Nama :

- Salsabila Karin (140810190015)
- Prianggara Hadyan Almer (140810190065)

Kelas : A

Praktikum Kriptografi

Hill Cipher

1. Exercise

Enkripsi dengan key $\begin{bmatrix}7 & 6\\2 & 5\end{bmatrix}$

Plain teks : GOPHER

6 14| 15 7| 4 17

$\begin{bmatrix}7 & 6\\2 & 5\end{bmatrix}\begin{bmatrix}6\\14\end{bmatrix}$

= 126 mod26 = 22

82 mod26 = 4

$\begin{bmatrix}7 & 6\\2 & 5\end{bmatrix}\begin{bmatrix}15\\7\end{bmatrix}$

= 147 mod 26 = 17

65 mod 26 = 13

$\begin{bmatrix}7 & 6\\2 & 5\end{bmatrix}\begin{bmatrix}4\\17\end{bmatrix}$

= 130 mod 26 = 0

93 mod 26 = 15

22 4| 17 13 |0 15

Cipher teks : W E R N A P

A. Penjelasan Hill Cipher

Hill Cipher merupakan salah satu algoritma kriptografi yang memanfaatkan matriks sebagai kunci untuk melakukan enkripsi dan Dekripsi dan aritmatika modulo. Setiap karakter pada plaintext ataupun ciphertext dikonversikan kedalam bentuk angka. Enkripsi dilakukan dengan mengalikan matriks kunci dengan matriks plaintext, sedangkan Dekripsi dilakukan dengan mengalikan invers matriks kunci dengan matriks ciphertext. Karena itulah, Hill Cipher hanya bisa menggunakan matriks persegi sebagai matriks kuncinya. Invers semu atau pseudo invers dapat dimanfaatkan pada algoritma

Hill Cipher, sehingga matriks kunci yang digunakan tidak terbatas pada matriks persegi saja. Penggunaan matriks persegi panjang menjadikan ciphertext lebih panjang dari plaintext. Hal ini tentunya membuat pesan menjadi lebih tersamarkan. Pada tulisan ini, penulis menggunakan modulo 95 artinya inputan data ada 95 simbol. Untuk mempermudah penghitungan pada saat inisialisasi matriks kunci, proses enkripsi dan proses Dekripsi menggunakan program aplikasi C++.

B. Enkripsi Hill Cipher

Tahapan dalam melakukan enkripsi pada hill cipher adalah :

1. Tentukan Plaintext (pesan) selanjutnya, susun plaintext dalam bentuk blok matriks (2x1 jika ordo kunci 2x2, 3x1 jika ordo kunci 3x3).
2. Tentukan matriks kunci dengan persyaratan nilai determinasi matriks kunci harus nilai bilangan ganjil positif atau negatif.
3. Lakukan proses Enkripsi, dengan rumus:

$$C = Mk * Mp$$

Keterangan :

C = Ciphertext

Mk = Matriks Kunci

Mp = Matriks Plaintext

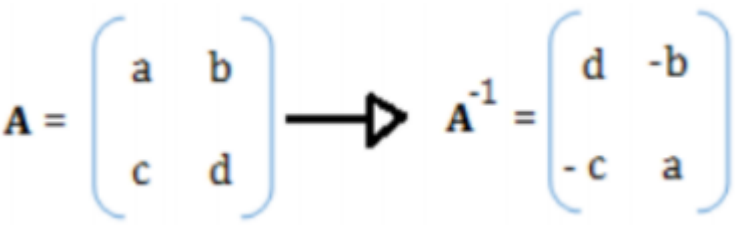
C. Dekripsi Hill Cipher

Tahapan dalam melakukan dekripsi pada Hill Cipher adalah :

1. Menentukan matriks Chipertext (Ct).
2. Tentukan determinan matriks kunci K.

$$\mathbf{A} = \begin{pmatrix} a & b \\ c & d \end{pmatrix} \longrightarrow \det \mathbf{A} = \begin{pmatrix} a & b \\ c & d \end{pmatrix} = ad - bc$$

