Institut
Teknologi
Sepuluh Nopember

TUGAS AKHIR - KI141502

DESAIN DAN ANALISIS ALGORITMA PENENTUAN DIMENSI FUNGSI PERIODIK BERBASIS METODE PENJUMLAHAN EULER TOTIENT STUDI KASUS : SPOJ (22269) PERIODIC FUNCTION, TRIP 1

MICHAEL DAVE
NRP 05111440000131

Dosen Pembimbing 1
Rully Soelaiman, S.Kom.,M.Kom.

Dosen Pembimbing 2
Rizky Januar Akbar, S.Kom.,M.Eng.

DEPARTEMEN INFORMATIKA
Fakultas Teknologi Informasi dan Komunikasi
Institut Teknologi Sepuluh Nopember
Surabaya, 2018

[*Halaman ini sengaja dikosongkan*]

TUGAS AKHIR - KI141502

DESAIN DAN ANALISIS ALGORITMA PENENTUAN DIMENSI FUNGSI PERIODIK BERBASIS METODE PENJUMLAHAN EULER TOTIENT STUDI KASUS : SPOJ (22269) PERIODIC FUNCTION, TRIP 1

MICHAEL DAVE
NRP 05111440000131

Dosen Pembimbing 1
Rully Soelaiman, S.Kom.,M.Kom.

Dosen Pembimbing 2
Rizky Januar Akbar, S.Kom.,M.Eng.

DEPARTEMEN INFORMATIKA
Fakultas Teknologi Informasi dan Komunikasi
Institut Teknologi Sepuluh Nopember
Surabaya, 2018

[*Halaman ini sengaja dikosongkan*]

UNDERGRADUATE THESES - KI141502

DESIGN AND ANALYSIS ALGORITHM TO DETERMINE DIMENSION OF PERIODIC FUNCTION USING SUMMATION OF EULER TOTIENT FUNCTION CASE STUDY: SPOJ (22269) PERIODIC FUNCTION, TRIP 1

MICHAEL DAVE
NRP 05111440000131

Supervisor 1
Rully Soelaiman, S.Kom.,M.Kom.

Supervisor 2
Rizky Januar Akbar, S.Kom.,M.Eng.

INFORMATICS DEPARTMENT
Faculty of Information and Communication Technology
Institut Teknologi Sepuluh Nopember
Surabaya, 2018

[*Halaman ini sengaja dikosongkan*]

LEMBAR PENGESAHAN

DESAIN DAN ANALISIS ALGORITMA PENENTUAN DIMENSI FUNGSI PERIODIK BERBASIS METODE PENJUMLAHAN EULER TOTIENT STUDI KASUS : SPOJ (22269) PERIODIC FUNCTION, TRIP 1

TUGAS AKHIR

Diajukan Guna Memenuhi Salah Satu Syarat
Memperoleh Gelar Sarjana Komputer
pada
Bidang Studi Algoritma Pemrograman
Program Studi S-1 Departemen Informatika
Fakultas Teknologi Informasi dan Komunikasi
Institut Teknologi Sepuluh Nopember

Oleh:
Michael Dave
NRP. 05111440000131

Disetujui oleh Dosen Pembimbing Tugas Akhir:

Rully Soelaiman, S.Kom.,M.Kom. .....
NIP. 197002131994021001 (Pembimbing 1)

Rizky Januar Akbar, S.Kom.,M.Eng. .....
NIP. 198701032014041001 (Pembimbing 2)

SURABAYA
JULI 2018

vii

[*Halaman ini sengaja dikosongkan*]

ABSTRAK

DESAIN DAN ANALISIS ALGORITMA PENENTUAN DIMENSI FUNGSI PERIODIK BERBASIS METODE PENJUMLAHAN EULER TOTIENT STUDI KASUS : SPOJ (22269) PERIODIC FUNCTION, TRIP 1

Nama	:	Michael Dave
NRP	:	05111440000131
Departemen	:	Departemen Informatika, Fakultas Teknologi Informasi dan Komunikasi, ITS
Pembimbing I	:	Rully Soelaiman, S.Kom.,M.Kom.
Pembimbing II	:	Rizky Januar Akbar, S.Kom.,M.Eng.

