



TUGAS AKHIR - IF184802

# DESAIN DAN ANALISIS ALGORITMA PENYANDIAN MESIN ENIGMA M3 PADA STUDI KASUS PERMASALAHAN SPOJ ENIGMA MACHINE

DANIEL LUMBANTOBING  
0511164000042

Dosen Pembimbing I:  
Rully Soelaiman, S.Kom., M.Kom.

Dosen Pembimbing II:  
Abdul Munif, S.Kom., M.Sc.

DEPARTEMEN TEKNIK INFORMATIKA  
Fakultas Teknologi Elektro dan Informatika Cerdas  
Institut Teknologi Sepuluh Nopember  
Surabaya 2020

***(Halaman ini sengaja dikosongkan)***



**TUGAS AKHIR - IF184802**

# **DESAIN DAN ANALISIS ALGORITMA PENYANDIAN MESIN ENIGMA M3 PADA STUDI KASUS PERMASALAHAN SPOJ ENIGMA MACHINE**

**DANIEL LUMBANTOBING**  
0511164000042

**Dosen Pembimbing I:**  
Rully Soelaiman, S.Kom., M.Kom.

**Dosen Pembimbing II:**  
Abdul Munif, S.Kom., M.Sc.

**DEPARTEMEN TEKNIK INFORMATIKA**  
Fakultas Teknologi Elektro dan Informatika Cerdas  
Institut Teknologi Sepuluh Nopember  
Surabaya 2020

*(Halaman ini sengaja dikosongkan)*



**UNDERGRADUATE THESIS - IF184802**

**DESIGN AND ANALYSIS OF ENIGMA M3  
ENCODING MACHINE ALGORITHM IN CASE  
STUDY SPOJ ENIGMA MACHINE PROBLEM**

**DANIEL LUMBANTOBING  
0511164000042**

**Supervisor I:  
Rully Soelaiman, S.Kom., M.Kom.**

**Supervisor II:  
Abdul Munif, S.Kom., M.Sc.**

**DEPARTMENT OF INFORMATICS  
Faculty of Intelligent Electrical and Informatics Technology  
Institut Teknologi Sepuluh Nopember  
Surabaya 2020**

*(Halaman ini sengaja dikosongkan)*

## LEMBAR PENGESAHAN

# DESAIN DAN ANALISIS ALGORITMA PENYANDIAN MESIN ENIGMA M3 PADA STUDI KASUS PERMASALAHAN SPOJ ENIGMA MACHINE

## TUGAS AKHIR

Diajukan Untuk Memenuhi Salah Satu Syarat  
Memperoleh Gelar Sarjana Komputer  
pada  
Bidang Studi Algoritma Pemrograman  
Program Studi S-1 Teknik Informatika  
Departemen Teknik Informatika  
Fakultas Teknologi Elektro dan Informatika Cerdas  
Institut Teknologi Sepuluh Nopember

Oleh:

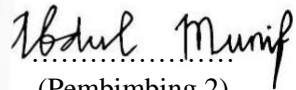
Daniel Lumbantobing  
NRP: 051116 40000 042

Disetujui oleh Dosen Pembimbing Tugas Akhir:

Rully Soelaiman, S.Kom., M.Kom.  
NIP. 19700213 199402 1 001

  
.....  
(Pembimbing 1)

Abdul Munif, S.Kom., M.Sc.  
NIP. 19860823 201504 1 004

  
.....  
(Pembimbing 2)



**SURABAYA 2020**

*(Halaman ini sengaja dikosongkan)*



**DESAIN DAN ANALISIS ALGORITMA  
PENYANDIAN MESIN ENIGMA M3 PADA STUDI  
KASUS PERMASALAHAN SPOJ ENIGMA  
MACHINE**

**Nama Mahasiswa** : Daniel Lumbantobing  
**NRP** : 051116 40000 042  
**Departemen** : Teknik Informatika, Fakultas  
Teknologi Elektro dan Informatika  
Cerdas, ITS  
**Dosen Pembimbing 1** : Rully Soelaiman, S.Kom., M.Kom.  
**Dosen Pembimbing 2** : Abdul Munif, S.Kom., M.Sc.

**Abstrak**

*Permasalahan SPOJ Enigma Machine merupakan permasalahan mengenai enkripsi yang dilakukan oleh mesin Enigma M3. Mesin Enigma M3 memiliki pengaturan yang dilakukan sebelum memulai proses enkripsi, yaitu: tiga rotor yang digunakan dari lima, posisi awal rotor, pengaturan cincin dan pasangan huruf yang ditukar pada plugboard.*

*Dalam menyelesaikan permasalahan ini diperlukan pemahaman mengenai bagaimana mesin Enigma M3 bekerja, terkhususnya proses enkripsi pada bagian rotor yang posisinya terus berubah dan juga plugboard yang menukar huruf masukan dan huruf keluaran pada mesin.*

*Dari serangkaian percobaan yang telah dilakukan, diperoleh kesimpulan bahwa algoritma yang dirancang dan diimplementasikan dapat menyelesaikan enkripsi pada mesin Enigma M3 dengan waktu 0,0 detik dan penggunaan memori sebesar 4,61 MB sedangkan pada komputer lokal waktunya adalah 0,029 detik.*

***Kata Kunci: Enigma M3, Enkripsi.***

*(Halaman ini sengaja dikosongkan)*

**DESIGN AND ANALYSIS OF ENIGMA M3  
ENCODING MACHINE ALGORITHM IN CASE STUDY  
SPOJ ENIGMA MACHINE PROBLEM**

**Name** : Daniel Lumbantobing  
**NRP** : 051116 40000 042  
**Department** : Informatics, Faculty of Intelligent  
Electrical and Informatics Technology, ITS  
**Supervisor 1** : Rully Soelaiman, S.Kom., M.Kom.  
**Supervisor 2** : Abdul Munif, S.Kom., M.Sc.

**Abstarct**

*Enigma Machine problem in SPOJ is a problem about Enigma M3 machine encryption. Enigma M3 machine has several setting that need to be set before starting encryption process i.e. three of five used rotor, starting position of rotor, ring setting and swapped letter pairs on plugboard.*

*Understanding how Enigma M3 machine work is fundamental for solving this problem, especially encryption process on rotor which is gradually change and plugboard for switching letter.*

*From series of experiment that have been done, it can be concluded that designed and implemented algorithm could solve the encryption of Enigma M3 machine with time 0.0 seconds and 4.61 MB memory usage while time with local computer is 0.029 seconds.*

**Keyword: Encryption, Enigma M3.**

*(Halaman ini sengaja dikosongkan)*

## **KATA PENGANTAR**

Segala puji dan syukur bagi Tuhan Yesus Kristus yang telah melimpahkan segala berkat dan rahmatnya sehingga penulis dapat menyelesaikan Tugas Akhir ini yang berjudul:

### **DESAIN DAN ANALISIS ALGORITMA PENYANDIAN MESIN ENIGMA M3 PADA STUDI KASUS PERMASALAHAN SPOJ ENIGMA MACHINE**

Dengan selesainya Tugas Akhir ini diharapkan apa yang telah dikerjakan penulis dapat memberikan kontribusi bagi perkembangan ilmu pengetahuan terutama di bidang teknologi informasi serta bagi diri penulis sendiri selaku peneliti.

Penulis mengucapkan terima kasih kepada semua pihak yang telah memberikan dukungan, baik secara langsung maupun tidak langsung, selama penulis mengerjakan Tugas Akhir maupun selama menempuh masa studi antara lain:

1. Bapak Rully Soelaiman, S.Kom., M.Kom. selaku dosen pembimbing yang telah banyak meluangkan waktu untuk memberikan ilmu, nasihat, motivasi, bimbingan dan didikan kepada penulis dengan sabar selama perkuliahan maupun pengerjaan Tugas Akhir ini.
2. Abdul Munif, S.Kom., M.Sc. selaku dosen pembimbing yang telah memberikan masukan dan bimbingan kepada penulis selama pengerjaan Tugas Akhir ini.
3. Bapak (Togar Lumbantobing, S.E.), ibu (Melva Damaik, S.H.), dan kedua kakak (Andre P. M. Lumbantobing S.E. dan Jeremia Lumbantobing, S.H.) penulis yang selalu memberikan dukungan, perhatian, dan kasih sayang bagi penulis yang menjadi semangat selama perkuliahan maupun pengerjaan Tugas Akhir ini.
4. Seluruh tenaga pengajar dan karyawan Departemen Teknik Informatika ITS yang telah memberikan tenaga

dan waktunya demi kelancaran proses belajar mengajar penulis selama perkuliahan.

5. Teman – teman Pembinaan PKMBK PMK ITS Sabhi (Henokh, Mahanaim, Gary, Titius, Verlin, Vayo, Magda, Mayshel, Kezia) dan juga kedua koor Pembinaan (Ronald dan Irene) yang selalu memberikan dukungan dan menemani kehidupan perkuliahan penulis.
6. Seluruh teman – teman PMK terkhususnya Icon, Cintaka, dan Irene yang terus memberikan dukungan dan menemani kehidupan perkuliahan penulis.
7. Fitri Sri Lestari Sinaga yang telah menemani penulis dan membantu penulis dalam pengerjaan Tugas Akhir ini.
8. Yoshima Syach Putri yang telah membantu penulis untuk mengoreksi penulisan buku ini.
9. Teman-teman angkatan 2016 Teknik Informatika ITS yang telah menemani perjuangan penulis dalam menempuh studi selama empat tahun.
10. Serta pihak-pihak lain yang tidak dapat disebutkan disini yang telah banyak membantu penulis dalam penyusunan Tugas Akhir ini.

Penulis menyadari masih ada kekurangan pada Tugas Akhir ini sehingga penulis mengharapkan kritik dan saran yang membangun untuk pembelajaran dan perbaikan supaya Tugas Akhir ini menjadi lebih baik. Semoga melalui Tugas Akhir ini penulis dapat memberikan manfaat untuk pembaca.

Surabaya, Juni 2020

Daniel Lumbantobing

# DAFTAR ISI

<b>LEMBAR PENGESAHAN.....</b>	<b>vii</b>
<b>Abstrak .....</b>	<b>ix</b>
<b>Abstarct .....</b>	<b>xi</b>
<b>KATA PENGANTAR .....</b>	<b>xiii</b>
<b>DAFTAR ISI.....</b>	<b>xv</b>
<b>BAB I PENDAHULUAN .....</b>	<b>1</b>
1.1    Latar Belakang .....	1
1.2    Rumusan Permasalahan.....	1
1.3    Batasan permasalahan .....	2
1.4    Tujuan Pembuatan Tugas Akhir .....	2
1.5    Manfaat Tugas Akhir.....	2
1.6    Metodologi .....	3
1.6.1    Penyusunan Proposal Tugas Akhir.....	3
1.6.2    Studi Literatur.....	3
1.6.3    Desain dan Analisis .....	3
1.6.4    Implementasi Perangkat Lunak .....	3
1.6.5    Uji Coba Kebenaran .....	4
1.6.6    Penyusunan Buku Tugas Akhir .....	4
1.7    Sistematika Penulisan.....	4
<b>BAB II DASAR TEORI.....</b>	<b>7</b>
2.1    Deskripsi Umum Permasalahan .....	7
2.2    Deskripsi Umum Teori .....	8
2.2.1    Enkripsi dan Dekripsi .....	8
2.2.2 <i>Polyalphabetic Cipher</i> .....	9
2.2.3 <i>Caeser Cipher</i> .....	10
2.2.4 <i>Enigma Cipher</i> .....	10
<b>BAB III DESAIN DAN ANALISIS .....</b>	<b>17</b>
3.1    Analisis Observasi Permasalahan.....	17

3.1.1	Posisi Rotor dan Pengaturan Cincin pada Proses Enkripsi .....	17
3.1.2	<i>Double-stepping</i> pada Rotor .....	22
3.2	Deskripsi Umum Program .....	23
3.3	Desain Algoritma.....	25
3.3.1	Desain Tahap <i>init_plugboard</i> .....	26
3.3.2	Desain Tahap <i>swap_char</i> .....	26
3.3.3	Desain Tahap <i>enkripsi</i> .....	27
3.3.4	Desain Tahap <i>rotor_step</i> .....	27
<b>BAB IV</b>	<b>IMPLEMENTASI.....</b>	<b>29</b>
4.1	Lingkungan Implementasi .....	29
4.2	Penggunaan <i>Library</i> dan Inisialisasi Variabel .....	29
4.3	Implementasi Tahap <i>init_plugboard</i> .....	30
4.4	Implementasi Tahap <i>swap_char</i> .....	31
4.5	Implementasi Tahap <i>enkripsi</i> .....	31
4.6	Implementasi Tahap <i>rotor_step</i> .....	32
4.7	Implementasi Tahap <i>main</i> .....	32
<b>BAB V</b>	<b>UJI COBA DAN EVALUASI.....</b>	<b>33</b>
5.1	Lingkungan Uji Coba .....	33
5.2	Uji Coba Kebenaran .....	33
5.2.1	Uji Coba Kebenaran Lokal .....	33
5.2.2	Uji Coba Kebenaran pada Situs Daring SPOJ .....	44
<b>BAB VI</b>	<b>KESIMPULAN DAN SARAN .....</b>	<b>49</b>
6.1	Kesimpulan.....	49
6.2	Saran .....	49
<b>DAFTAR PUSTAKA</b>	<b>.....</b>	<b>51</b>
<b>BIODATA PENULIS</b>	<b>.....</b>	<b>53</b>



