Nama & NPM Anggota : - M. Rafiq ‘A (140810190011)
- Aidil Fitra (140810190053)

Pembahasan Program Hill Cipher

Class

1. Class untuk mencari modinverse

```
int modInverse(int a, int m)
{
    a = a % m;
    for (int x = -m; x < m; x++)
        if ((a * x) % m == 1)
            return x;
    return 0;
}
```

2. Class untuk mendapatkan cofactor

```
void getCofactor(vector<vector<int>> &a, vector<vector<int>> &temp, int p, int q, int n)
{
    int i = 0, j = 0;
    for (int row = 0; row < n; row++)
    {
        for (int col = 0; col < n; col++)
        {
            if (row != p && col != q)
            {
                temp[i][j++] = a[row][col];
                if (j == n - 1)
                {
                    j = 0;
                    i++;
                }
            }
        }
    }
}
```

3. Class untuk mencari determinant

```
int determinant(vector<vector<int>> &a, int n, int N)
{
    int D = 0;
    if (n == 1)
        return a[0][0];
    vector<vector<int>> temp(N, vector<int>(N));
    int sign = 1;
    for (int f = 0; f < n; f++)
    {
        getCofactor(a, temp, 0, f, n);
        D += sign * a[0][f] * determinant(temp, n - 1, N);
        sign = -sign;
    }
    return D;
}
```

4. Class untuk mencari adjoint

```
void adjoint(vector<vector<int>> &a, vector<vector<int>> &adj, int N)
{
    if (N == 1)
    {
        adj[0][0] = 1;
        return;
    }
    int sign = 1;
    vector<vector<int>> temp(N, vector<int>(N));
    for (int i = 0; i < N; i++)
    {
        for (int j = 0; j < N; j++)
        {
            getCofactor(a, temp, i, j, N);
            sign = ((i + j) % 2 == 0) ? 1 : -1;
            adj[j][i] = (sign) * (determinant(temp, N - 1, N));
        }
    }
}
```

5. Class untuk mencari inverse matriks

```
bool inverse(vector<vector<int>> &a, vector<vector<int>> &inv, int N)
{
    int det = determinant(a, N, N);
    if (det == 0)
    {
        cout << "Inverse does not exist";
        return false;
    }
    int invDet = modInverse(det, 26);
//    cout << det % 26 << ' ' << invDet << '\n';
    vector<vector<int>> adj(N, vector<int>(N));
    adjoint(a, adj, N);
    for (int i = 0; i < N; i++)
        for (int j = 0; j < N; j++)
            inv[i][j] = (adj[i][j] * invDet) % 26;
    return true;
}
```

Mencari Enkripsi

Screenshot program

```
==== Program Hill Cipher ====

Input (Huruf Kapital): GOPHER
Ukuran matriks kunci : 2
Matriks kunci
Matriks element (0,0) : 7
Matriks element (0,1) : 6
Matriks element (1,0) : 2
Matriks element (1,1) : 5

Output :
Hasil Enskripsi : WERNAP
Hasil Deskripsi : GOPHER
Apakah ingin diulang ? (Y/N)
```

penjelasan

1. Pertama inputkan kata yang ingin dienkripsi / plaintext. Kemudian inputkan ordo matriksnya beserta elemennya.

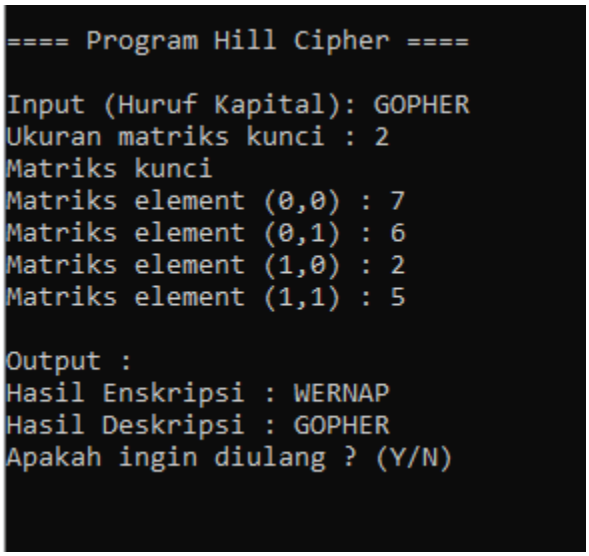
```
cout << "\n==== Program Hill Cipher ====\n\n" << "Input (Huruf Kapital): ";
cin >> s;
cout << "Ukuran matriks kunci : ";
cin >> n;
cout << "Matriks kunci\n";
vector<vector<int>> a(n, vector<int>(n));
vector<vector<int>> adj(n, vector<int>(n));
vector<vector<int>> inv(n, vector<int>(n));
for (i = 0; i < n; i++){
    for (j = 0; j < n; j++){
        cout<<"Matriks element ("<<i<<","<<j<<") : ";
        cin >> a[i][j];
    }
}
```

2. Inisiasi nilai k=0 untuk di looping hingga sejumlah panjang plaintext. Kemudian didalam loopingan k terdapat looping i sebanyak ordo matriks dan menginisiasi nilai sum=0 dan nilai temp=k. Didalam looping i terdapat juga looping j sebanyak ordo matriks dan terdapat perhitungan nilai sum (Matriks Plaintext * Matriks Kunci) dimana matriks a[i][j] mod 26 (agar menjadi angka) kemudian dikalikan dengan s[temp++] yang dikurang A kemudian di mod 26 sebanyak 2 kali (agar menjadi angka). Setelah didapat nilai sum akhir, ditambahkan ke variabel ans yang akan menyimpan hasil sum. Berulang terus hingga looping k,i,j berakhir dan didapat hasil enkripsinya.

```
k = 0;
string ans = "";
while (k < s.size()){
    for (i = 0; i < n; i++){
        int sum = 0;
        int temp = k;
        for (j = 0; j < n; j++){
            sum += (a[i][j] % 26 * (s[temp++] - 'A') % 26) % 26;
            sum = sum % 26;
        }
        ans += (sum + 'A');
    }
    k += n;
}
```

