 1$, $y_0 \leftarrow 0$;
- $x_1 \leftarrow 0$, $y_1 \leftarrow 1$;
- **while** $b \neq 0$ **do**
  - $q \leftarrow \lfloor \frac{a}{b} \rfloor$;
  - $r \leftarrow a \mod b$;
  - $a \leftarrow b$;
  - $b \leftarrow r$;
  - $x \leftarrow x_0 - q \times x_1$;
  - $y \leftarrow y_0 - q \times y_1$;
  - $x_0 \leftarrow x_1$, $y_0 \leftarrow y_1$;
  - $x_1 \leftarrow x$, $y_1 \leftarrow y$;
- **end**
- **Return** a, x_0, y_0

Return `IterEGCD(`a, b`)`

Affine Cipher

TASK

- Write the $affine\_encrypt$ function

Algorithm 41: Affine Cipher Encryption

Input: $plaintext, multiplier, shift, modulus$
Output: $ctxt$

if $is\_valid\_key(multiplier, modulus) = False$ **then**
  return Error
end
$ctxt \leftarrow$ "";
foreach p *in* $plaintext$ **do**
  if $p \in [A : Z]$ **then**
    $c \leftarrow multiplier * p + shift$;
    $c \leftarrow \text{custom\_mod}(c, A, Z)$;
    $ctxt \leftarrow ctxt + c$;
  end
  else
    $ctxt \leftarrow ctxt + p$;
  end
end
return $ctxt$;

Affine Cipher

TASK

- Write the $affine\_decrypt$ function

Algorithm 42: Affine Cipher Decryption

Input: $ciphertext, multiplier, shift, modulus$
Output: $ptxt$

$(x, inv, y) \leftarrow \text{egcd}(multiplier, modulus)$;
$ptxt \leftarrow$ "";
foreach c *in* $ciphertext$ **do**
    if $c \in [A : Z]$ **then**
        $p \leftarrow inv \times (c - shift)$;
        $p \leftarrow \text{custom\_mod}(p, A, Z)$;
        $ptxt \leftarrow ptxt + p$;
    end
    else
        $ptxt \leftarrow ptxt + c$;
    end
end
return $ptxt$;

Affine Cipher

TASK

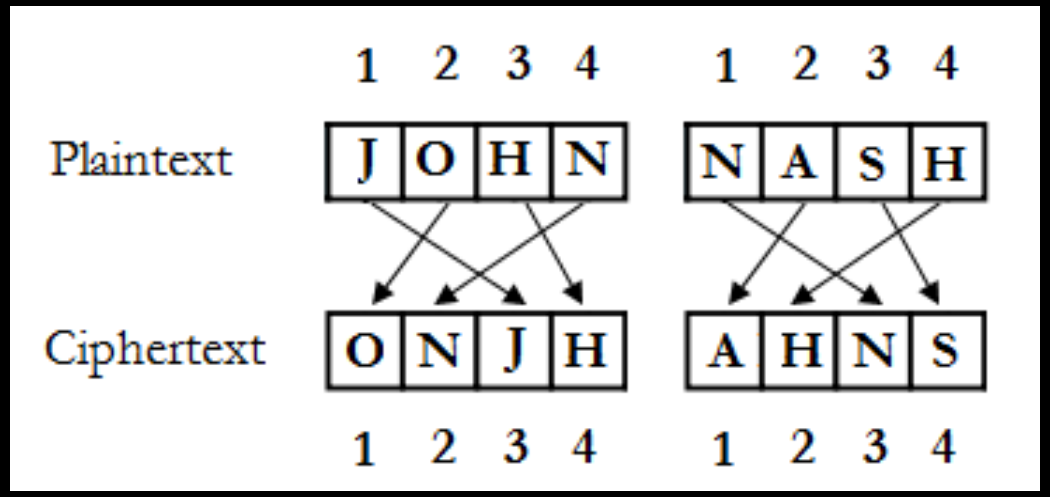
- Brute force this affine cipher ciphertext: "IBPW{J3G_FS0GE3K_BWF55P5B_BPOE3K}"

Content

Content
Caesar and Rot13 Ciphers
Affine Cipher
Permutation Cipher

Permutation Cipher

- Divides the plaintext into blocks and applies the same permutation to each block.
  - It belongs to the family of **transposition ciphers**.
  - Rearranges the letters of a text but does no substitutions.
- If the last block is too short, it is padded with nulls to fill it out.