## DAFTAR GAMBAR

Gambar 2.1 Contoh <i>Test Case</i> pada Permasalahan Enigma <i>Machine</i> .....	7
Gambar 2.2 Alur Penyandian Mesin Enigma.....	11
Gambar 2.3 Alur Penyandian Mesin Enigma M3 .....	14
Gambar 3.1 <i>Pseudocode</i> Main .....	26
Gambar 3.2 <i>Pseudocode</i> Tahap <i>init_plugboard</i> .....	26
Gambar 3.3 <i>Pseudocode</i> Tahap <i>swap_char</i> .....	27
Gambar 3.4 <i>Pseudocode</i> Tahap <i>enkripsi</i> .....	27
Gambar 3.5 <i>Pseudocode</i> Tahap <i>rotor_step</i> .....	28
Gambar 5.1 Kasus Uji .....	34
Gambar 5.2 Hasil Uji Coba pada Komputer Lokal .....	43
Gambar 5.3 Hasil Uji Kebenaran pada situs SPOJ.....	44
Gambar 5.4 Hasil Uji Kebenaran pada Situs SPOJ sebanyak 10 kali.....	45
Gambar 5.5 Grafik Waktu Uji Coba 10 Kali pada SPOJ dan Komputer Lokal .....	46
Gambar 5.6 Grafik Memori Uji Coba 10 kali pada SPOJ .....	46

*(Halaman ini sengaja dikosongkan)*

## DAFTAR TABEL

Tabel 2.1 Contoh <i>Polyalphabetic Cipher</i> .....	9
Tabel 2.2 Contoh <i>Caesar Cipher</i> .....	10
Tabel 2.3 Enkripsi pada Setiap Rotor dan Reflektor Enigma M3 .....	12
Tabel 2.4 Perubahan Enkripsi dengan Nilai Pengaturan Cincin .	12
Tabel 2.5 Posisi <i>Notch</i> pada Rotor .....	13
Tabel 2.6 Posisi Awal Rotor Relatif Terhadap <i>Input</i> .....	15
Tabel 3.1 Perubahan Posisi Rotor dengan Pengaturan Cincin AAA .....	18
Tabel 3.2 Perubahan Posisi Rotor dengan Pengaturan Cincin ABC.....	20
Tabel 3.3 Keadaan Rotor I dan II dengan Pengaturannya.....	21
Tabel 3.4 <i>Double-stepping</i> pada Mesin Enigma M3.....	23
Tabel 3.5 Daftar Variabel.....	24
Tabel 4.1 Spesifikasi Lingkungan Implementasi .....	29
Tabel 5.1 Spesifikasi Lingkungan Uji Coba.....	33
Tabel 5.2 Spesifikasi <i>R</i> .....	35
Tabel 5.3 Pasangan Huruf <i>Plugboard</i> pada Kasus Uji-1 .....	36
Tabel 5.4 Keadaan Awal Mesin Enigma M3 pada Kasus Uji-1..	36
Tabel 5.5 Pasangan Huruf <i>Plugboard</i> pada Kasus Uji-2 .....	38
Tabel 5.6 Keadaan Awal Mesin Enigma M3 pada Kasus Uji-2..	38
Tabel 5.7 Pasangan Huruf <i>Plugboard</i> pada Kasus Uji-3 .....	40
Tabel 5.8 Keadaan Awal Mesin Enigma M3 pada Kasus Uji-3..	41
Tabel 5.9 Hasil Uji Coba 10 Kali pada Komputer Lokal .....	43
Tabel 5.10 Hasil Uji Coba 10 Kali pada SPOJ.....	45

*(Halaman ini sengaja dikosongkan)*

## DAFTAR KODE SUMBER

Kode Sumber 4.1 Implementasi <i>Library</i> .....	29
Kode Sumber 4.2 Inialisasi Variabel .....	30
Kode Sumber 4.3 Tahap <i>init_plugboard</i> .....	30
Kode Sumber 4.4 Tahap <i>swap_char</i> .....	31
Kode Sumber 4.5 Tahap <i>enkripsi</i> .....	31
Kode Sumber 4.6 Tahap <i>rotor_step</i> .....	32
Kode Sumber 4.7 Fungsi <i>main</i> .....	32

*(Halaman ini sengaja dikosongkan)*

# **BAB I**

## **PENDAHULUAN**

Pada bab ini akan dipaparkan mengenai garis besar Tugas Akhir yang meliputi latar belakang, tujuan, rumusan masalah, batasan permasalahan, metodologi pembuatan Tugas Akhir, dan sistematika penulisan.

### **1.1 Latar Belakang**

Permasalahan SPOJ bertipe tantangan berjudul “Enigma Machine” mengangkat pencarian solusi dari hasil penyandian mesin Enigma M3 yang digunakan pada perang dunia kedua.

Dengan kunci sandi yang berubah secara linier untuk menyandikan tiap huruf dan susunan huruf yang diacak dari awal mesin dijalankan, diperlukan tahapan – tahapan penyandian yang jelas untuk membentuk algoritma penyelesaian permasalahan tersebut.

Hasil tugas akhir ini diharapkan dapat menentukan algoritma yang tepat sehingga dapat menyelesaikan permasalahan di atas dengan optimal dan diharapkan dapat memberikan kontribusi pada perkembangan ilmu pengetahuan dan teknologi informasi.

### **1.2 Rumusan Permasalahan**

Rumusan masalah yang diangkat dalam Tugas Akhir ini adalah sebagai berikut:

1. Bagaimana desain algoritma yang sesuai dan efisien untuk menyandikan teks berdasarkan pengaturan mesin Enigma M3?
2. Bagaimana uji coba untuk mengetahui kebenaran dan kinerja dari implementasi algoritma yang sudah didesain berdasarkan mesin Enigma M3?

### 1.3 Batasan Permasalahan

Permasalahan yang dibahas pada Tugas Akhir ini memiliki beberapa batasan, yaitu sebagai berikut:

1. Implementasi algoritma menggunakan bahasa pemrograman C.
2. Pembuktian kebenaran didasarkan pada hasil *submission* pada sistem penilaian daring SPOJ.

Batasan pada SPOJ:

1. Batas maksimum jumlah kueri  $T$  adalah 100.
2.  $U$  sebagai tiga digit angka yang dimasukkan berupa bilangan bulat dimana tiap angka memiliki rentang satu sampai lima.
3.  $X$  dan  $P$  sebagai huruf yang dimasukkan masing - masing satu kata yang terdiri dari tiga huruf, dimana masing – masing huruf memiliki rentang A – Z.
4.  $H$  sebagai pasangan huruf dalam rentang A - Z yang unik dan maksimal 13 pasang.
5. Keluaran program harus dalam format berlipat lima.
6. Batas maksimum penggunaan memori adalah 1536 MB
7. Batas waktu adalah 10 detik.
8. Batas ukuran program adalah 50 KB.

### 1.4 Tujuan Pembuatan Tugas Akhir

Tujuan dalam pembuatan tugas akhir ini adalah sebagai berikut:

1. Melakukan analisis untuk mencari desain algoritma yang sesuai dan efisien untuk menyelesaikan permasalahan SPOJ *Enigma Machine*.
2. Mengevaluasi hasil dan kinerja dari implementasi.

### 1.5 Manfaat Tugas Akhir

Manfaat dari pembuatan Tugas Akhir ini adalah sebagai berikut:



1. Mampu memberikan pemahaman dan penjelasan mencari desain algoritma yang efisien untuk penyelesaian kasus SPOJ Enigma *Machine*.
2. Memberikan hasil uji coba dari implementasi desain algoritma yang telah dilakukan.

## **1.6 Metodologi**

Langkah-langkah yang ditempuh dalam pengerjaan Tugas Akhir ini yaitu:

### **1.6.1 Penyusunan Proposal Tugas Akhir**

Pada tahap ini dilakukan penyusunan proposal Tugas Akhir yang berisi permasalahan pada permasalahan SPOJ Enigma *Machine* serta gagasan solusi yang akan dibahas pada Tugas Akhir ini.

### **1.6.2 Studi Literatur**

Tahap kedua adalah mencari informasi dan studi literatur yang relevan untuk dijadikan referensi dalam melakukan pengerjaan tugas akhir. Informasi dan studi literatur ini didapatkan dari buku, internet, dan materi - materi kuliah yang berhubungan dengan metode yang akan digunakan.

### **1.6.3 Desain dan Analisis**

Pada tahap ini dilakukan desain rancangan algoritma yang digunakan dalam solusi untuk pemecahan masalah SPOJ Enigma *Machine*.

### **1.6.4 Implementasi Perangkat Lunak**

Pada tahap ini dilakukan implementasi dari rancangan desain ke dalam bentuk program dalam bahasa pemrograman C.

### **1.6.5 Uji Coba Kebenaran**

Pada tahap ini dilakukan uji coba kebenaran implementasi yang dilakukan pada sistem penilaian daring SPOJ.

### **1.6.6 Penyusunan Buku Tugas Akhir**

Pada tahap ini dilakukan penyusunan laporan yang menjelaskan dasar teori dan metode yang digunakan dalam Tugas Akhir ini serta hasil dari implementasi aplikasi perangkat lunak yang telah dibuat.

## **1.7 Sistematika Penulisan**

Buku Tugas Akhir ini merupakan laporan secara lengkap mengenai tugas akhir yang telah dikerjakan baik dari sisi teori, rancangan, maupun implementasi sehingga memudahkan bagi pembaca dan juga pihak yang ingin mengembangkan lebih lanjut. Sistematika penulisan buku Tugas Akhir secara garis besar dapat dilihat seperti di bawah ini.

### **Bab I Pendahuluan**

Bab ini berisi penjelasan latar belakang, rumusan masalah, batasan masalah dan tujuan pembuatan Tugas Akhir. Selain itu, metodologi pengerjaan dan sistematika penulisan laporan Tugas Akhir juga terdapat di dalamnya.

### **Bab II Dasar Teori**

Bab ini berisi penjelasan secara detail mengenai dasar-dasar penunjang dan teori-teori yang digunakan untuk mendukung pembuatan Tugas Akhir ini.

### **Bab III Desain dan Analisis**

Bab ini berisi desain algoritma dan struktur data yang digunakan dalam penyelesaian permasalahan.

## **Bab IV Implementasi**

Bab ini berisi penjelasan implementasi dalam bahasa pemrograman C berdasarkan desain algoritma yang telah dilakukan pada tahap desain.

## **Bab V Uji Coba dan Evaluasi**

Bab ini berisi uji coba kebenaran implementasi yang dilakukan pada sistem penilaian daring SPOJ.

## **Bab VI Kesimpulan dan Saran**

Bab ini merupakan bab terakhir yang menyampaikan kesimpulan dari hasil uji coba yang dilakukan dan saran untuk pengembangan perangkat lunak ke depannya.

*(Halaman ini sengaja dikosongkan)*

## BAB II DASAR TEORI

Pada bab ini akan dijelaskan mengenai dasar teori yang digunakan untuk menyelesaikan permasalahan pada kasus SPOJ Enigma *Machine*.

### 2.1 Deskripsi Umum Permasalahan

Pada permasalahan SPOJ Enigma *Machine*, diberikan *input* yang berisi *test case*  $T$ . Baris pertama pada masing - masing *test case* berisi tiga rangkaian kata di mana tiap kata memiliki tiga buah huruf atau angka. Kata pertama adalah  $U$  yang berisi tiga angka unik pada rentang 1-5 yang digunakan sebagai susunan rotor mesin enigma. Urutan penyusunan rotor sesuai dengan urutan penyusunan angka yang dimasukkan. Kata kedua adalah  $X$  yang berisi tiga huruf dan masing – masing huruf memiliki rentang A-Z.  $X$  digunakan sebagai pengaturan cincin tiap rotor secara berurut dari kiri ke kanan. Kata ketiga adalah  $P$  yang memiliki format sama dengan  $X$ , tetapi  $P$  digunakan sebagai kondisi awal rotor sebelum memulai penyandian. Pada baris berikutnya berisi  $B$  yang berupa pasangan huruf unik pada rentang A-Z untuk tiap hurufnya. Di mana  $B$  maksimal memiliki 13 pasang huruf. Baris berikutnya berisi masukan kata yang akan disandikan berdasarkan seluruh pengaturan yang telah dimasukkan sebelumnya [1].

```
123 JAN DER  
SP OJ RU LZ  
THISX ISXAN XEXAM PLEXI NPUT
```

**Gambar 2.1** Contoh *Test Case* pada Permasalahan Enigma *Machine*

Sebagai contoh, diberikan pengaturan mesin enigma seperti pada Gambar 2.1, dua baris pertama merupakan pengaturan

mesin Enigma M3, dan baris terakhir adalah pesan yang harus disandikan berdasarkan pengaturannya.