Abstrak

Permasalahan pencarian dimensi fungsi periodik merupakan sebuah permasalahan yang ada dan tergambarkan pada situs SPOJ dengan kode soal PERIOD1. Permasalahan ini mencakup pencarian rank dari keluarga fungsi periodik yang memiliki periode maksimal tertentu. Tujuan dari penelitian ini adalah mendesain serta mengimplementasikan algoritma untuk menentukan dimensi dari keluarga fungsi periodik yang demikian dan juga dapat dibuktikan dengan menyelesaikan permasalahan pada studi kasus SPOJ PERIOD1. Algoritma yang dibuat perlu efektif karena batasan waktu 1 detik sedangkan niliai periode maksimal yang diizinkan adalah 10^8.

Penyelesaian permasalahan ini terbagi dalam 2 domain utama yaitu domain konseptual dan domain komputasional. Domain konseptual bergerak dari fungsi periodik yang perlu direpresentasikan sebagai vektor dan penggabungannya membentuk matriks. Ma-

triks yang terbentuk memiliki attribut yaitu rank matriks. Salah satu syarat pencarian rank matriks adalah sifat bebas linear yang berdasarkan hasil observasi memiliki keterkaitan dengan konsep pada teori bilangan yang dikenal dengan fungsi Euler Totient. Konsep Euler Totient inilah yang nantinya akan didesain dan diimplementasikan sedemikian rupa sehingga dapat menjadi solusi penentuan dimensi keluarga fungsi periodik.

Hasil dari uji coba dan pembuktian kebenarannya menyatakan bahwa penjumlahan Euler Totient dengan pengoptimalan menggunakan sifat faktor bilangan dapat menjadi solusi penyelesaian permasalahan kasus PERIOD1 pada SPOJ serta menjadi solusi algoritma penentuan dimensi dari keluarga fungsi periodik. Algoritma yang diimplementasikan memiliki kompleksitas $O(\sqrt{(N)})$ dan dapat menyelesaikan permasalahan studi kasus SPOJ PERIOD1 dalam rata-rata waktu 0.21 ms.

Kata Kunci: dimensi, Euler Totient, fungsi periodik, matriks, rank matriks, Sieve of Eratosthenes, vektor

ABSTRACT

DESIGN AND ANALYSIS ALGORITHM TO DETERMINE DIMENSION OF PERIODIC FUNCTION USING SUMMATION OF EULER TOTIENT FUNCTION CASE STUDY: SPOJ (22269) PERIODIC FUNCTION, TRIP 1

Name	:	Michael Dave
Student ID	:	05111440000131
Department	:	Informatics Department, Faculty of Information and Communication Technology, ITS
Supervisor I	:	Rully Soelaiman, S.Kom.,M.Kom.
Supervisor II	:	Rizky Januar Akbar, S.Kom.,M.Eng.

Abstract

A problem about finding dimension of a periodic function can be found in SPOJ with problem code PERIOD1. This problem is about finding rank of the family of periodic function that have a maximum value of a period. The purpose of this observation were designing and implementing an algorithm to determine the dimension of the family of periodic function which can be shown by solving the problem of SPOJ PERIOD1. The algortihm should be effective in case of 1 second for the time limit of computation while the maximum allowable value of period is 10^8.

The solution of this problem consist 2 domain which is conceptual domain and computational domain. The conceptual domain start from the periodic function that represented as a vector and its concatenation to form a matrix. The matrix has an attribute called rank matrix. One of the condition of finding the rank matrix is linearly independent which is in this observation sum up to have a

corelation with a number theory concept called Euler Totient. This Euler Totient function should be designed and implemented in such a way that it may be the solution to determine the dimension of this periodic function's family.

The result from the observation show that the sum of the Euler Totient and its optimization using a correlation with number factor can be a solution for the PERIOD1's problem which can also be the algorithm to determine the dimension of a periodic function's family. The algorithm has a $O(\sqrt{(N)})$ *and can solved the problem in the average of 0.21 ms.*

Keywords: dimension, Euler Totient, periodic function, matrix, matrix rank, Sieve of Eratosthenes, vector

KATA PENGANTAR

Puji syukur penulis panjatkan kepada Tuhan Yesus Kristus. Atas kasih setia dan anugerahNya, penulis dapat menyelesaikan tugas akhir dan laporan akhir dalam bentuk buku ini.