Permutation Cipher

- Hands-on example: encrypt this message with the permutation (4 2 5 1 3 0):

THIS MESSAGE IS ENCRYPTED WITH A TRANSPOSITION CIPHER

Permutation Cipher

- Hands-on example: encrypt this message with the permutation (4 2 5 1 3 0):

THIS MESSAGE IS ENCRYPTED WITH A TRANSPOSITION CIPHER

- Break the message into blocks 6 letters = size of the key

THISME	SSAGEI	SENCRY	PTEDWI	THATRA	NSPOSI	TIONCI	PHER--

Permutation Cipher

- Hands-on example: encrypt this message with the permutation (4 2 5 1 3 0):

THIS MESSAGE IS ENCRYPTED WITH A TRANSPOSITION CIPHER

- Break the message into blocks 6 letters = size of the key

THISME	SSAGEI	SENCRY	PTEDWI	THATRA	NSPOSI	TIONCI	PHER--

- Apply the permutation to each block to get the ciphertext:

******	******	******	******	******	******	******	******

Permutation Cipher

- Hands-on example: encrypt this message with the permutation (4 2 5 1 3 0):

THIS MESSAGE IS ENCRYPTED WITH A TRANSPOSITION CIPHER

- Break the message into blocks 6 letters = size of the key

THISME	SSAGEI	SENCRY	PTEDWI	THATRA	NSPOSI	TIONCI	PHER--

- Apply the permutation to each block to get the ciphertext:

****T*	******	******	******	******	******	******	******

Permutation Cipher

- Hands-on example: encrypt this message with the permutation (4 2 5 1 3 0):

THIS MESSAGE IS ENCRYPTED WITH A TRANSPOSITION CIPHER

- Break the message into blocks 6 letters = size of the key

THISME	SSAGEI	SENCRY	PTEDWI	THATRA	NSPOSI	TIONCI	PHER--

- Apply the permutation to each block to get the ciphertext:

**H*T*	******	******	******	******	******	******	******

Permutation Cipher

- Hands-on example: encrypt this message with the permutation (4 2 5 1 3 0):

THIS MESSAGE IS ENCRYPTED WITH A TRANSPOSITION CIPHER

- Break the message into blocks 6 letters = size of the key

THISME	SSAGEI	SENCRY	PTEDWI	THATRA	NSPOSI	TIONCI	PHER--

- Apply the permutation to each block to get the ciphertext:

**H*TI	******	******	******	******	******	******	******

Permutation Cipher

- Hands-on example: encrypt this message with the permutation (4 2 5 1 3 0):

THIS MESSAGE IS ENCRYPTED WITH A TRANSPOSITION CIPHER

- Break the message into blocks 6 letters = size of the key

THISME	SSAGEI	SENCRY	PTEDWI	THATRA	NSPOSI	TIONCI	PHER--

- Apply the permutation to each block to get the ciphertext:

*SH*TI	******	******	******	******	******	******	******

Permutation Cipher

- Hands-on example: encrypt this message with the permutation (4 2 5 1 3 0):

THIS MESSAGE IS ENCRYPTED WITH A TRANSPOSITION CIPHER

- Break the message into blocks 6 letters = size of the key

THISME	SSAGEI	SENCRY	PTEDWI	THATRA	NSPOSI	TIONCI	PHER--

- Apply the permutation to each block to get the ciphertext:

*SHMTI	******	******	******	******	******	******	******

Permutation Cipher

- Hands-on example: encrypt this message with the permutation (4 2 5 1 3 0):

THIS MESSAGE IS ENCRYPTED WITH A TRANSPOSITION CIPHER

- Break the message into blocks 6 letters = size of the key

THISME	SSAGEI	SENCRY	PTEDWI	THATRA	NSPOSI	TIONCI	PHER--

- Apply the permutation to each block to get the ciphertext:

ESHMTI	******	******	******	******	******	******	******

Permutation Cipher

- Hands-on example: encrypt this message with the permutation (4 2 5 1 3 0):

THIS MESSAGE IS ENCRYPTED WITH A TRANSPOSITION CIPHER

- Break the message into blocks 6 letters = size of the key

THISME SSAGEI SENCRY PTEDWI THATRA NSPOSI TIONCI PHER--

- Apply the permutation to each block to get the ciphertext:

ESHMTI IGSESA YCERSN IDTWPE ATHRTA IOSSNP INICTO -RH-PE

Permutation Cipher

To find the inverse of a permutation:

- write the permutation under the identity:

Identity	**0**	**1**	**2**	**3**	**4**	**5**
Key	4	2	5	1	3	0

- Reorder pairs so that the second row is the identity.