## 2.2 Deskripsi Umum Teori

Pada subbab ini, akan dijelaskan berbagai landasan teori secara umum yang digunakan untuk melakukan pendekatan terhadap penyelesaian permasalahan.

### 2.2.1 Enkripsi dan Dekripsi

Enkripsi merupakan sebuah metode untuk mengubah informasi yang memiliki makna menjadi sebuah kode rahasia yang menyembunyikan makna sesungguhnya informasi tersebut. Pesan yang akan dienkripsi menggunakan istilah *plaintext* sedangkan pesan yang sudah melalui tahap enkripsi disebut sebagai *ciphertext* [2].

Dekripsi sendiri merupakan kebalikan dari enkripsi, di mana dekripsi merupakan suatu metode untuk mengubah informasi yang sudah dikodekan dan tidak memiliki makna atau disebut juga *ciphertext*, menjadi informasi yang memiliki makna atau menjadi *plaintext*.

Dalam melakukan enkripsi diperlukan tahapan – tahapan yang jelas dan terstruktur sehingga dapat dimengerti bagaimana proses pengubahan *plaintext* menjadi *ciphertext* ataupun sebaliknya. Oleh sebab itu, diperlukan algoritma pasti untuk melakukan enkripsi maupun dekripsi. *Key* merupakan variabel pada enkripsi dan dekripsi yang menyebabkan hasil dari algoritma enkripsi atau dekripsi menghasilkan keluaran yang sesuai.

Enkripsi memiliki dua jenis metode yaitu simetrik dan asimetrik. Kedua metode ini dipisahkan berdasarkan penggunaan *key* pada prosesnya. Dikatakan simetrik jika *key* untuk melakukan enkripsi dan dekripsi adalah sama, dan dikatakan asimetrik jika *key* yang digunakan pada proses enkripsi dan dekripsi berbeda [2].

$$c = F(m, k) = F_k(m) \quad (2.1)$$

$$m = F^{-1}(c, k) = F_k^{-1}(c) \quad (2.2)$$

Pada Persamaan 2.1 dan Persamaan 2.2,  $c$  adalah *ciphertext*,  $F$  adalah fungsi enkripsi,  $F^{-1}$  adalah fungsi dekripsi dan  $m$  merupakan *plaintext* sedangkan  $k$  merupakan *key*. Dari kedua persamaan tersebut, maka enkripsi simetrik harus memenuhi Persamaan 2.3 berikut.

$$F^{-1}(F(m, k), k) = m \quad (2.3)$$

### 2.2.2 Polyalphabetic Cipher

*Polyalphabetic cipher* adalah sandi yang didasarkan pada pertukaran huruf yang sama pada *plaintext* tapi menghasilkan huruf yang berbeda pada *ciphertext*. Hal ini dapat diartikan bahwa satu huruf dapat disandikan menjadi beberapa kemungkinan huruf [3].

**Tabel 2.1** Contoh *Polyalphabetic Cipher*

<i>Plaintext</i>	T	E	K	N	I	K
<i>Key</i>	A	C	A	C	A	C
<i>Ciphertext</i>	U	H	L	Q	J	N

Tabel 2.1 merupakan contoh dari *polyalphabetic cipher* di mana pada enkripsi menggunakan  $key = AC$  dan rumus yang digunakan adalah  $(plaintext + key) \text{ modulo } 26 = ciphertext$ . Di mana tiap huruf dimisalkan sebagai angka secara berurut, di mana A adalah 1, B adalah 2 dan seterusnya hingga Z adalah 26. Maka dari *plaintext* TEKNIK dan  $key = AC$  dihasilkan *ciphertext* UHLQJN. Terlihat pada *plaintext* bahwa huruf K muncul sebanyak dua kali. Namun, dienkripsi menjadi dua huruf berbeda yaitu L dan N.

### 2.2.3 Caesar Cipher

*Caesar cipher* adalah sebuah metode enkripsi *monoalphabetic cipher*, dimana setiap satu huruf pada *plaintext* dienkripsi hanya menjadi satu huruf pada *ciphertext*-nya. *Key* pada *caesar cipher* berupa angka yang berkisar dari 1 sampai 26. Di mana fungsi *key* hanya sebagai penambah untuk *plaintext* sehingga menghasilkan *ciphertext* [4].

**Tabel 2.2** Contoh Caesar Cipher

<i>Plaintext</i>	T	E	K	N	I	K
<i>Key</i>	5					
<i>Ciphertext</i>	Y	J	P	S	N	P

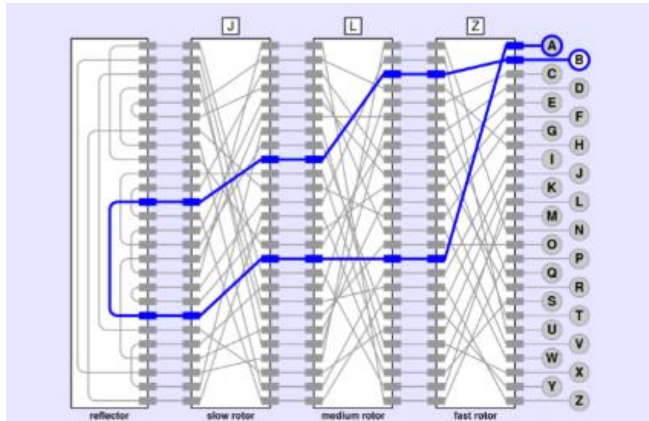
Pada Tabel 2.2 terlihat bahwa *ciphertext* hanya penambahan *key* yang hanya memiliki satu nilai terhadap *plaintext*. Untuk melakukan dekripsi hanya perlu mengurangi *ciphertext* dengan *key*, maka akan kembali kepada *plaintext*. Berbeda pada Tabel 2.1 di mana huruf K dienkripsi menjadi dua huruf berbeda, pada *caesar cipher* huruf K dienkripsi menjadi huruf yang sama yaitu P.

### 2.2.4 Enigma Cipher

*Enigma Cipher* merupakan metode enkripsi yang menggunakan rotor sebagai alat penyandiannya. *Enigma* juga merupakan salah satu jenis *polyalphabetic cipher*. Pada umumnya, *enigma* menggunakan tiga buah rotor untuk melakukan enkripsi dan ini menyebabkan setiap huruf pada *plaintext* akan dienkripsi sebanyak tujuh kali. Hal ini dapat dimungkinkan karena alur penyandian yang terjadi, di mana setelah melewati ketiga rotor, maka enkripsi akan dipantulkan menggunakan reflektor sehingga



enkripsi terjadi lagi pada ketiga rotor tetapi dengan arah yang terbalik [5, 6].



**Gambar 2.2** Alur Penyandian Mesin Enigma

Pada Gambar 2.2 huruf A merupakan huruf yang akan dienkripsi dan B adalah hasilnya yang didapatkan dari alur enkripsi. Susunan pada Gambar 2.2 dari kiri ke kanan adalah reflektor dan ketiga rotor yang digunakan untuk melakukan enkripsi. Tiap – tiap rotor menggunakan *monoalphabetic cipher* di mana tiap huruf pada rotor hanya akan dienkripsikan oleh satu huruf lainnya. Yang membuat enigma menjadi *polyalphabetic cipher* adalah pergerakan ketiga rotor yang terus menerus mengubah alur penyandian. Tabel 2.3 ini adalah spesifikasi enkripsi yang dilakukan tiap rotor dan reflektor pada mesin Enigma M3.

**Tabel 2.3** Enkripsi pada Setiap Rotor dan Reflektor Enigma M3

Rotor\Input	ABCDEFGHIJKLMNOPQRSTUVWXYZ
I	EKMFLGDQVZNTOWYHXUSPAIBRCJ
II	AJDKSIRUXBLHWTMCQGZNPYFVOE
III	BDFHJLCPRTXVZNYEIWGAKMUSQO
IV	ESOVZPJAYQUIRHXLNFTGKDCMWB
V	VZBRGITYUPSDNHLXAWMJQOFECK
Reflektor B	YRUHQSLDPXNGOKMIEBFZCWVJAT

Setiap rotor memiliki pengaturan cincin kecuali reflektor. Pengaturan cincin ini berfungsi untuk mengatur rangkaian enkripsi pada rotor tersebut dengan aturan yang pasti. Pada Tabel 2.3 pengaturan cincin bernilai standar yaitu A. Pengaturan cincin dapat memiliki nilai antara A – Z. Pengaturan cincin pada rotor dilakukan sesaat sebelum rotor terpilih dimasukkan ke dalam rangkaian mesin enigma. Nilai pengaturan cincin mengubah huruf enkripsi dengan aturan tiap huruf pada enkripsi ditambah sebanyak nilai pengaturan cincin dan digeser ke kanan sebanyak nilai pengaturan. Sebagai contoh, jika pengaturan cincin pada rotor I diubah menjadi B atau 2, dan rotor III diubah menjadi C atau 3, maka enkripsinya akan berubah menjadi seperti pada Tabel 2.4.

**Tabel 2.4** Perubahan Enkripsi dengan Nilai Pengaturan Cincin

Rotor\Input	ABCDEFGHIJKLMNOPQRSTUVWXYZ
I	KFLNGMHERWAOUPXZIYVTQBJCSD
III	SQDFHJLNERTVZXBPAKGYICMOWU

Pada Tabel 2.4 dapat dilihat bahwa pengaturan cincin mengubah enkripsi pada rotor I dan III. Ini merupakan salah satu pendekatan yang dapat dilakukan untuk memahami efek dari pengaturan cincin. Pendekatan lainnya adalah dengan memandang hubungan posisi rotor dengan pengaturan cincin terhadap hasil enkripsi, sehingga tidak perlu menganggap adanya perubahan

enkripsi secara langsung. Kedua pendekatan ini akan dibahas pada bab 3.

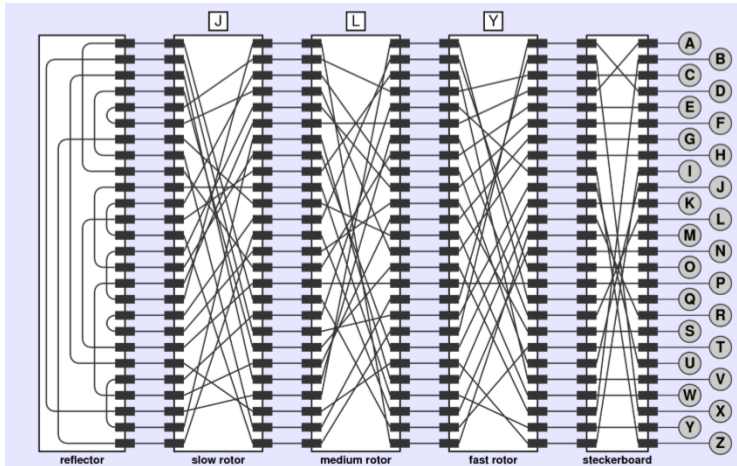
Mesin Enigma M3 menggunakan tiga rotor dari lima rotor yang ada dengan susunan yang bebas. Rotor terpilih ini akan menempati tiga bagian, yaitu rotor kanan, tengah dan kiri pada mesin. Rotor kanan adalah rotor yang berputar setiap huruf ingin dienkripsi, rotor tengah berputar setiap rotor kanan berputar melewati *notch*-nya, dan rotor kiri berputar jika rotor tengah berputar melewati *notch*-nya. Tiap rotor memiliki posisi *notch* yang berbeda. *Notch* tidak dipengaruhi oleh nilai pengaturan cincin pada rotor, melainkan akan tetap berada di posisinya terhadap bagian *input* rotor. Untuk posisi tiap *notch* dapat dilihat pada Tabel 2.5.

**Tabel 2.5** Posisi *Notch* pada Rotor

Rotor	<i>Posisi Notch</i>
I	R
II	F
III	W
IV	K
V	A

Salah satu hal yang membuat mesin Enigma M3 menjadi kompleks adalah *plugboard*. *Plugboard* digunakan untuk menukar dua buah huruf pada mesin. Jika huruf A ditukar dengan huruf J pada *plugboard*. Maka setiap huruf A ditekan pada keyboard, yang diproses ke dalam rotor adalah huruf J. Ketika hasil dari rangkaian enkripsi rotor mengeluarkan huruf J, maka keluaran sesungguhnya akan bertukar menjadi huruf A. Visualisasi dari *plugboard* dapat dilihat pada Gambar 2.3 dengan urutan reflektor, ketiga rotor, dan *plugboard* secara berurut dari kiri ke kanan. Gambar 2.2 dan Gambar 2.3 dapat dibandingkan untuk melihat perbedaan yang

lebih jelas. Gambar 2.3 adalah visualisasi dari reflektor, ketiga rotor dan *plugboard* secara berurut dari kiri ke kanan.



