

Kriptografi

Kriptografi Klasik (Cipher)

Tugas Kelompok

- Arief Saferman (1806148656)
- Kevin Darmawan(1806148744)
- Ramadhan K.S (1806148826)
- Yogie Wisesa (1806148851)

1. Terdapat script matlab untuk menguji coba kelima fungsi cipher yang kelompok kami buat. Melalui script 'Source.m' akan membaca teks dari sebuah file bernama 'sample.txt' kemudian melakukan enkripsi menggunakan lima algoritma cipher. Key untuk setiap algoritma cipher telah di-define dibagian awal script.

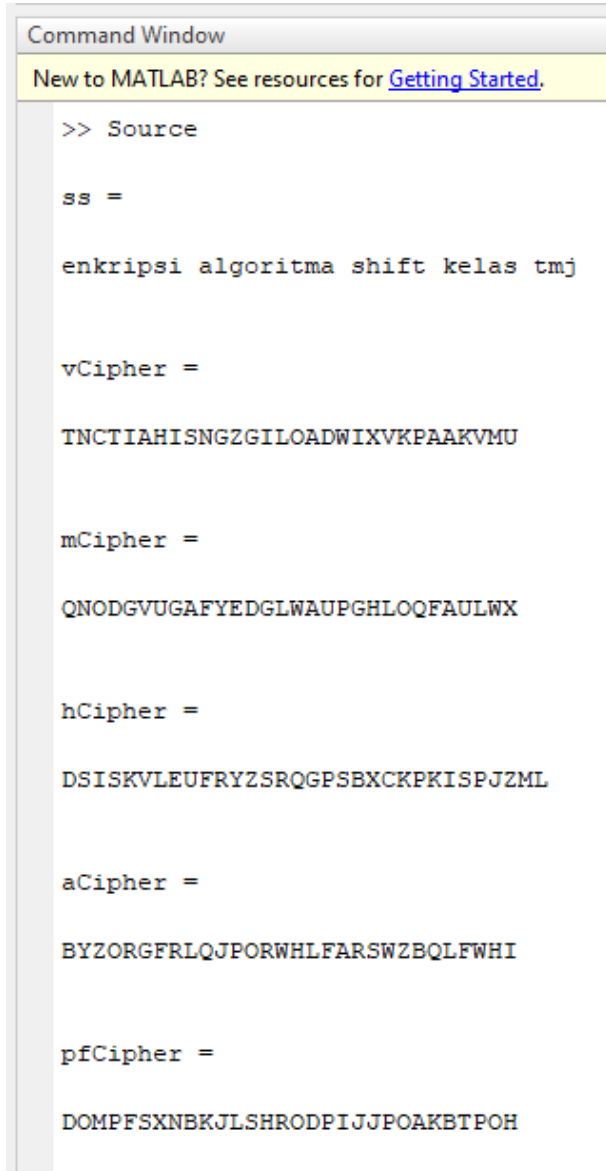
```
2  viginereKey = 'pascal';
3  multiplicativeKey = 17;
4  hillKey = [9 7 11 13; 4 7 5 6; 2 21 14 9; 3 23 21 8];
5  affineKey1 = 17;
6  affineKey2 = 11;
7  playfairKey = [0:1:4; 5:1:9; 10:1:14; 15:1:19; 20:1:24];
8  playfairKey = char(playfairKey + 65);
9
10 data = importdata('sample.txt');
11
12 for i = 1:size(data)
13
14     ss = char(data(i));
15     display(ss);
16
17     vCipher = Viginere(ss, viginereKey);
18     display(vCipher);
19
20     mCipher = Multiplicative(ss, multiplicativeKey);
21     display(mCipher);
22
23     hCipher = HillCipher(ss, hillKey);
24     display(hCipher);
25
26     aCipher = Affine(ss, affineKey1, affineKey2);
27     display(aCipher);
28
29     pfCipher = PlayFair(ss, playfairKey);
30     display(pfCipher);
31
32 end
```