Inv(Key)	**5**	**3**	**1**	**4**	**0**	**2**
Identity	0	1	2	3	4	5

Permutation Cipher

- Hands-on example: decrypt this message with the permutation (5 3 1 4 0 2):

ESHMTI	IGSESA	YCERSN	IDTWPE	ATHRTA	IOSSNP	INICTO	-RH-PE

******	******	******	******	******	******	******	******

Permutation Cipher

- Hands-on example: decrypt this message with the permutation (5 3 1 4 0 2):

ESHMTI	IGSESA	YCERSN	IDTWPE	ATHRTA	IOSSNP	INICTO	-RH-PE

*****E	******	******	******	******	******	******	******

Permutation Cipher

- Hands-on example: decrypt this message with the permutation (5 3 1 4 0 2):

ESHMTI	IGSESA	YCERSN	IDTWPE	ATHRTA	IOSSNP	INICTO	-RH-PE

***S*E	******	******	******	******	******	******	******

Permutation Cipher

- Hands-on example: decrypt this message with the permutation (5 3 1 4 0 2):

ESHMTI	IGSESA	YCERSN	IDTWPE	ATHRTA	IOSSNP	INICTO	-RH-PE

*H*S*E	******	******	******	******	******	******	******

Permutation Cipher

- Hands-on example: decrypt this message with the permutation (5 3 1 4 0 2):

ESHMTI	IGSESA	YCERSN	IDTWPE	ATHRTA	IOSSNP	INICTO	-RH-PE

*H*SME	******	******	******	******	******	******	******

Permutation Cipher

- Hands-on example: decrypt this message with the permutation (5 3 1 4 0 2):

ESHMTI	IGSESA	YCERSN	IDTWPE	ATHRTA	IOSSNP	INICTO	-RH-PE

TH*SME	******	******	******	******	******	******	******

Permutation Cipher

- Hands-on example: decrypt this message with the permutation (5 3 1 4 0 2):

ESHMTI	IGSESA	YCERSN	IDTWPE	ATHRTA	IOSSNP	INICTO	-RH-PE

THISME	******	******	******	******	******	******	******

Permutation Cipher

- Hands-on example: decrypt this message with the permutation (5 3 1 4 0 2):

ESHMTI	IGSESA	YCERSN	IDTWPE	ATHRTA	IOSSNP	INICTO	-RH-PE

THISME	SSAGEI	SENCRY	PTEDWI	THATRA	NSPOSI	TIONCI	PHER--

Permutation Cipher

TASK

- Write a function to get a random permutation of a specific length, and another function to find the inverse of the permutation

Algorithm 43: Generate Random Permutation Key

Input: *size*
Output: *key*

Use *permutations* in from *itertools* in Python
return a random permutation from the list of permutations.

Algorithm 44: Calculate Inverse Permutation Key

Input: *key*
Output: *ik*

$ik \leftarrow \text{list}(\text{range}(\text{len}(key)))$;
for $i \leftarrow 0$ **to** $len(key) - 1$ **do**
    $k \leftarrow key[i]$;
    $ik[k] \leftarrow i$;
end
return ik;

Permutation Cipher

TASK

- Write a permutation encryption function

Algorithm 45: Permutation Cipher Encryption

Input: *plaintext*, *key*
Output: *ctxt*

pad(*plaintext*, *len*(*key*));
Split the plaintext into blocks, each of size *len*(*key*);
ctxt_blocks ← [' '] ×len(*ptxt_blocks*);
foreach *block in ptxt_blocks* **do**
    permuted ← [''] × len(*key*);
    for *i* ← 0 **to** *len*(*block*) − 1 **do**
        permuted[*key*[*i*]] ← *block*[*i*];
    end
    Append the permuted block to the ciphertext;
end
return *ctxt*;

Permutation Cipher

TASK

- Write a permutation decryption function

Algorithm 46: Permutation Cipher Decryption

Input: *ciphertext*, *key*
Output: *ptxt*

keylen ← len(*key*);
key ← inv_key(*key*);
Split the ctxt into blocks of size *keylen*
ptxt_blocks ← [' '] ×len(*ctxt_blocks*);
foreach *block in ctxt_blocks* **do**
    permuted ← [''] × len(*key*);
    for *i* ← 0 **to** *len*(*block*) − 1 **do**
        permuted[*key*[*i*]] ← *block*[*i*];
    end
    Append the permuted block to the plaintext
end
return *ctxt*;