**Gambar 2.3** Alur Penyandian Mesin Enigma M3

Untuk menjalankan mesin Enigma M3, hal yang perlu diatur setelah pengaturan cincin, urutan rotor dan juga *plugboard* adalah posisi awal ring sebelum enkripsi dimulai. Posisi rotor didefinisikan dengan huruf. Jika posisi awal rotor adalah ABC, maka rotor kiri diputar ke posisi A, rotor tengah diputar ke huruf B, dan rotor kanan di tengah diputar ke huruf C. Seluruh posisi ini relatif terhadap masukan sesungguhnya. Sehingga jika huruf A masuk untuk dienkripsikan pada rotor kanan, maka huruf A dianggap masuk ke huruf C. Jika huruf B yang masuk pada rotor kanan ketika posisinya adalah C, maka B akan dianggap sebagai huruf D. Tabel 2.6 berisi gambaran mengenai posisi rotor ketika melakukan enkripsi dengan pengaturan cincin AAA, posisi rotor ABC, rotor I sebagai rotor kanan, rotor II sebagai rotor tengah, dan rotor III sebagai rotor kiri.

**Tabel 2.6** Posisi Awal Rotor Relatif Terhadap *Input*

Rotor\ <i>Input</i>	<b>A</b> BCDEFGHIJKLMNOPQRSTUVWXYZ
<i>Input</i> Rotor I	<b>C</b> DEFGHIJKLMNOPQRSTUVWXYZAB
Enkripsi Rotor I	MFLGDQVZNTOWYHXUSPAIBRCJEK
<i>Input</i> Rotor II	<b>B</b> CDEFGHIJKLMNOPQRSTUVWXYZA
Enkripsi Rotor II	JKSIRUXBLHWTMCQGNPYFVOEA
<i>Input</i> Rotor III	<b>A</b> BCDEFGHIJKLMNOPQRSTUVWXYZ
Enkripsi Rotor III	BDFHJLCPRTXVZNYEIWGAKMUSQO

Dengan seluruh perpaduan bagian mesin Enigma M3 di atas, dapat diketahui bahwa enkripsi dilakukan beberapa kali. Dan untuk huruf yang sama sebagai *input* awal, dapat memiliki berbagai hasil enkripsi. Hal ini yang menyebabkan mesin Enigma M3 tergolong sebagai *polyalphabetic cipher*.

*(Halaman ini sengaja dikosongkan)*

## **BAB III**

### **DESAIN DAN ANALISIS**

Pada bagian ini akan dijelaskan analisis dan desain sistem yang digunakan untuk menyelesaikan permasalahan pada Tugas Akhir ini.

#### **3.1 Analisis Observasi Permasalahan**

Pada bagian ini akan diberikan beberapa observasi dan analisis yang akan membantu jalannya implementasi.

##### **3.1.1 Posisi Rotor dan Pengaturan Cincin pada Proses Enkripsi**

Rotor secara keseluruhan akan terus berubah posisi ketika ada *input* dari *keyboard*. Perubahan posisi pada rotor terjadi melalui perputaran rotor yang artinya perputaran rotor sejalan dengan posisi rotor, di mana setiap rotor berputar, maka posisi rotor akan menunjukkan huruf selanjutnya. Seperti yang tertulis pada subbab 2.2.4 bahwa perputaran rotor menyebabkan posisi rotor bergerak dari posisi awal rotor. Perputaran rotor juga mengikuti aturan susunan kanan tengah dan kiri, di mana rotor kanan adalah rotor yang terus berputar setiap huruf yang akan dienkrpsi masuk, sedangkan rotor tengah berputar ketika rotor kanan berada pada posisi *notch* dan rotor kiri bergerak ketika rotor tengah berada pada posisi *notch*.

Dalam memahami hasil enkripsi dari proses yang terjadi pada rotor. Ada dua pendekatan yang dapat dilakukan. Pertama adalah dengan memperhitungkan bahwa pengaturan cincin secara langsung mengubah nilai enkripsi pada rotor. Kedua adalah dengan memperhitungkan pengaturan cincin dan posisi rotor hanya menyebabkan selisih pada masukan maupun keluaran pada rotor.

Penjelasan dari pendekatan pertama dapat diawali dengan membaca Tabel 2.6 yang memperlihatkan susunan enkripsi relatif terhadap posisi rotor yaitu ABC dan pengaturan cincin AAA. Ketika huruf mulai dienkrpsi dengan menekan tombol pada

*keyboard*, maka posisi akan berubah menjadi ABD, ABE dan seterusnya mengikuti aturan yang sudah dijelaskan sebelumnya. Perubahan enkripsi relatif terhadap *input* pada setiap perubahan posisi dapat dilihat pada Tabel 3.1.

**Tabel 3.1** Perubahan Posisi Rotor dengan Pengaturan Cincin AAA

Rotor\input	ABCDEFGHIJKLMN <strong>OP</strong> QRSTUVWXYZ
Posisi: A B C	
<i>Input</i> Rotor I	<b>C</b> DEFGHIJKLMN <strong>OP</strong> QRSTUVWXYZAB
Enkripsi Rotor I	MFLGDQVZNTOWYHXUSPAIBRCJEK
<i>Input</i> Rotor II	<b>BC</b> DEFGHIJKLMN <strong>OP</strong> QRSTUVWXYZA
Enkripsi Rotor II	JDKSIRUXBLHWTMCQGNPYFVOEA
<i>Input</i> Rotor III	<b>ABC</b> DEFGHIJKLMN <strong>OP</strong> QRSTUVWXYZ
Enkripsi Rotor III	BDFHJLCPRTXVZNYEIWGAKMUSQO
Posisi: A B D	
<i>Input</i> Rotor I	<b>DE</b> FGHIJKLMN <strong>OP</strong> QRSTUVWXYZABC
Enkripsi Rotor I	FLGDQVZNTOWYHXUSPAIBRCJEKM
<i>Input</i> Rotor II	<b>BC</b> DEFGHIJKLMN <strong>OP</strong> QRSTUVWXYZA
Enkripsi Rotor II	JDKSIRUXBLHWTMCQGNPYFVOEA
<i>Input</i> Rotor III	<b>ABC</b> DEFGHIJKLMN <strong>OP</strong> QRSTUVWXYZ
Enkripsi Rotor III	BDFHJLCPRTXVZNYEIWGAKMUSQO
Posisi: A B E	
<i>Input</i> Rotor I	<b>EFG</b> HIJKLMN <strong>OP</strong> QRSTUVWXYZABCD
Enkripsi Rotor I	LGDQVZNTOWYHXUSPAIBRCJEKMF
<i>Input</i> Rotor II	<b>BC</b> DEFGHIJKLMN <strong>OP</strong> QRSTUVWXYZA
Enkripsi Rotor II	JDKSIRUXBLHWTMCQGNPYFVOEA
<i>Input</i> Rotor III	<b>ABC</b> DEFGHIJKLMN <strong>OP</strong> QRSTUVWXYZ
Enkripsi Rotor III	BDFHJLCPRTXVZNYEIWGAKMUSQO



Pergerakan posisi mempengaruhi huruf yang akan dienkripsi. Sebagai contoh, ketika huruf A ingin dienkripsi pada posisi ABC, maka huruf A akan masuk terlebih dahulu ke rotor kanan yaitu rotor I sebagai C dan dienkripsi sebagai M, lalu ketika masuk ke *input* rotor tengah yaitu rotor II bukanlah huruf M yang akan masuk sebagai *input* melainkan L dikarenakan jarak C ke B adalah mundur sebanyak satu langkah, maka huruf M pun harus mundur sebanyak satu langkah untuk masuk ke rotor II. Begitu pun seterusnya untuk proses enkripsi. Ketika posisi berubah menjadi ABD, akan terlihat bahwa mengenkripsi huruf A kini masuk pada *input* Rotor I yang berbeda. Begitu pula dengan posisi ABE.

Ketika kembali dari reflektor, maka alurnya akan berbalik, dari enkripsi menuju *input* pada setiap rotor. Aturan posisi pun tetap berlaku dalam prosesnya. Sebagai contoh, posisi rotor adalah ABC dan sudah dalam kondisi kembali dari reflektor. Asumsikan *output* dari reflektor adalah D. karena posisi Rotor III adalah pada A, maka D akan masuk sebagai D pada enkripsi rotor III, lalu keluar melalui *input* Rotor III sebagai B. Lalu B diteruskan ke enkripsi Rotor II yang memiliki posisi B. Karena selisih antara B dan A adalah mundur satu langkah dan dalam kondisi invers dari enkripsi menuju *input*, maka operasinya menjadi maju satu langkah. Artinya ketika masuk ke enkripsi Rotor II, D akan masuk sebagai E, dan begitu seterusnya.

Perubahan posisi menyebabkan adanya selisih antar rotor ketika dilewati untuk proses enkripsi. Perubahan tersebut juga bersifat teratur dan mengikuti aturan dari *Caesar Cipher*. Pada kasus enigma, *key* pada *Caesar Cipher* hanya menunjukkan bagaimana perubahan posisi rotor menyebabkan selisih pada antar rotornya.

Pada subbab 2.2.4 telah dijelaskan juga mengenai pengaturan cincin pada tiap rotor yang dapat mengubah enkripsi yang terjadi dengan aturan tertentu. Pada Tabel 3.1 posisi rotor berada pada ABC dan nilai pengaturan cincin adalah AAA. Jika nilai pengaturan cincin adalah ABC dengan posisi rotor yang sama

maka hasilnya akan terlihat seperti pada Tabel 3.2. Aturan selisih antar posisi rotor tetap berlaku untuk huruf yang masuk pada tiap rotor.

**Tabel 3.2** Perubahan Posisi Rotor dengan Pengaturan Cincin ABC

Rotor\input	ABCDEFGHIJKLMNOPQRSTUVWXYZ
Posisi: A B C	
Input Rotor I	CDEFGHIJKLMNOPQRSTUVWXYZAB
Enkripsi Rotor I	GMOHNIFSXBVPQYAJZWURCKDTEL
Input Rotor II	BCDEFGHIJKLMNOPQRSTUVWXYZA
Enkripsi Rotor II	BKELTJSVYCMIXUNDRHAOQZGWPF
Input Rotor III	ABCDEFGHIJKLMNOPQRSTUVWXYZ
Enkripsi Rotor III	BDFHJLCPRTXVZNYEIWGAKMUSQO
Posisi: A B D	
Input Rotor I	DEFGHIJKLMNOPQRSTUVWXYZABC
Enkripsi Rotor I	MOHNIFSXBVPQYAJZWURCKDTELG
Input Rotor II	BCDEFGHIJKLMNOPQRSTUVWXYZA
Enkripsi Rotor II	BKELTJSVYCMIXUNDRHAOQZGWPF
Input Rotor III	ABCDEFGHIJKLMNOPQRSTUVWXYZ
Enkripsi Rotor III	BDFHJLCPRTXVZNYEIWGAKMUSQO
Posisi: A B E	
Input Rotor I	EFGHIJKLMNOPQRSTUVWXYZABCD
Enkripsi Rotor I	OHNIFSXBVPQYAJZWURCKDTELMG
Input Rotor II	BCDEFGHIJKLMNOPQRSTUVWXYZA
Enkripsi Rotor II	BKELTJSVYCMIXUNDRHAOQZGWPF
Input Rotor III	ABCDEFGHIJKLMNOPQRSTUVWXYZ
Enkripsi Rotor III	BDFHJLCPRTXVZNYEIWGAKMUSQO

Dengan penjelasan di atas, maka formula enkripsi untuk pendekatan pertama pada rotor dapat diciptakan di mana selisih antar rotor dan juga nilai pengaturan cincin menjadi bagian penting dalam pembentukannya. Berikut adalah formula yang dapat

digunakan berdasarkan penjelasan di atas untuk menentukan huruf enkripsi pada rotor terkait.

$$c_n = F_n(c_{n-1} - (Px_{n-1} - Px_n) - Rs_n) + Rs_n \quad (3.1)$$

Pada Persamaan 3.1  $n$  adalah representasi dari urutan tahapan rotor yang aktif pada mesin enigma, di mana nilai 1 untuk rotor kanan, 2 untuk rotor tengah dan 3 untuk rotor kiri.  $F$  fungsi enkripsi pada rotor,  $c$  adalah nilai enkripsi yang dihasilkan,  $Px$  adalah posisi rotor, dan  $Rs$  adalah nilai pengaturan cincin pada rotor. Nilai dari  $Px_0$  adalah konstan yaitu 1, karena merupakan *input*. Persamaan 3.1 pun terbatas pada huruf dengan rentang A – Z.