sample - Notepad

File Edit Format View Help

enkripsi algoritma shift kelas tmj

2. Berikut adalah Output yang diberikan oleh MatLab:



```
Command Window
New to MATLAB? See resources for Getting Started.

>> Source

ss =

enkripsi algoritma shift kelas tmj

vCipher =

TNCTIAHISNGZGILOADWIXVKPAKVMU

mCipher =

QNODGVUGAFYEDGLWAUPGHLOQFAULWX

hCipher =

DSISKVLEUFYZSRQGPSBXCKPKISPJZML

aCipher =

BYZORGFRQLQJPORWHLFARSWZBQLFWHI

pfCipher =

DOMPFSXNBKJLSHRODPIJJPOAKBTPOH
```

Setiap fungsi telah kami standardisasi agar memperjelas implementasi dari teori yang sedang kami pelajari. Standardisasi tersebut berupa:

- Sumber teks tidak sensitif terhadap kapital. Walaupun pada 'sample.txt' kita mencampur dengan huruf kapital, fungsi cipher akan menganggap sebagai satu seri dengan nilai 0 – 25.
- Sumber teks hanya menerima alfabet (a-z/A-Z). Apabila terdapat simbol pada 'sample.txt', setiap fungsi cipher akan mengabaikannya.
- Setiap fungsi cipher akan mengabaikan spasi. Apabila terdapat spasi pada sumber teks, maka enkripsi akan menggabungkannya sebagai satu string kontinu.
- Output akan ditampilkan dalam karakter kapital.

a) Affine Cipher

Karakter alfabet akan dianggap sebagai suatu nilai antara 0 – 25. Kemudian nilai cipher akan dikalikan dengan key1 dan ditambah dengan key2. Hasil-nya akan di-mod 26 untuk tetap berada pada range 0 – 25. Setelah seluruh karakter diproses, sebelum di return akan ditambah 65 untuk mengubahnya ke bentuk kapital dalam bentuk ASCII.

```
1 function str = Affine(string, key1, key2)
2     str = [];
3     for i = 1:length(string)
4         x = string(i);
5         if (x >= 65 && x <= 90) y = x - 65;
6         elseif (x >= 97 && x <= 122) y = x - 97;
7         else continue;
8     end
9     y = (key1 * y + key2);
10    str(end+1) = mod(y, 26);
11 end
12 str = char(str + 65);
13 end
```

b) Multiplicative Cipher

Proses yang dikerjakan hampir sama dengan affine cipher. Namun dengan ini mendapat nilai cipher dengan mengalikan nilai alfabet 0 – 25 dengan key.

```
1 function str = Multiplicative(string, key)
2     str = [];
3     for i = 1:length(string)
4         x = string(i);
5         if (x >= 65 && x <= 90) y = x - 65;
6         elseif (x >= 97 && x <= 122) y = x - 97;
7         else continue;
8     end
9     str(end + 1) = mod(y * key, 26);
10 end
11 str = char(str + 65);
12 end
13
```

c) Play Fair Cipher

- Corner Cases dan Persiapan

Sumber teks dan key akan diubah kedalam bentuk kapital. Ini digunakan untuk melakukan perbandingan, sehingga kita dapat menghilangkan karakter dari sumber yang tidak ada dalam matrix key. Kemudian kita harus memastikan bahwa panjang teks tersebut dapat saling berpasangan. Kedua hal berikut merupakan tuntutan algoritma Play-Fair.