Mencari Dekripsi

Screenshot



Penjelasan

1. Pertama lakukan inverse key matrix

```
if (inverse(a, inv, n))
{
    cout << '\n';
};
```

2. Lalu setelah itu fungsi dibawah akan dijalankan

```
//fungsi untuk mencari dekripsi dari hasil ekripsi diatas
k = 0;
string deskripsi;
while (k < ans.size())
{
    for (i = 0; i < n; i++)
    {
        int sum = 0;
        int temp = k;
        for (j = 0; j < n; j++)
        {
            sum += ((inv[i][j] + 26) % 26 * (ans[temp++] - 'A') % 26) % 26;
            sum = sum % 26;
        }
        deskripsi += (sum + 'A');
    }
    k += n;
}
```

variabel sum akan menyimpan hasil dari rumus P=K^-1 . C mod 26

3. Lalu akan diouput melalui command dibawah

```
//output
cout << "Output : "
     << "\nHasil Enskripsi : " << ans << "\nHasil Deskripsi : " << deskripsi;

string pilih;
cout << "\nApakah ingin diulang ? (Y/N) ";
cin >> pilih;
```

Mencari Key

Screenshot hasil

```
Silahkan masukan plaintext (Huruf Kapital): FRIDAY
Silahkan masukan ciphertext (Huruf Kapital): PQCFKU

Plaintext
5 8
17 3

Ciphertext
15 2
16 5


Inverse Plaintext
9 2
1 15

Key sebelum di mod 26
137 60
149 107

Key setelah di mod 26
7 8
19 3
```

Penjelasan program

1. Input plaintext dan ciphertext
   Pertama deklarasi variabel array untuk menyimpan data matriks, vektor pl menyimpan matriks plaintext, vektor inv akan menyimpan matriks inverse dari plaintext

```
int n= 2;
vector<vector<int>> pl(n, vector<int>(n));
vector<vector<int>> adj(n, vector<int>(n));
vector<vector<int>> inv(n, vector<int>(n));

string p,c;
```

   ci[2][2] akan menyimpan matriks ciphertext
   Helper disini berperan sebagai counter untuk membantu memasukan data string kedalam matriks

```
cout<<"Silahkan masukan plaintext (Huruf Kapital): "; cin>>p;
cout<<"Silahkan masukan ciphertext (Huruf Kapital): "; cin>>c;

int helper=0;
int ci[2][2], pli[2][2], output[2][2];
for(int i= 0 ; i < 2; i++){
    for(int j= 0 ; j < 2; j++){

        ci[j][i]= c[helper]-'A';
        pl[j][i]= p[helper]-'A';
        helper++;
    }

}
helper=0;
```

2. Menampilkan plaintext dan juga ciphertext

```
//plaintext
cout<<"\nPlaintext"<<endl;
for(int i= 0 ; i < n; i++){
    for(int j= 0 ; j < n; j++){
        cout<<pl[i][j]<<" ";
    }
    cout<<endl;
}

//ciphertext
cout<<"\nCiphertext"<<endl;
for(int i= 0 ; i < n; i++){
    for(int j= 0 ; j < n; j++){
        cout<<ci[i][j]<<" ";
    }
    cout<<endl;
}
```

3. Melakukan inverse plaintext dan menampilkannya

```
//inverse plaintext
if(inverse(pl,inv,n)){
    cout<<"\n";
}
cout<<"\nInverse Plaintext"<<endl;
for(int i= 0 ; i < n; i++){
    for(int j= 0 ; j < n; j++){
        inv[i][j]= (inv[i][j]+26) % 26;
        cout<<inv[i][j]<<" ";


    }
    cout<<endl;
}
```

4. Melakukan perkalian ciphertext dengan inverse plaintext sesuai rumus K=C.P^-1 dan menampilkan hasil key dari hill cipher

```
for(int i= 0 ; i < 2; i++){
    for(int j= 0 ; j < 2; j++){
        output[i][j] = 0;
    }
}

//perkalian ciphertext dengan inverse plaintext
for(int i= 0 ; i < 2; i++){
    for(int j= 0 ; j < 2; j++){
        for(int k= 0 ; k < 2; k++){
            output[i][j]+=ci[i][k] * inv[k][j];
        }
    }
}

//menampilkan key sebelum mod 26
cout<<"\nKey sebelum di mod 26"<<endl;
for(int i= 0 ; i < 2; i++){
    for(int j= 0 ; j < 2; j++){
        cout<<output[i][j]<<" ";
    }
    cout<<endl;
}

//menampilkan key setelah di mod 26
cout<<"\nKey setelah di mod 26"<<endl;
for(int i= 0 ; i < 2; i++){
    for(int j= 0 ; j < 2; j++){
        cout<<output[i][j]%26<<" ";
    }
    cout<<endl;
}
```

5. Perulangan untuk mengulangi program atau mengakhiri program

```
string pilih;
cout << "\n\nApakah ingin diulang ? (Y/N) ";
cin >> pilih;
if (pilih == "Y" || pilih == "y")
{
    goto mulai;
}
else
{
    return 0;
}
```