Pendekatan kedua adalah dengan memandang pengaturan cincin dan posisi rotor hanya menyebabkan selisih pada masukan dan keluaran pada rotor. Tidak seperti pendekatan pertama yang mengubah hasil enkripsi ketika pengaturan cincin diterapkan. Pendekatan kedua hanya menghitung selisih dari pengaturan cincin dan posisi rotor untuk menentukan masukan pada rotor dan hasil enkripsi pada rotor.

**Tabel 3.3** Keadaan Rotor I dan II dengan Pengaturannya

<i>Input Statis</i>		ABCDEFGHIJKL <u>M</u> NOPQRSTUVWXYZ
Rotor I Posisi: K Pengaturan Cincin: B	<i>Input</i>	JKLMNO <u>P</u> QRSTUVWXYZABCDEFGHI
	Enkripsi	ZNTOWY <u>X</u> USPAIBRCJMKFLGDQV
	Enkripsi (Diurut)	JKLMNOPQRSTUVWXYZABCDEFGHI <u>J</u>
	Cincin Rotor	JKLMNOPQR <u>S</u> TUVWXYZABCDEFGHIJ
<i>Input Statis</i>		ABCDEFGHIJKL <u>M</u> NOPQRSTUVWXYZ
Rotor II Posisi: F Pengaturan Cincin: A	<i>Input</i>	FGHIJKLMNO <u>P</u> QRSTUVWXYZABCDE
	Enkripsi	AJDKSIRUXBLHWTMCQGNPYFV <u>O</u> E
	Enkripsi (Diurut)	FGHIJKLMNO <u>P</u> QRSTUVWXYZABCDE
	Cincin Rotor	FGHIJKLMNO <u>P</u> QRSTUVWXYZABCDE

Tabel 3.3 menunjukkan bagaimana Rotor I dan Rotor II terhubung ketika mengenkripsikan. Seperti terlihat bahwa Rotor I berada pada posisi K dan representasi dari posisinya adalah cincin rotor, serta Rotor II dengan posisi F. Rotor I memiliki pengaturan cincin B dan Rotor I memiliki pengaturan cincin A.

Dimisalkan pada keadaan tersebut huruf yang akan dienkripsi adalah G, maka pada *input* statis paling atas akan meneruskan pada *input* Rotor I sebesar selisih antara posisi Rotor I dan pengaturan cincinnya. Posisi Rotor I adalah K dan pengaturan cincin adalah B maka selisihnya adalah  $K - B = 11 - 2 = 9$ . G pada *input* statis dan P pada *input* Rotor I memiliki selisih 9. Begitu pula dengan selisih hasil enkripsi yang diurutkan dengan *input* statis menuju Rotor II yang memiliki selisih 9, yaitu jarak dari H ke Y. Oleh sebab itu Persamaan 3.2 dapat diberlakukan pada pendekatan ini.

$$c_n = F_n(c_{n-1} + (Px_n - Rs_n)) - (Px_n - Rs_n) \quad (3.2)$$

Persamaan 3.2 mengikuti ketentuan pada Persamaan 3.1 mengenai keterangan dari penggunaan variabel dan juga ketentuan perhitungan. Dalam mewujudkan implementasi untuk menyelesaikan permasalahan penyandian mesin Enigma M3, persamaan yang digunakan adalah Persamaan 3.2

### 3.1.2 *Double-stepping* pada Rotor

Rotor kanan berputar berdasarkan masukan *keyboard*, rotor tengah berputar ketika rotor kanan melewati *notch* dan rotor kiri berputar ketika rotor tengah melewati *notch*. Namun, pada mesin enigma ada sebuah peristiwa yang dinamakan *double-stepping* yang menyebabkan rotor tengah bergerak sebanyak dua kali.

*Double-stepping* hanya terjadi ketika rotor tengah berada pada posisi tepat satu sebelum *notch*-nya. Di mana rotor tengah tidak menunggu rotor kanan melewati *notch* untuk bergerak, melainkan langsung bergerak dari posisi satu sebelum *notch* ke posisi *notch* setelah masuk satu enkripsi untuk memutar rotor

kanan. Tabel 3.4 menampilkan bagaimana posisi berubah ketika masukan diberikan. *Notch* pada rotor dapat dilihat pada Tabel 2.5. Pada masukan ke-3 rotor tengah berada tepat satu sebelum *notch* yaitu E, maka pada masukan berikutnya, rotor langsung berputar ke F.

**Tabel 3.4** *Double-stepping* pada Mesin Enigma M3

Masukan Ke-	1	2	3	4	5
Posisi Rotor Kanan (I)	P	Q	R	S	T
Posisi Rotor Tengah (II)	D	D	E	F	F
Posisi Rotor Kiri (III)	F	F	F	G	G

### 3.2 Deskripsi Umum Program

Program diawali dengan memberikan masukan berupa *input* bilangan bulat  $T$  sebagai jumlah *test case* yang akan dilakukan, di mana  $0 < T \leq 100$ . Selanjutnya program akan menjalankan tahap *init\_plugboard* untuk membuat kondisi awal *plugboard* ketika penukaran huruf belum terjadi. Selanjutnya akan menerima masukan berupa *input* untuk  $U$ ,  $X$ ,  $P$  sebagai urutan dan penggunaan rotor, nilai pengaturan cincin tiap rotor, dan posisi awal rotor. Selanjutnya akan langsung menjalankan tahap *swap\_char* yang diawali dengan penerimaan *input* untuk  $H$  sebagai pasangan huruf yang akan ditukar. Fungsi pada seluruh bagian ini adalah tahap persiapan mesin Enigma M3 sebelum tahap enkripsi.

Setelah tahap persiapan selesai, lalu tahap enkripsi dimulai dengan masukan berupa *input* pada  $E$ . setelah itu tahap *enkripsi* dijalankan di mana pada tahapan *enkripsi* terdapat tahap *rotor\_step*. Hasil dikeluarkan setelah seluruh proses enkripsi selesai.

Dalam menyelesaikan kasus mesin Enigma M3 ini, diperlukan setidaknya 13 variabel untuk menyimpan nilai – nilai yang diperlukan dalam menjalankan program. Daftar nama variabel dan kegunaannya dapat dilihat pada Tabel 3.5.

**Tabel 3.5** Daftar Variabel

Tipe Variabel	Nama Variabel	Deskripsi
int	T	Menyimpan nilai <i>test case</i> .
int	i,j	Berfungsi sebagai iterator.
char	B[26]	Menyimpan huruf pada <i>plugboard</i> .
char	R[11][27]	Menyimpan nilai enkripsi dan invers enkripsi serta reflektor untuk penyandian.
char	N[5]	Menyimpan nilai <i>notch</i> tiap rotor.
char	H[40]	Menyimpan nilai perubahan untuk digunakan pada <i>plugboard</i> dan dapat bernilai null.
char	U[4]	Menyimpan nilai rotor yang digunakan secara berurut.
char	X[4]	Menyimpan nilai pengaturan rotor secara berurut.
char	P[4]	Menyimpan posisi awal rotor secara berurut.
char	E[650000]	Menyimpan <i>plaintext</i> dan digunakan langsung dalam operasi enkripsi sehingga sekaligus menyimpan <i>ciphertext</i> .
int	l	Menyimpan nilai <i>flag</i> untuk mengakses rotor, pengaturan rotor, dan posisi rotor pada tahapan <i>enkripsi</i> .
int	k	Menyimpan nilai <i>flag</i> untuk mengakses reflektor pada tahapan <i>enkripsi</i> .

Tahapan algoritma untuk menyelesaikan permasalahan mesin Enigma M3 untuk menyandikan adalah sebagai berikut.

1. Menerima *input T* untuk *test case*.
2. Untuk setiap *test case*, *plugboard* akan dikembalikan pada pengaturan standar melalui tahapan *init\_plugboard*.
3. Menerima *input U, X, P* untuk rotor yang digunakan, pengaturan cincin untuk setiap rotor yang digunakan dan posisi awal rotor ketika mulai menyandikan.
4. Menerima *input H* lalu menukar huruf pada *plugboard* berdasarkan *H* melalui tahap *swap\_char*.
5. Menerima *input plaintext* yang akan dienkripsikan.
6. Pada tahap *enkripsi*, setiap mengakses satu huruf untuk dienkripsi, dijalankan tahapan *rotor\_step* untuk mengubah posisi rotor sesuai dengan aturan yang telah dijelaskan pada bagian sebelumnya.
7. Setelah melalui *rotor\_step* maka huruf akan diubah dengan mengakses *B* dan juga proses enkripsi pada rotor berdasarkan Persamaan 3.2.
8. Hasil enkripsi dari rotor kemudian diteruskan ke *B* untuk diubah terakhir kalinya. Lalu lanjut mengenkripsi huruf berikutnya.
9. Jika enkripsi telah selesai, maka hasil enkripsi ditampilkan dan lanjut ke *test case* berikutnya.
10. Jika *test case* habis, program dinyatakan selesai.

### 3.3 Desain Algoritma

Pada bagian ini akan dipaparkan *pseudocode* dari semua tahapan algoritma yang digunakan pada program. Gambar 3.1 merupakan alur kerja program secara umum.

main()	
1.	$T \leftarrow \text{input}()$
2.	<i>for</i> $i \leftarrow 0$ <i>to</i> $T$
3.	<i>init_plugboard</i> ()
4.	$U, X, P \leftarrow \text{input}()$
5.	<i>swap_char</i> ()
6.	$E \leftarrow \text{input}()$
7.	<i>enkripsi</i> ()
8.	$\text{output}() \leftarrow E$

**Gambar 3.1** *Pseudocode Main*

### 3.3.1 Desain Tahap *init\_plugboard*

Tahap *init\_plugboard* digunakan untuk menginisialisasi isi daripada *plugboard* yang akan digunakan untuk menukar huruf pada mesin Enigma M3. *Pseudocode*-nya dapat dilihat pada Gambar 3.2. Nilai 65 pada baris 2 adalah untuk membentuk kode ASCII.

<i>init_plugboard</i>	
1.	<i>for</i> $i \leftarrow 0$ <i>to</i> 26
2.	$B[i] = i + 65$

**Gambar 3.2** *Pseudocode Tahap init\_plugboard*

### 3.3.2 Desain Tahap *swap\_char*

Tahap *swap\_char* akan memasukkan *input* untuk  $H$  dan menukar huruf yang ada pada  $B$  yang berperan sebagai *plugboard* berdasarkan  $H$ . Gambar 3.3 merupakan *pseudocode* dari tahap *swap\_char*.



<i>swap_char</i>	
1.	$H \leftarrow \text{input}()$
2.	<i>For</i> $i \leftarrow 0$ <i>to</i> $\text{strlen}() \leftarrow B$
3.	$B[i] \leftarrow H[i+1]$
4.	$B[i+1] \leftarrow H[i]$
5.	$i \leftarrow i + 3$

**Gambar 3.3** Pseudocode Tahap *swap\_char*

### 3.3.3 Desain Tahap *enkripsi*

Tahap *enkripsi* merupakan bagian utama dalam program. Pada tahapan ini, *array of char* yaitu *E*, akan diakses satu persatu lalu dienkripsi berdasarkan tahapan pada mesin Enigma M3. Pada bagian ini juga terdapat tahap *rotor\_step* pada Gambar 3.5. Gambar 3.4 menampilkan *pseudocode* dari tahapan *enkripsi*. Nilai 65 dan 26 pada bagian kode ini merupakan nilai yang berfungsi mengonversi nilai yang ada berdasarkan kode ASCII.

<i>enkripsi</i>	
1.	<i>for</i> $k \leftarrow 0$ <i>to</i> $\text{strlen}() \leftarrow E$
2.	<i>If</i> $(E[k] == ' ')$ <i>continue</i>
3.	<i>rotor_step</i> $()$
4.	$E[j] \leftarrow B[E[j]-65]$
5.	<i>for</i> $i \leftarrow -2$ <i>to</i> $-4$
6.	<i>if</i> $(i == -1)$ $E[j] = R[5][E[j]-65]$ , <i>continue</i>
7.	<i>if</i> $(i < -1)$ $k=6$ , $l=l+2$
8.	$E[j] \leftarrow (R[U[i+l]-49+k][(E[j]-65+26+(P[i+l]-$
9.	$X[i+l]))\%26)-(P[i+l]-X[i+l])+26-65)\%26+65$
10.	$E[j] \leftarrow B[E[j]-65]$

**Gambar 3.4** Pseudocode Tahap *enkripsi*

### 3.3.4 Desain Tahap *rotor\_step*

Kegunaan dari tahap *rotor\_step* adalah sebagai penggerak posisi rotor. Tahapan ini juga sekaligus menyelesaikan permasalahan yang dipaparkan pada subbab 3.1.2. Gambar 3.5

merupakan *pseudocode* dari tahap *rotor\_step*. Nilai 49 pada kode berfungsi untuk mengonversi nilai berdasarkan kode ASCII.