- Contoh saling berpasangan: krypto = kr ip to
- Contoh tidak berpasangan: kuaci = ku ac i (karakter 'i' tidak memiliki pasangan)

Cara yang kami ambil adalah dengan mengecek apakah panjang teks habis dibagi dua. Jika tidak, maka kita akan insert satu karakter di belakang teks dengan nilai kesepakatan misalnya 'X'. Setelah itu, kita lebarkan ukuran matriks key dengan menambah satu kolom dan satu baris. Ini dilakukan untuk memudahkan komputer dalam proses looping.

```
1  function str = PlayFair(string, key)
2  -     str = [];
3      % set all to upper case for shifting
4  -     string = upper(string);
5  -     key = upper(key);
6      % remove character that's not in the key
7  -     temp = [];
8  -     for i = 1:length(string)
9  -         x = string(i);
10 -         [row, col] = find(key == x, 1);
11 -         if (isempty(row)) continue; end
12 -         temp(end + 1) = x;
13 -     end
14 -     string = temp;
15     % string length should be divisible by 2
16     % the algorithm should detect this with the same char
17     % (x1 == x2), so it can fill the gap with 'X'
18 -     len = length(string);
19 -     if (mod(len, 2) ~= 0) string(end+1) = 'X'; end
20     %increase matrix size for by 1-1
21 -     key(end+1,end+1) = 0;
22 -     key(:,end) = key(:,1);
23 -     key(end,:) = key(1,:);
```

- Algoritma

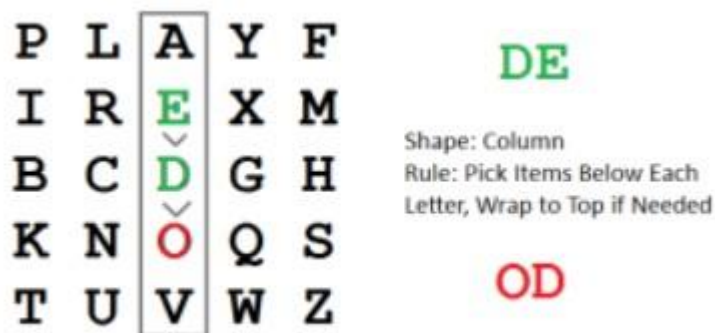
Melakukan looping dan memproses setiap karakter secara berpasangan. Apabila kedua karakter yang di-evaluasi sama, maka karakter kedua diisi dengan nilai kesepakatan sebelumnya yaitu 'X' (seperti saat kita mengisi karakter yang tidak memiliki pasangan).

Contoh: balloon → ba ll oo n → ba ll oo nx

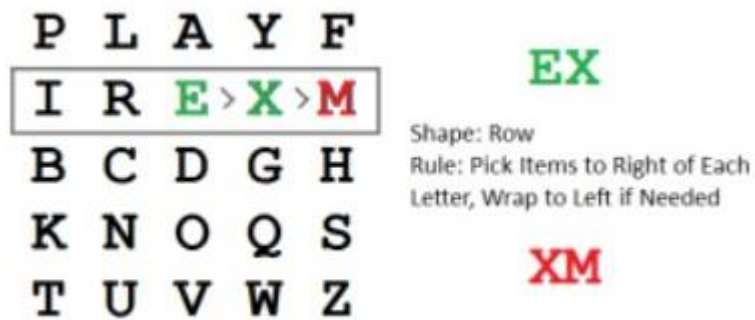
Sehingga: ba ll oo nx → ba lx ox nx

Terdapat beberapa metode yang berbeda dalam memproses karakter cipher berdasarkan kondisi berikut:

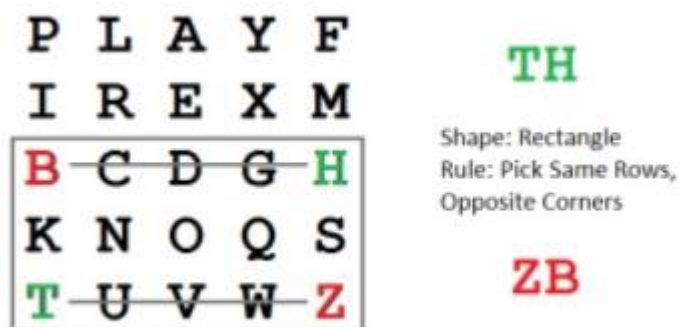
a) Apabila Posisi Kolom Sama



b) Apabila Posisi Baris Sama



c) Apabila Tidak memenuhi Diatas



Apabila karakter berada di indeks akhir kolom pada matriks, misal: XM, maka nilai cipher-nya adalah MI, kembali ke indeks kolom awal. Begitu juga untuk karakter yang berada pada indeks akhir baris pada matriks, misal: OV, maka nilai cipher-nya adalah VA. Untuk mempermudah pemrograman, kami memperbesar matriks dengan menambah satu kolom dan satu baris seperti yang dilakukan pada tahap persiapan. Ini membuat program jadi lebih mudah dengan memanfaatkan indexing key (row1+1) tanpa takut keluar dari matriks key.

```
% play-fair
for i = 1:2:length(string)
    x1 = string(i);
    x2 = string(i+1);
    % char should be different
    if (x1 == x2) x2 = 'X'; end
    % find index in the key
    [row1, col1] = find(key == x1, 1);
    [row2, col2] = find(key == x2, 1);
    if (row1 == row2)
        str(i) = key(row1, col1 + 1);
        str(i+1) = key(row2, col2 + 1);
    elseif (col1 == col2)
        str(i) = key(row1 + 1, col1);
        str(i+1) = key(row2 + 1, col2);
    else
        str(i) = key(row1, col2);
        str(i+1) = key(row2, col1);
    end
end
str = char(str);
end
```

d) Hill Cipher

Pertama kita akan mengambil hanya karakter alfabet dari sebuah teks. Kemudian kita harus mengubah bentuk teks menjadi matriks yang dapat dikali dengan matriks key. Oleh karena itu kita buat jumlah kolom pada teks sama dengan jumlah baris pada key. Apabila terdapat nilai yang kurang untuk mengisi elemen matriks dari teks, maka kita isi saja dengan nilai kesepakatan misalnya 'z'. Kemudian setiap elemen pada matriks akan dibuat menjadi sebuah alfabet dengan nilai 0 – 25. Setelah itu, matriks teks dikalikan dengan matriks key dan di mod 26. Selanjutnya kita ubah lagi bentuk matriks kedalam bentuk 1 dimensi layaknya sebuah teks. Dan terakhir akan ditambahkan dengan 65 untuk memberikan karakter kapital dalam ASCII.

```

1  function str = HillCipher(string, key)
2  -     str = [];
3  -     string = string(isstrprop(string, 'alpha'));
4  -     string = lower(string);
5  -     [rows cols] = size(key);
6  -     remainder = mod(length(string), rows);
7  -     if (remainder)
8  -         fill = 'z' * ones(1, rows - remainder);
9  -         string = [string, fill];
10 -     end
11 -     % matrix of string should be able to multiply with matrix of key
12 -     % if not, fill the gap with 'z' values
13 -     rowsize = length(string) / rows;
14 -     str = reshape(string, rows, rowsize).';
15 -     str = str - 97;
16 -     str = str * key;
17 -     str = mod(str, 26);
18 -     str = reshape(str.', 1, []);
19 -     str = char(str + 65);
20 - end

```

e) Viginere Cipher

Loop kesetiap elemen (karakter) pada teks. Nilai dari suatu elemen pada teks akan ditambah dengan nilai suatu elemen pada key kemudian di mod 26. Apabila key lebih pendek dibanding teks, maka indeks pada key akan kembali lagi dari awal.

```

1  function str = Viginere(string, key)
2  -     str = [];
3  -     key = double(lower(key) - 97);
4  -     j = 1;
5  -     for i = 1:length(string)
6  -         x = string(i);
7  -         if (x >= 65 && x <= 90) y = x - 65;
8  -         elseif (x >= 97 && x <= 122) y = x - 97;
9  -         else continue;
10 -        end
11 -        str(end + 1) = mod(y + key(j), 26);
12 -        if (j == length(key)) j = 0; end
13 -        j = j + 1;
14 -    end
15 -    str = char(str + 65);
16 - end

```