3. Tentukan nilai invers modulo.
4. Tentukan invers matriks kunci K



5. Tentukan kunci dekripsi Hill Cipher K-1
6. Rumus dekripsi Hill Cipher

$$P = Mk^{-1} * Mc$$

Keterangan:

P = *Plaintext*

M_k^{-1} = Matriks *Kunci Invers*

M_c = Matriks *Ciphertext*

D. Mencari K

1. Menentukan Ciphertext dan Plaintext.
2. Mengubah Plaintext dan Ciphertext dalam bentuk numeric.
3. Membagi plaintext dan ciphertext sesuai dengan ordo yang ingin dicari, jika 3x3 maka plaintext dan ciphertext dibagi menjadi 3 huruf.
4. Menjadikan Plaintext dan Ciphertext ke dalam bentuk matriks
5. Rumus mencari Kunci :

$$K = C * P{-}1$$

Keterangan :

K = Kunci Matrix

C = Ciphertext

$P{-}1$ = Plaintext yang di invers

E. Screenshot

```
C:\Users\User\Documents>hillcipher
Masukkan key :
key[0][0]: 7
key[0][1]: 6
key[1][0]: 2
key[1][1]: 5

Program Hill Cipher 2x2
Menu :
1. Enkripsi
2. Dekripsi
3. Cari Key
4. Exit
Pilih Menu : 1

Input Plaintext: gopher
Ciphertext : WERNAP


Program Hill Cipher 2x2
Menu :
1. Enkripsi
2. Dekripsi
3. Cari Key
4. Exit
Pilih Menu : 2

Input Ciphertext: wernap
Plaintext : GOPHER


Program Hill Cipher 2x2
Menu :
1. Enkripsi
2. Dekripsi
3. Cari Key
4. Exit
Pilih Menu : 3

Masukan Plainteks : friday
Masukan Cipherteks : pqcfku
7       8
19      3
```

F. Pembahasan

```
11  void getInverseMatrix(int key[2][2])
12  {
13      int tempKey[2][2];
14      tempKey[0][0] = (int)(key[1][1]);
15      tempKey[0][1] = (int)((-1) * key[0][1]);
16      tempKey[1][0] = (int)((-1) * key[1][0]);
17      tempKey[1][1] = (int)(key[0][0]);
18      int determinant = (key[0][0] * key[1][1]) - (key[0][1] * key[1][0]);
19      int det_inv = 0;
20      int flag = 0;
21      for (int i = 0; i < 26; i++)
22      {
23          flag = (determinant * i) % 26;
24          if (flag < 0)
25          {
26              flag = flag + 26;
27          }
28          if (flag == 1)
29          {
30              det_inv = i;
31          }
32      }
33      for (int i = 0; i < 2; i++)
34          for (int j = 0; j < 2; j++)
35          {
36              if (tempKey[i][j] < 0)
37              {
38                  int tempNumber = tempKey[i][j] * det_inv;
39                  inversedKey[i][j] = ((tempNumber % 26) + 26) % 26;
40              }
41              else
42              {
43                  inversedKey[i][j] = (tempKey[i][j] * det_inv % 26);
44              }
45          }
46  }
```

Fungsi kodingan diatas adalah menginverskan matriks

```
string encrypt(string plain, int key[2][2])
{
    string cipher = "";
    int stringLength = plain.length();
    if (plain.length() % 2 == 1){
        stringLength += 1;
    }
    char plainMatrix[2][stringLength];
    int count = 0;
    for (int i = 0; i < stringLength / 2; i++)
        for (int j = 0; j < 2; j++)
        {
            if (plainMatrix[j][i] == 32)
            {
                break;
            }
            plainMatrix[j][i] = plain[count];
            count++;
        }
    for (int i = 0; i < stringLength / 2; i++){
        for (int j = 0; j < 2; j++){
            int tempCipher = 0;
            for (int k = 0; k < 2; k++){
                int l = key[j][k] * (plainMatrix[k][i] % 65);
                tempCipher += l;
            }
            tempCipher = (tempCipher % 26) + 65;
            cipher += (char)tempCipher;
        }
    }
    return cipher;
}
```