<i>rotor_step</i>	
1.	$P[2] \leftarrow (P[2] + 1 - 65) \% 26 + 65$
2.	If $P[2] == N[U[2] - 49]$    $P[1] == N[U[1] - 49] - 1$
3.	$P[1] \leftarrow (P[1] + 1 - 65) \% 26 + 65$
4.	If $P[1] == N[U[1] - 49]$
5.	$P[0] \leftarrow (P[0] + 1 - 65) \% 26 + 65$

**Gambar 3.5** *Pseudocode* Tahap *rotor\_step*

## BAB IV IMPLEMENTASI

Pada bab ini akan dijelaskan mengenai implementasi dari algoritma berdasarkan desain yang telah dilakukan.

### 4.1 Lingkungan Implementasi

Lingkungan implementasi menggunakan sebuah laptop dengan spesifikasi perangkat keras dan perangkat lunak seperti yang tercantum pada Tabel 4.1.

**Tabel 4.1** Spesifikasi Lingkungan Implementasi

No.	Jenis Perangkat	Spesifikasi
1.	Perangkat Keras	<ul style="list-style-type: none"><li>• <i>Processor</i> Intel(R) Core(TM) i5-6200 CPU @ 2.30 GHz 2.40 GHz</li><li>• <i>Memory</i> 4GB</li></ul>
2.	Perangkat Lunak	<ul style="list-style-type: none"><li>• Windows 10 Pro 64-bit</li><li>• <i>Integrated Development Environment</i> Dev-C++ 5.11</li></ul>

### 4.2 Penggunaan *Library* dan Inisialisasi Variabel

Pada bagian ini dijelaskan *library* dari bahasa C dan juga inisiasi variabel dan juga konstanta yang akan digunakan pada seluruh program. Terdapat dua *library* yang digunakan, yaitu: *stdio.h*, dan *string.h*. Kode sumber pemanggilan *library* dapat dilihat pada Kode Sumber 4.1.

```
1. #include<stdio.h>
2. #include<string.h>
```

**Kode Sumber 4.1** Implementasi *Library*

Pada *library stdio.h* fungsi yang digunakan adalah fungsi standar dan *string.h* digunakan untuk memanggil fungsi *strlen* dalam menghitung panjang karakter pada suatu variabel.

```

1. int T,i,j;
2. scanf("%d",&T);
3. while(T--){
4.     char E[65000],U[4],X[4],P[4],H[40],B[26],
5.     N[5]={ "RFWKA" },
6.     R[11][27]={ "EKMFLGDQVZNTOWYHXUSPAIBRCJ",
7.                 "AJDKSIRUXBLHWTMCQGZNPYFVOE",
8.                 "BDFHJLCPRTXVZNYEIWGAKMUSQO",
9.                 "ESOVZJAYQUIRHXLNFTGKDCMWB",
10.                "VZBRGITYUPSDNHLXAWMJQOFECK",
11.                "YRUHQSLDPXNGOKMIEBFZCWVJAT",
12.                "UWYGADFPVZBECKMTHXSLRINQOJ",
13.                "AJPCZWRLFBDKOTYUQGENHXMIVS",
14.                "TAGBPCSDQEUFVNZHYIXJWLKROM",
15.                "HZWVARTNLGUPXQCEJMBSKDYOIF",
16.                "QCYLXWENFTZOSMVJUDKGIARPHB" };

```

#### Kode Sumber 4.2 Inisialisasi Variabel

Inisiasi variabel dilakukan ke dalam dua tahap. Tahap pertama sebelum *input test case T* dan tahap kedua adalah di dalam perulangan dengan batasan *T*. Ini dimaksudkan agar setiap *test case* berikutnya, maka semua nilai di-*reset*. Setiap kegunaan dari variabel mengacu pada Tabel 3.2. Implementasi dari inisialisasi variabel dapat dilihat pada Kode Sumber 4.2.

#### 4.3 Implementasi Tahap *init\_plugboard*

Tahapan ini akan membuat pasangan huruf yang homogen sebelum terjadinya pertukaran huruf pada *plugboard*. Implementasi dari tahap ini sesuai dengan penjelasan pada subbab 3.3.1 dan dapat dilihat pada Kode Sumber 4.3.

```

1. for(i=0;i<26;i++){ B[i]=i+65; }

```

#### Kode Sumber 4.3 Tahap *init\_plugboard*

#### 4.4 Implementasi Tahap *swap\_char*

Sesuai dengan penjelasan pada subbab 3.3.2. Tahap *swap\_char* digunakan untuk melakukan pertukaran huruf pada *plugboard*. Implementasi dari tahap ini dapat dilihat pada Kode Sumber 4.4.

```

1. gets(H);
2. for(i=0;i<strlen(H);i+=3){
3.     B[H[i]-65]=H[i+1];
4.     B[H[i+1]-65]=H[i];
5. }
```

**Kode Sumber 4.4** Tahap *swap\_char*

#### 4.5 Implementasi Tahap *enkripsi*

Implementasi ini merujuk kepada subbab 3.3.3 di mana tahap *enkripsi* merupakan bagian untuk mengenkripsi huruf satu per satu dengan mengakses *array of char* pada *E*. Implementasi dari tahap *enkripsi* dapat dilihat pada Kode Sumber 4.5.

```

1. for(j=0;j<strlen(E);j++){
2.     int l=0, k=0;
3.     if(E[j]!=' '){continue;}
4.     //Implementasi Kode Sumber 4.6
5.     E[j]=B[E[j]-65];
6.     for(i=2;i>=-4;i--){
7.         if(i==-1){E[j]=R[5][E[j]-65];continue;}
8.         if(i<-1){k=6;l+=2;}
9.         E[j]=(R[U[i+1]-49+k][(E[j]-65+26+(P[i+1]-
X[i+1]))%26]-(P[i+1]-X[i+1])+26-65)%26+65;
10.     }
11.     E[j]=B[E[j]-65];
12. }
```

**Kode Sumber 4.5** Tahap *enkripsi*

#### 4.6 Implementasi Tahap *rotor\_step*

Sesuai dengan penjelasan pada subbab 3.3.4, tahap *rotor\_step* adalah sebagai pengubah posisi rotor selama proses enkripsi belum berakhir pada mesin Enigma M3. Implementasi dari tahap *rotor\_step* dapat dilihat pada Kode Sumber 4.6.

```

1. P[2]=(P[2]+1-65)%26+65;
2. if(P[2]==N[U[2]-49]||P[1]==N[U[1]-49]-1){
3.     P[1]=(P[1]+1-65)%26+65;
4.     if(P[1]==N[U[1]-49]){
5.         P[0]=(P[0]+1-65)%26+65;
6.     }
7. }
```

**Kode Sumber 4.6** Tahap *rotor\_step*

#### 4.7 Implementasi Fungsi *Main*

Fungsi *main* merupakan bagian utama dalam program ini. Pada awal sistem akan menerima input *T* sebagai *test case*. Pada setiap *test case*, diharuskan memasukkan *input* untuk pengaturan mesin Enigma M3 yaitu: *U*, *X*, *P*. Selanjutnya akan menjalankan tahap *swap\_char* dan *enkripsi*. Kode Sumber 4.7 merupakan implementasi fungsi *main*.

```

1. #include<stdio.h>
2. #include<string.h>
3. int main(){
4.     int T,i,j;
5.     scanf("%d",&T);
6.     while(T--){
7.         //Implementasi Kode Sumber 4.2
8.         //Implementasi Kode Sumber 4.3
9.         scanf("%3s %3s %3s",U,X,P);getchar();
10.        //Implementasi Kode Sumber 4.4
11.        gets(E);
12.        //Implementasi Kode Sumber 4.5
13.        printf("%s\n", E);
14.    }
15.    return 0;}
```

**Kode Sumber 4.7** Fungsi *main*

## **BAB V**

### **UJI COBA DAN EVALUASI**

Pada bab ini akan dijelaskan mengenai hasil uji coba program yang telah dirancang pada tugas akhir ini. Uji coba dilakukan untuk mengetahui kinerja algoritma dengan lingkungan uji coba yang telah ditentukan.

#### **5.1 Lingkungan Uji Coba**

Lingkungan implementasi menggunakan sebuah laptop dengan spesifikasi perangkat keras dan perangkat lunak seperti yang tercantum pada Tabel 5.1.

**Tabel 5.1** Spesifikasi Lingkungan Uji Coba

No.	Jenis Perangkat	Spesifikasi
1.	Perangkat Keras	<ul style="list-style-type: none"><li>• <i>Processor</i> Intel(R) Core(TM) i5-6200 CPU @ 2.30 GHz 2.40 GHz</li><li>• <i>Memory</i> 4GB</li></ul>
2.	Perangkat Lunak	<ul style="list-style-type: none"><li>• Windows 10 Pro 64-bit</li><li>• <i>Integrated Development Environment</i> Dev-C++ 5.11</li></ul>

#### **5.2 Uji Coba Kebenaran**

Pada subbab ini akan dijelaskan uji coba kebenaran yang dilakukan untuk menguji desain dan implementasi yang sudah dibuat untuk menyelesaikan tugas akhir ini.

##### **5.2.1 Uji Coba Kebenaran Lokal**

Uji coba kebenaran lokal dilakukan dengan cara analisis penyelesaian dari beberapa contoh kasus uji menggunakan metode penyelesaian sesuai seperti yang telah dijelaskan pada bab 2 dan bab 3. Kasus yang akan digunakan sebagai bahan uji coba kebenaran lokal adalah contoh kasus uji yang diberikan SPOJ dan satu kasus tambahan untuk membandingkan pilihan dan urutan rotor dengan pengaturan cincin, posisi rotor dan *plugboard* yang

sama. Ada tiga contoh kasus uji yang akan dibahas dan dapat dilihat pada Gambar 5.1.

<pre>123 JAN DER SP OJ RU LZ THISX ISXAN XEXAM PLEXI NPUT</pre>	<pre>543 SPO JPL SH OR TE NI YUQKD YVPSF HCQEI VHAPE NAQZQ I</pre>
(a)	(b)
<pre>421 JAN DER SP OJ RU LZ THISX ISXAN XEXAM PLEXI NPUT</pre>	
(c)	

**Gambar 5.1** Kasus Uji (a) Kasus Uji-1 dari SPOJ (b) Kasus Uji-2 dari SPOJ dan (c) Kasus Uji-3 sebagai pembandingan dengan kasus Uji-1

Pada setiap kasus uji, hal pertama yang dilakukan terlebih dahulu adalah menyiapkan *plugboard* dengan tahap *init\_plugboard*. Dengan begitu isi dari *array B* adalah huruf yang dimulai dari A – Z secara berurut dari *array* ke-0 sampai ke-25 pada *B*. *R* yang merupakan *array of char* dua dimensi berisi enkripsi tiap rotor, invers enkripsi tiap rotor dan juga reflektor. Spesifikasi dari isi *R* dapat dilihat pada Tabel 5.2.



**Tabel 5.2** Spesifikasi *R*

R[ ]	Value	Keterangan
0	EKMFLGDQVZNTOWYHXUSPAIBRCJ	Rotor I
1	AJDKSIRUXBLHWTMCQGZNPYFVOE	Rotor II
2	BDFHJLCPRTXVZNYEIWGAKMUSQO	Rotor III
3	ESOVZPJAYQUIRHXLNFTGKDCMWB	Rotor IV
4	VZBRGITYUPSDNHLXAWMJQOFECK	Rotor V
5	YRUHQSLDPXNGOKMIEBFZCWVJAT	Reflektor
6	UWYGADFPVZBECKMTHXSLRINQOJ	Inv. Rotor I
7	AJPCZWRLFBDKOTYUQGENHXMIVS	Inv. Rotor II
8	TAGBPCSDQEUFVNZHYIXJWLKROM	Inv. Rotor III
9	HZWVARTNLGUPXQCEJMBSKDYOIF	Inv. Rotor IV
10	QCYLXWENFTZOSMVJUDKGIARPHB	Inv. Rotor V

Mesin Enigma M3 memiliki urutan untuk perubahan setiap hurufnya. Dalam tugas akhir ini, urutan pengerjaannya akan tertulis dengan urutan sebagai berikut.

Huruf awal → *Plugboard* → Rotor kanan → Rotor tengah → Rotor kiri → Reflektor → Invers rotor kiri → Invers rotor tengah → Invers rotor kanan → *Plugboard* (hasil akhir).