Pengerjaan buku ini penulis tujukan untuk mengeksplorasi lebih mendalam topik-topik yang tidak diwadahi oleh kampus, namun banyak menarik perhatian penulis. Selain itu besar harapan penulis bahwa pengerjaan tugas akhir sekaligus pengerjaan buku ini dapat menjadi batu loncatan penulis dalam menimba ilmu yang bermanfaat.

Penulis ingin menyampaikan rasa terima kasih kepada banyak pihak yang telah membimbing, menemani dan membantu penulis selama masa pengerjaan tugas akhir maupun masa studi.

1. Ibu Yohanna Gozali, selaku ibu penulis yang senantiasa mendukung dalam berbagai hal, baik doa, nasihat maupun moral. Juga pihak yang selalu mengingatkan untuk selalu berdoa dan mengandalkan Tuhan selalu dalam mengerjakan tugas akhir.
2. Alm. Bapak Tjipta Mindarta Theodurus, selaku ayah penulis yang mendukung selalu dalam penulis menempuh masa studi di permulaan tahun perkuliahan dan memberikan teladan dalam segala hal.
3. Janet Sheila Theodurus, selaku kakak penulis yang selalu mendukung dan mengingatkan penulis untuk tekun mengerjakan tugas akhir ini.
4. Bapak Rully Soelaiman S.Kom.,M.Kom., selaku pembimbing penulis yang telah mengajarkan ilmu serta banyak hal lain, membimbing, memberikan masukan dan nasihat serta teguran-teguran yang membantu penuils akhirnya menyelesaikan pengerjaan tugas akhir ini serta menyelesaikan perkuliahan.
5. Bapak Rizky Januar Akbar, S.Kom., M.Eng., selaku pembim-

bing penulis yang telah memberikan arahan semasa pengerjaan buku tugas akhir ini.
6. Keluarga besar dari ibu dan ayah penulis yang mendukung dalam moral maupun materi selama masa studi penulis.
7. Hamba Tuhan Gereja GKA Gloria Pacar yang telah mendukung dalam moral maupun materi selama masa studi penulis.
8. Rekan-rekan kelompok kecil penulis yang telah mendukung dalam doa dan kesediaan mendengar segala ungkapan hati penulis.
9. Rekan-rekan angkatan 2014 mahasiswa Teknik Informatika dan mahasiswa Teknik Informatika ITS dari berbagai angkatan yang tidak lelah membantu penulis semasa masa studi.

Penulis menyadari bahwa buku ini jauh dari kata sempurna. Maka dari itu, penulis memohon maaf apabila terdapat salah kata maupun makna pada buku ini. Akhir kata, penulis mempersembahkan buku ini sebagai wujud nyata kontribusi penulis dalam ilmu pengetahuan dan segala kemuliaan hanya bagi Tuhan Yesus.

Surabaya, Juli 2018

Michael Dave

DAFTAR ISI

[*Halaman ini sengaja dikosongkan*]

DAFTAR GAMBAR

DAFTAR TABEL

[*Halaman ini sengaja dikosongkan*]

DAFTAR KODE SUMBER

[*Halaman ini sengaja dikosongkan*]

BAB I
PENDAHULUAN

Pada bab ini, akan dijelaskan mengenai latar belakang, rumusan masalah, batasan masalah, tujuan, metodologi pengerjaan, dan sistematika penulisan dari tugas akhir ini.

1.1. Latar Belakang

Salah satu problem yang ada pada situs Sphere Online Judge (SPOJ) adalah periodic function, trip 1 dengan kode soal PERIOD1[1]. Pada problem ini dijelaskan bahwa ada sebuah fungsi periodik[2] dari Z ke R. Sebagai contoh f adalah fungsi 3-periodik dengan f(0) = f(3) = f(6) = f(9) = 4, f(1) = f(4) = f(7) = f(10) = -6 dan f(2) = f(5) = f(8) = f(11) = 7 atau dengan notasi yang lebih sederhana sebuah fungsi periodik f dapat dituliskan sebagai f = $\left[4,-6,7\right]$. Untuk dua buah fungsi periodik dengan periodik yang integral maka perbandingan kedua buah periode dari fungsi itu akan rasional, maka dapat disimpulkan bahwa jumlah dari dua fungsi periodik adalah juga sebuah fungsi periodik. Maka dari itu himpunan fungsi-fungsi yang demikian ada dalam ruang vektor R. Fokus dari permasalahan ini adalah dimensi dari ruang vektor ini ketika periodenya dibatasi oleh sebuah nilai N[1].