Fungsi kodingan diatas adalah untuk mencari enkripsi dari plaintext yang kita masukkan.

```
string decrypt(string cipher, int key[2][2])
{
    string plain = "";
    int stringLength = cipher.length();
    if (plain.length() % 2 == 1)
        stringLength = cipher.length() + 1;
    getInverseMatrix(key);
    char cipherMatrix[2][stringLength / 2];
    int count = 0;
    for (int i = 0; i < stringLength / 2; i++)
        for (int j = 0; j < 2; j++)
        {
            cipherMatrix[j][i] = cipher[count];
            count++;
        }

    for (int i = 0; i < cipher.length() / 2; i++)
    {
        for (int j = 0; j < 2; j++)
        {
            int tempPlain = 0;
            for (int k = 0; k < 2; k++)
            {
                int l = inversedKey[j][k] * (cipherMatrix[k][i] % 65);
                tempPlain += l;
            }
            tempPlain = (tempPlain % 26) + 65;
            plain += (char)tempPlain;
        }
    }
    return plain;
}
```

Fungsi kodingan di atas merupakan kodingan yang mencari dekripsi dari ciphertext.

```
116  int gcd(int a, int b) {
117      if (b == 0)
118          return a;
119      return gcd(b, a % b);
120  }
121  int findInvers(int m, int n)
122  {
123      int t0 = 0, t1 = 1, invers, q, r, b = m;
124      while (r != 1)
125      {
126          q = m / n;
127          r = m % n;
128          invers = t0 - q * t1;
129          if (invers < 0)
130          {
131              invers = b - (abs(invers) % b);
132          }
133          else
134          {
135              invers %= b;
136          }
137          t0 = t1;
138          t1 = invers;
139          m = n;
140          n = r;
141      }
142      return invers;
143  }
```

Fungsi GCD yaitu untuk menginput alpha dan beta yang akan dipakai untuk proses pencarian Invers pada void findInvers.

```
145  void findKey()
146  {
147      //deklarasi
148      string plainteks, cipherteks;
149      int key[2][2], det, detInv, adj[2][2], plainteksInv[2][2], plainMatrix[2][2], cipMatrix[2][2], counter;
150      int p, c;
151      int transpose[2][2];
152
153      //input plainteks
154      cout << "Masukan Plainteks : ";
155      cin.ignore();
156      getline(cin, plainteks);
157
158      //assign plainteks ke plainMatrix
159      counter = 0;
160      for (int i = 0; i < 2; i++)
161      {
162          for (int j = 0; j < 2; j++)
163          {
164              p = toupper(plainteks[counter]) - 65;
165              plainMatrix[i][j] = p;
166              counter++;
167          }
168      }
169
170      //input cipherteks
171      cout << "Masukan Cipherteks : ";
172      getline(cin, cipherteks);
173
174      //assign cipherteks ke cipMatrix
175      counter = 0;
176      for (int i = 0; i < 2; i++)
177      {
178          for (int j = 0; j < 2; j++)
179          {
180              c = toupper(cipherteks[counter]) - 65;
181              cipMatrix[i][j] = c;
182              counter++;
183          }
184      }
```