### 5.2.1.1 Kasus Uji-1

- Rotor kiri, tengah dan kanan secara berurut adalah Rotor I, Rotor II, dan Rotor III
- Nilai pengaturan rotor kiri, tengah dan kanan secara berurut adalah J, A, dan N.
- Posisi awal rotor pada rotor kiri, tengah dan kanan adalah D, E dan R.
- Huruf yang ditukar pada *plugboard* adalah S dengan P, O dengan J, R dengan U, dan L dengan Z.
- *Plaintext*-nya adalah *THISX ISXAN XEXAM PLEXI NPUT*

**Tabel 5.3** Pasangan Huruf *Plugboard* pada Kasus Uji-1

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
A B C D E F G H I O K Z M N J S Q U P T R V W X Y L

**Tabel 5.4** Keadaan Awal Mesin Enigma M3 pada Kasus Uji-1

Rotor <input style="width: 100px; border: none;" type="text"/>	ABCDEFGHIJKLMNOPQRSTUVWXYZ
Posisi: D E R	
<i>Input</i> Rotor III	DEFGHIJKLMNOPQRSTUVWXYZABC
Enkripsi Rotor III	WYPCEGKIMALRVJTNXZHFDBOQSU
<i>Input</i> Rotor II	EFGHIJKLMNOPQRSTUVWXYZABCD
Enkripsi Rotor II	SIRUXBLHWTMCQZNPYFVOEAJDK
<i>Input</i> Rotor I	RSTUVWXYZABCDEFGHIJKLMNOQP
Enkripsi Rotor I	JRKALSNTVOUPMZEIWCXFHQGDBY

Hasil secara bertahap dari tiap huruf pada Kasus Uji-1 adalah sebagai berikut:

- Posisi Rotor: E F S  
Tahapan: **T**→T→L→L→I→P→G→F→P→**S**
- Posisi Rotor: E F T  
Tahapan: **H**→H→H→R→T→Z→W→E→O→**J**
- Posisi rotor: E F U  
Tahapan: **I**→I→X→Y→U→C→V→V→Z→**L**
- Posisi rotor: E F V  
Tahapan: **S**→P→K→X→X→J→F→Y→K→**K**
- Posisi rotor: E G W  
Tahapan: **X**→X→T→Y→U→C→V→D→M→**M**
- Posisi rotor: E G X  
Tahapan: **I**→I→W→X→X→J→F→E→P→**S**
- Posisi rotor: E G Y  
Tahapan: **S**→P→Q→Z→F→S→P→R→V→**V**
- Posisi rotor: E G Z  
Tahapan: **X**→X→H→N→A→Y→Q→G→L→**Z**

- Posisi rotor: E G A  
Tahapan: **A**→A→A→L→I→P→G→I→Y→Y
- Posisi rotor: E G B  
Tahapan: **N**→N→P→S→B→R→H→N→M→M
- Posisi rotor: E G C  
Tahapan: **X**→X→K→K→L→G→B→F→H→H
- Posisi rotor: E G D  
Tahapan: **E**→E→U→U→M→O→E→X→X→X
- Posisi rotor: E G E  
Tahapan: **X**→X→H→N→A→Y→Q→G→T→T
- Posisi rotor: E G F  
Tahapan: **A**→A→O→J→Q→E→O→B→R→U
- Posisi rotor: E G G  
Tahapan: **M**→M→S→I→K→N→A→L→W→W
- Posisi rotor: E G H  
Tahapan: **P**→S→F→B→G→L→K→K→V→V
- Posisi rotor: E G I  
Tahapan: **L**→Z→P→S→B→R→H→N→V→V
- Posisi rotor: E G J  
Tahapan: **E**→E→F→B→G→L→K→K→W→W
- Posisi rotor: E G K  
Tahapan: **X**→X→N→H→R→B→S→P→Y→Y
- Posisi rotor: E G L  
Tahapan: **I**→I→E→F→J→X→X→W→Y→Y
- Posisi rotor: E G M  
Tahapan: **N**→N→A→L→I→P→G→I→E→E
- Posisi rotor: E G N  
Tahapan: **P**→S→G→Q→Y→A→N→H→D→D
- Posisi rotor: E G O  
Tahapan: **U**→R→F→B→G→L→K→K→E→E
- Posisi rotor: E G P  
Tahapan: **T**→T→K→K→L→G→B→F→B→B

Hasil enkripsi atau *ciphertext* pada Kasus Uji-1 adalah **SJLKM SVZYM HXTUW VVWYY EDEB**.

### 5.2.1.2 Kasus Uji-2

- Rotor kiri, tengah dan kanan secara berurut adalah Rotor V, Rotor IV, dan Rotor III
- Huruf yang ditukar pada *plugboard* adalah S dengan H, O dengan R, T dengan E, dan N dengan I.
- Nilai pengatur rotor kiri, tengah dan kanan secara berurut adalah S, P, dan O.
- Posisi awal rotor pada rotor kiri, tengah dan kanan adalah J, P dan L.
- *Plaintext*-nya adalah *YUQKD YVPSF HCQEI VHAPE NAQZQ I*

**Tabel 5.5** Pasangan Huruf *Plugboard* pada Kasus Uji-2

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
A	B	C	D	T	F	G	S	N	J	K	L	M	I	R	P	Q	O	H	E	U	V	W	X	Y	Z

**Tabel 5.6** Keadaan Awal Mesin Enigma M3 pada Kasus Uji-2

Rotor <input style="width: 100px; border: none;" type="text"/>	ABCDEFGHIJKLMNOPQRSTUVWXYZ
	Posisi: D E R
<i>Input</i> Rotor III	OPQRSTUVWXYZABCDEFGHIJKLMN
Enkripsi Rotor III	SUWNACEIGKYJPTHRLVXFDBZMOQ
<i>Input</i> Rotor IV	PQRSTUVWXYZABCDEFGHIJKLMNO
Enkripsi Rotor IV	THDKEOYPNFJXGWMACUIVZSRBLQ
<i>Input</i> Rotor V	STUVWXYZABCDEFGHIJKLMNOPQR
Enkripsi Rotor V	YBMWQUGJFVSZXONLTEIKAPRCHD

Hasil secara bertahap dari tiap huruf pada Kasus Uji-2 adalah sebagai berikut:

- Posisi rotor: J P M  
Tahapan: **Y**→**Y**→**W**→**C**→**S**→**F**→**A**→**H**→**E**→**T**

- Posisi rotor: J P N  
Tahapan: **U**→U→B→S→Y→A→M→X→S→**H**
- Posisi rotor: J P O  
Tahapan: **Q**→Q→I→Y→G→L→H→N→N→**I**
- Posisi rotor: J P P  
Tahapan: **K**→K→U→K→I→P→N→Q→H→**S**
- Posisi rotor: J P Q  
Tahapan: **D**→D→J→Q→H→D→R→M→X→**X**
- Posisi rotor: J P R  
Tahapan: **Y**→Y→A→E→X→J→Z→F→N→**I**
- Posisi rotor: J P S  
Tahapan: **V**→V→K→U→M→O→F→R→H→**S**
- Posisi rotor: J P T  
Tahapan: **P**→P→F→Z→J→X→E→A→X→**X**
- Posisi rotor: J P U  
Tahapan: **S**→H→H→A→F→S→C→W→A→**A**
- Posisi rotor: J P V  
Tahapan: **F**→F→S→T→B→R→O→C→X→**X**
- Posisi rotor: J Q W  
Tahapan: **H**→S→T→J→E→Q→W→N→D→**D**
- Posisi rotor: J Q X  
Tahapan: **C**→C→M→G→N→K→L→W→T→**E**
- Posisi rotor: J Q Y  
Tahapan: **Q**→Q→R→S→Y→A→M→P→C→**C**
- Posisi rotor: J Q Z  
Tahapan: **E**→T→Y→A→F→S→C→U→R→**O**
- Posisi rotor: J Q A  
Tahapan: **I**→N→C→U→M→O→F→S→D→**D**
- Posisi rotor: J Q B  
Tahapan: **V**→V→E→Y→G→L→H→K→X→**X**
- Posisi rotor: J Q C  
Tahapan: **H**→S→O→K→I→P→N→B→T→**E**
- Posisi rotor: J Q D  
Tahapan: **A**→A→P→M→A→Y→S→R→D→**D**

- Posisi rotor: J Q E  
Tahapan: **P**→P→V→B→V→W→V→X→X→**X**
- Posisi rotor: J Q F  
Tahapan: **E**→T→G→Z→J→X→E→Q→M→**M**
- Posisi rotor: J Q G  
Tahapan: **N**→I→J→T→B→R→O→D→T→**E**
- Posisi rotor: J Q H  
Tahapan: **A**→A→H→X→U→C→P→I→H→**S**
- Posisi rotor: J Q I  
Tahapan: **Q**→Q→D→O→R→B→T→J→H→**S**
- Posisi rotor: J Q J  
Tahapan: **Z**→Z→P→M→A→Y→S→R→A→**A**
- Posisi rotor: J Q K  
Tahapan: **Q**→Q→D→O→R→B→T→J→G→**G**
- Posisi rotor: J Q L  
Tahapan: **I**→N→A→R→D→H→Q→L→T→**E**

Hasil enkripsi atau *ciphertext* pada Kasus Uji-2 adalah *THISX ISXAX DECOD XEDXM ESSAG E*.

### 5.2.1.3. Kasus Uji-3

- Rotor kiri, tengah dan kanan secara berurut adalah Rotor IV, Rotor II, dan Rotor I
- Nilai pengaturan rotor kiri, tengah dan kanan secara berurut adalah J, A, dan N.
- Posisi awal rotor pada rotor kiri, tengah dan kanan adalah D, E dan R.
- Huruf yang ditukar pada *plugboard* adalah S dengan P, O dengan J, R dengan U, dan L dengan Z.
- *Plaintext*-nya adalah *THISX ISXAN XEXAM PLEXI NPUT*

**Tabel 5.7** Pasangan Huruf *Plugboard* pada Kasus Uji-3

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
A	B	C	D	E	F	G	H	I	O	K	Z	M	N	J	S	Q	U	P	T	R	V	W	X	Y	L

**Tabel 5.8** Keadaan Awal Mesin Enigma M3 pada Kasus Uji-3

Rotor <input/>	ABCDEFGHIJKLMNOPQRSTUVWXYZ
Posisi: D E R	
<i>Input</i> Rotor I	OPQRSTUVWXYZABCDEFGHIJKLMN
Enkripsi Rotor I	SUWNACEIGKYJPTHRLVXFDBZMOQ
<i>Input</i> Rotor II	PQRSTUVWXYZABCDEFGHIJKLMNO
Enkripsi Rotor II	THDKEOYPNFJXGWMACUIVZSRBLQ
<i>Input</i> Rotor IV	STUVWXYZABCDEFGHIJKLMNOPQR
Enkripsi Rotor IV	YBMWQUGJFVSZXONLTEIKAPRCHD

Hasil secara bertahap dari tiap huruf pada Kasus Uji-3 adalah sebagai berikut:

- Posisi Rotor: E F S  
Tahapan: **T**→T→X→Y→L→G→E→W→R→U
- Posisi Rotor: E F T  
Tahapan: **H**→H→Q→T→C→U→J→T→D→**D**
- Posisi Rotor: E F U  
Tahapan: **I**→I→A→D→B→R→C→G→D→**D**
- Posisi Rotor: E F V  
Tahapan: **S**→P→J→H→T→Z→P→C→T→**T**
- Posisi Rotor: E F W  
Tahapan: **X**→X→U→Z→P→I→A→R→L→**Z**
- Posisi Rotor: E F X  
Tahapan: **I**→I→I→O→V→W→R→H→N→**N**
- Posisi Rotor: E F Y  
Tahapan: **S**→P→T→J→U→C→T→Q→L→**Z**
- Posisi Rotor: E F Z  
Tahapan: **X**→X→N→U→Q→E→K→P→K→**K**
- Posisi Rotor: E F A  
Tahapan: **A**→A→J→H→T→Z→P→C→G→**G**
- Posisi Rotor: E F B  
Tahapan: **N**→N→W→E→G→L→Y→X→Q→**Q**

- Posisi Rotor: E F C  
Tahapan: **X**→X→Z→N→D→H→B→M→H→**H**
- Posisi Rotor: E F D  
Tahapan: **E**→E→K→X→Y→A→I→O→K→**K**
- Posisi Rotor: E F E  
Tahapan: **X**→X→H→R→W→V→O→I→S→**P**
- Posisi Rotor: E F F  
Tahapan: **A**→A→A→D→B→R→C→G→W→**W**
- Posisi Rotor: E F G  
Tahapan: **M**→M→N→U→Q→E→K→P→C→**C**
- Posisi Rotor: E F H  
Tahapan: **P**→S→U→Z→P→I→A→R→K→**K**
- Posisi Rotor: E F I  
Tahapan: **L**→Z→F→G→X→J→F→Y→Q→**Q**
- Posisi Rotor: E F J  
Tahapan: **E**→E→I→O→V→W→R→H→K→**K**
- Posisi Rotor: E F K  
Tahapan: **X**→X→D→S→M→O→L→L→Y→**Y**
- Posisi Rotor: E F L  
Tahapan: **I**→I→F→G→X→J→F→Y→P→**S**
- Posisi Rotor: E F M  
Tahapan: **N**→N→P→K→E→Q→U→N→D→**D**
- Posisi Rotor: E F N  
Tahapan: **P**→S→S→Q→N→K→W→E→A→**A**
- Posisi Rotor: E F O  
Tahapan: **U**→R→R→A→I→P→Z→U→H→**H**
- Posisi Rotor: E F P  
Tahapan: **T**→T→G→C→R→B→D→A→W→**W**

Hasil enkripsi atau *ciphertext* pada Kasus Uji-3 adalah **UDDTZ NZKGQ HKPWC KQKYS DAHW**.