Fungsi kodingan void findKey untuk mencari key dari plaintext dan ciphertext yang sudah diketahui.

```
186     //mencari determinan
187     det = (plainMatrix[0][0] * plainMatrix[1][1]) - (plainMatrix[0][1] * plainMatrix[1][0]);
188     if (gcd(det, 26) == 1)
189     {
190         //mencari inverse dari determinan
191         detInv = findInvers(26, det);
192 
193         //menghitung matriks adjoin
194         adj[0][0] = plainMatrix[1][1];
195         adj[0][1] = (-1) * plainMatrix[0][1];
196         adj[1][0] = (-1) * plainMatrix[1][0];
197         adj[1][1] = plainMatrix[0][0];
198 
199         //menghitung matriks invers dari plainteks
200         for (int i = 0; i < 2; i++)
201         {
202             for (int j = 0; j < 2; j++)
203             {
204                 plainteksInv[i][j] = detInv * adj[i][j];
205                 if (plainteksInv[i][j] < 0)
206                 {
207                     plainteksInv[i][j] = 26 - (abs(plainteksInv[i][j]) % 26);
208                 }
209                 else
210                 {
211                     plainteksInv[i][j] = plainteksInv[i][j];
212                     plainteksInv[i][j] = plainteksInv[i][j] % 26;
213                 }
214             }
215         }
216 
217         //mencari key
218         for (int i = 0; i < 2; i++)
219         {
220             for (int j = 0; j < 2; j++)
221             {
222                 key[i][j] = 0;
223                 for (int k = 0; k < 2; k++)
224                 {
225                     key[i][j] += (plainteksInv[i][k] * cipMatrix[k][j]);
226                 }
227                 key[i][j] %= 26;
228             }
229         }
```

gambar diatas merupakan cara mencari determinan (line 186-215) untuk digunakan dalam mencari key (line 217-229)

```
231         //output key
232         for (int i = 0; i < 2; i++)
233         {
234             for (int j = 0; j < 2; j++)
235             {
236                 transpose[j][i] = key[i][j];
237             }
238         }
239 
240         for (int i = 0; i < 2; i++)
241         {
242             for (int j = 0; j < 2; j++)
243             {
244                 cout << (transpose[i][j]) << "\t";
245             }
246             cout << endl;
247         }
248     }
249     else
250     {
251         cout << "Determinan tidak relatif prima dengan jumlah huruf" << endl;
252         cout << "Key tidak dapat dicari" << endl
253              << endl;
254     }
255     system("pause");
256     system("cls");
257 }
```

pada (line 231 - 248) merupakan kodingan jika key ditemukan dan apabila key tidak ditemukan maka akan muncul kalimat pada line 251 dan 252

```cpp
int main()
{
    bool menuActive = true;
    int key[2][2];
    cout << "Masukkan key : " << endl;
    for (int i = 0; i < 2; i++){
        for (int j = 0; j < 2; j++){
            cout << "key[" << i << "][" << j << "]: ";
            cin >> key[i][j];
        }
    }
    string plain, cipher;
    int pil;
    while (menuActive){
        cout << "\nProgram Hill Cipher 2x2" << endl;
        cout << "Menu : " << endl;
        cout << "1. Enkripsi" << endl;
        cout << "2. Dekripsi" << endl;
        cout << "3. Cari Key" << endl;
        cout << "4. Exit" << endl;
        cout << "Pilih Menu : ";
        cin >> pil;
        switch (pil){
        case 1:
            cout << "\nInput Plaintext: ";
            cin.ignore();
            getline(cin, plain);
            plain = removeSpaces(plain);
            transform(plain.begin(), plain.end(), plain.begin(), ::toupper);
            cout << "Ciphertext : " << encrypt(plain, key) << endl
                 << endl;
            break;
        case 2:
            cout << "\nInput Ciphertext: ";
            cin.ignore();
```

```
            getline(cin, cipher);
            cipher = removeSpaces(cipher);
            transform(cipher.begin(), cipher.end(), cipher.begin(), ::toupper);
            cout << "Plaintext : " << decrypt(cipher, key) << endl
                 << endl;
            break;
        case 3:
            cout << endl;
            findKey();
            break;
        case 4:
            menuActive = false;
            break;
        default:
            cout << "\nPilihan salah" << endl;
            break;
        }
    }
}
```

kodingan diatas adalah main berfungsi untuk menjadi menu menjalankan kodingan, mulai dari enkripsi, dekripsi, edit key, dan cari key.