#### 5.2.1.4. Performa Uji Coba Lokal

Performa dari uji lokal dinilai menggunakan contoh *test case* dari situs SPOJ. Uji performa ini dilakukan sebanyak 10 kali dan data yang diambil adalah waktu jalannya program. Gambar 5.2 adalah hasil eksekusi program pada komputer lokal.

```

Execution Time : 0.024
Execution Time : 0.021
Execution Time : 0.020
Execution Time : 0.031
Execution Time : 0.073
Execution Time : 0.030
Execution Time : 0.005
Execution Time : 0.030
Execution Time : 0.035
Execution Time : 0.030

```

**Gambar 5.2** Hasil Uji Coba pada Komputer Lokal

**Tabel 5.9** Hasil Uji Coba 10 Kali pada Komputer Lokal

Percobaan Ke-	Waktu (detik)
1	0,024
2	0,021
3	0,020
4	0,031
5	0,073
6	0,030
7	0,005
8	0,030
9	0,035
10	0,030

Rata – rata waktu eksekusi program pada uji coba di komputer lokal adalah 0.029 detik atau 29 milidetik. Grafik dapat dilihat pada Gambar 5.5.

### 5.2.2 Uji Coba Kebenaran pada Situs Daring SPOJ

Uji coba kebenaran program dilakukan dengan cara mengirimkan kode sumber ke situs penilaian *Sphere Online Judge*. Sistem pada situs penilaian tersebut akan menjalankan program yang dikirim oleh pengguna dan hasilnya kemudian akan dicocokkan dengan *file* keluaran yang telah disediakan oleh pembuat soal.

26104521	2020-06-08 13:49:37	dlt	1035	0.00	4.5M	C
----------	------------------------	-----	------	------	------	---

**Gambar 5.3** Hasil Uji Kebenaran pada situs SPOJ

Pada Gambar 5.3 dapat dilihat bahwa solusi diterima dengan skor 1035. Skor tersebut merupakan nilai besarnya *file* dari solusi yang diserahkan kepada situs daring SPOJ. Waktu yang dibutuhkan program adalah 0,0 detik dan memori yang dibutuhkan adalah 4,5 MB. Hasil uji coba di atas membuktikan bahwa implementasi yang dilakukan telah berhasil menyelesaikan permasalahan dengan batasan-batasan yang telah ditetapkan.

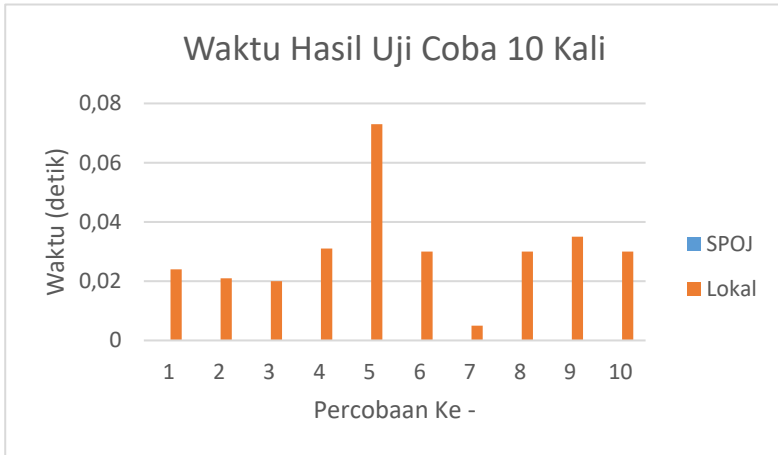
Untuk menilai kinerja dari implementasi yang telah dilakukan, pengiriman kode implementasi ke situs daring SPOJ dilaksanakan sebanyak 10 kali. Hasil dari pelaksanaannya dapat dilihat pada Gambar 5.4.

26196912	<input type="checkbox"/>	2020-06-25 18:47:07	Enigma Machine	1035 edit ideone it	0,00	4,6M	C
26196911	<input type="checkbox"/>	2020-06-25 18:46:55	Enigma Machine	1035 edit ideone it	0,00	4,4M	C
26196908	<input type="checkbox"/>	2020-06-25 18:46:48	Enigma Machine	1035 edit ideone it	0,00	4,7M	C
26196906	<input type="checkbox"/>	2020-06-25 18:46:40	Enigma Machine	1035 edit ideone it	0,00	4,7M	C
26196905	<input type="checkbox"/>	2020-06-25 18:46:31	Enigma Machine	1035 edit ideone it	0,00	4,7M	C
26196903	<input type="checkbox"/>	2020-06-25 18:46:23	Enigma Machine	1035 edit ideone it	0,00	4,6M	C
26196900	<input type="checkbox"/>	2020-06-25 18:46:18	Enigma Machine	1035 edit ideone it	0,00	4,8M	C
26196899	<input type="checkbox"/>	2020-06-25 18:46:09	Enigma Machine	1035 edit ideone it	0,00	4,6M	C
26196896	<input type="checkbox"/>	2020-06-25 18:45:58	Enigma Machine	1035 edit ideone it	0,00	4,5M	C
26196895	<input type="checkbox"/>	2020-06-25 18:45:49	Enigma Machine	1035 edit ideone it	0,00	4,5M	C

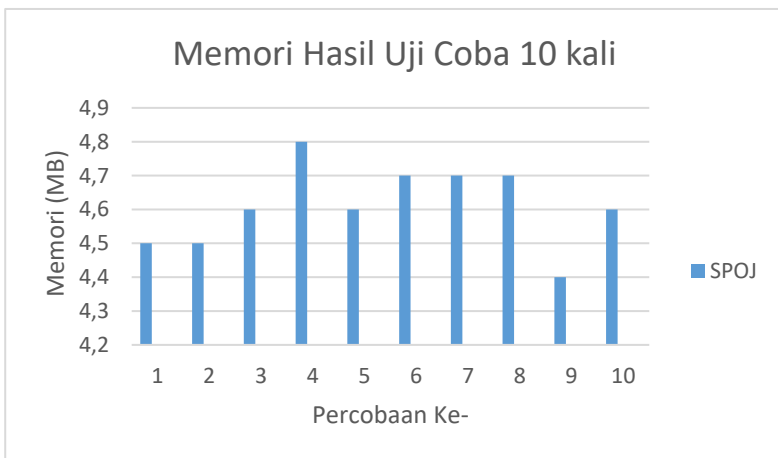
**Gambar 5.4** Hasil Uji Kebenaraan pada Situs SPOJ  
Sebanyak 10 kali.

**Tabel 5.10** Hasil Uji Coba 10 Kali pada SPOJ

Percobaan Ke -	Waktu (detik)	Memori (MB)
1	0,0	4,5
2	0,0	4,5
3	0,0	4,6
4	0,0	4,8
5	0,0	4,6
6	0,0	4,7
7	0,0	4,7
8	0,0	4,7
9	0,0	4,4
10	0,0	4,6



**Gambar 5.5** Grafik Waktu Uji Coba 10 Kali pada SPOJ dan Komputer Lokal



**Gambar 5.6** Grafik Memori Uji Coba 10 Kali pada SPOJ

Dari 10 kali uji coba yang dilakukan pada situs daring SPOJ dapat disimpulkan bahwa rata – rata waktu untuk program dari implementasi yang dilakukan adalah 0,0 detik dan rata – rata memori adalah 4,61 MB, sedangkan waktu eksekusi untuk komputer lokal memiliki rata – rata 0.029 detik.

*(Halaman ini sengaja dikosongkan)*

## **BAB VI**

### **KESIMPULAN DAN SARAN**

#### **6.1 Kesimpulan**

Dalam pengerjaan tugas akhir ini setelah melalui tahap perancangan aplikasi, implementasi metode, serta uji coba diperoleh kesimpulan sebagai berikut:

1. Permasalahan pada kasus penyandian mesin Enigma M3 telah berhasil diselesaikan berdasarkan penjelasan dan rangkaian tahapan mesin Enigma M3.
2. Ada dua pendekatan yang dapat digunakan untuk menyelesaikan kasus penyandian mesin Enigma M3 pada bagian rotor di mana pendekatan pertama dilakukan dengan menjadikan hasil enkripsi pada tiap rotor berubah berdasarkan pengaturan cincin serta memperhitungkan posisi antar rotornya dan pendekatan kedua adalah dengan memandang selisih antara posisi rotor dan pengaturan cincin yang mengakibatkan selisih untuk masukan dan hasil enkripsi.
3. Waktu dan penggunaan memori yang diperlukan untuk menyelesaikan permasalahan penyandian pada mesin Enigma M3 pada daring SPOJ adalah 0,0 detik dan memori sebesar 4,61 MB.

#### **6.2 Saran**

Pada tugas akhir ini, tentu tidak terlepas daripada kekurangan serta hal-hal penting yang dapat menjadi pertimbangan kelak. Berikut saran yang dapat diambil dari tugas akhir ini:

1. Implementasi untuk menyelesaikan kasus penyandian mesin Enigma M3 dapat dikembangkan lagi sehingga memiliki ukuran *file* kode yang lebih kecil.

*(Halaman ini sengaja dikosongkan)*



## DAFTAR PUSTAKA

- [1] Jander, "SPOJ.com - Problem ENIGMAS," Sphere Research Lab, 2010. [Online]. Available: <https://www.spoj.com/problems/ENIGMAS/>. [Accessed 15 June 2020].
- [2] M. Rouse, "What is Encryption and How Does It Work?," TechTarget, April 2020. [Online]. Available: <https://searchsecurity.techtarget.com/definition/encryption>. [Accessed 15 June 2020].
- [3] Wikipedia, "Substitution cipher - Wikipedia," Wikipedia, 29 April 2020. [Online]. Available: [https://en.wikipedia.org/wiki/Substitution\\_cipher](https://en.wikipedia.org/wiki/Substitution_cipher). [Accessed 15 June 2020].
- [4] J. Lyons, "Practical Cryptography," [Online]. Available: <http://practicalcryptography.com/ciphers/caesar-cipher/>. [Accessed 15 June 2020].
- [5] T. Sale, "The Enigma Cipher Machine," Rich Sale Limited, [Online]. Available: <http://www.codesandciphers.org.uk/enigma/index.htm>. [Accessed 15 June 2020].

- [6] D. Rijmenants, "Enigma Details," [Online]. Available: <http://users.telenet.be/d.rijmenants/en/enigmatech.htm>. [Accessed 15 June 2020].
- [7] Stanford, "29A-CryptographyChapter," [Online]. Available: <https://stanford.edu/class/archive/cs/cs106a/cs106a.1164/handouts/29A-CryptographyChapter.pdf>. [Accessed 15 June 2020].

## BIODATA PENULIS



**Daniel Lumbantobing**, lahir di Medan pada tanggal 16 Juni 1998. Penulis merupakan seorang mahasiswa yang sedang menempuh pendidikan di departemen Teknik Informatika, Institut Teknologi Sepuluh Nopember (ITS) Surabaya.

Memiliki beberapa hobi yaitu bermusik, berolahraga dan membaca buku. Pernah menjadi asisten dosen pada mata kuliah Struktur Data pada masa perkuliahan.

Selama masa perkuliahan penulis juga aktif dalam bidang organisasi kemahasiswaan, antara lain Staff Hubungan Masyarakat Metronome UKM Musik ITS pada tahun pertama perkuliahan. Tahun kedua aktif menjadi Staff Departemen Hubungan Luar Himpunan Mahasiswa Teknik Komputer (HMTC), Staff Pembinaan Pembinaan Kerohanian Mahasiswa Baru Kristen PMK ITS, dan Wakil Pengurus Harian Schematics ITS 2017 bidang Hubungan Masyarakat. Tahun ketiga aktif menjadi *Steering Comitee* (SC) PKMBK, dan Pengurus Harian Schematics 2018 bidang Hubungan Masyarakat. Pada tahun keempat perkuliahan penulis diamanahi menjadi Ketua Persekutuan Mahasiswa Kristen 2019. Selain aktif di bidang organisasi, penulis juga menjadi pembicara untuk kegiatan seminar ataupun kelas bertema branding. Penulis dapat dihubungi melalui surel di dtobing81@gmail.com.