Domain konseptual pada pengerjaan tugas akhir ini akan berada dalam konsep aljabar linier[3] sedangkan domain komputasional dan implementatif pada pengerjaan tugas akhir ini berfokus pada salah satu ranah dalam teori bilangan[4] yaitu Euler Totient. Fungsi Euler Totient dengan simbol $\phi(n)$ digunakan untuk mencari bilangan dari 1 hingga N yang relatif prima dengan bilangan N itu sendiri[5].

Pendekatan yang akan dilakukan adalah dengan merepresentasikan fungsi periodik menjadi vektor[6] untuk membentuk matriks[7] yang merupakan gabungan fungsi periodik. Kemudian meng-

gunakan hubungan antara bebas linear[8] dengan faktor bilangan untuk menentukan rank matriks[9] yang menghasilkan sebuah pola yang adalah fungsi Euler Totient[5]. Dengan demikian rank dari matriks tersebut adalah penjumlahan dari N fungsi Euler Totient yang menjadi keluaran yang diharapkan.

1.2. Rumusan Masalah

Permasalahan yang akan diselesaikan pada tugas akhir ini adalah sebagai berikut:

1. Bagaimana strategi penyelesaian permasalahan Periodic function, trip 1 pada situs penilaian SPOJ?
2. Bagaimana menganalisis dan membuat desain dari solusi penyelesaian permasalahan Periodic function, trip 1 pada situs penilaian SPOJ?
3. Bagaimana hasil kinerja dari solusi penyelesaian permasalahan Periodic function, trip 1 pada situs penilaian SPOJ?

1.3. Batasan Masalah

Masalah yang akan diselesaikan memiliki batasan-batasan sebagai berikut:

1. Implementasi algoritma menggunakan bahasa pemrograman C++.
2. Jumlah kasus uji dalam 1 kali percobaan maksimal 100 kasus.
3. N adalah bilangan bulat positif yang lebih dari 0 dan kurang dari atau sama dengan 10^8.
4. Batas waktu pada program dalam 1 kali percobaan di bawah 1 detik.
5. Penyimpanan yang dibutuhkan dalam 1 kali percobaan di bawah 1536 Megabyte.

1.4. Tujuan

Tujuan tugas akhir ini adalah sebagai berikut:

1. Merumuskan strategi penyelesaian permasalahan Periodic function, trip 1 pada situs penilaian SPOJ.
2. Melakukan analisis dan membuat desain solusi penyelesaian permasalahan Periodic function, trip 1 pada situs penilaian SPOJ.
3. Menguji hasil kinerja dari solusi penyelesaian permasalahan Periodic function, trip 1 pada situs penilaian SPOJ.

1.5. Metodologi

Metodologi pengerjaan yang digunakan pada tugas akhir ini memiliki beberapa tahapan. Tahapan-tahapan tersebut yaitu:

1. Penyusunan proposal
   Tahap pertama dalam proses pengerjaan Tugas Akhir ini adalah menyusun proposal Tugas Akhir. Pada proposal Tugas Akhir ini akan diajukan sebuah permasalahan optimisasi pada SPOJ Klasik Periodic Function, trip 1 yang akan diselesaikan dengan penggunaan algoritma yang berkaitan dengan permasalahan tersebut. Luaran dari penyusunan tahap ini adalah tersajinya proposal Tugas Akhir.

2. Observasi dan studi literatur
   Pada tahap pengerjaan Tugas Akhir ini, akan dilakukan studi mendalam terkait cara-cara mengoptimasi kecepatan kompiler C++, teori angka, dan algoritma algoritma yang terkait dengan permasalahan SPOJ Klasik Periodic Function, trip 1. Penulis juga melakukan banyak observasi terkait pola dari angka-angka yang terbentuk.

3. Implementasi algoritma

Implementasi algoritma adalah tahapan untuk membangun aplikasi yang digunakan untuk menyelesaikan permasalahan SPOJ Klasik Periodic Function, trip 1. Luaran dari tahapan ini adalah implementasi algoritma yang dibangun menggunakan bahasa pemrograman C++ dan menggunakan IDE Dev-C++.

4. Pengujian dan evaluasi
   Tahap pengujian dan evaluasi akan dilakukan dengan menguji program yang telah dibuat pada Online Judge SPOJ Klasik Periodic Function, trip 1. Pengujian dan evaluasi juga dilakukan dengan memberikan contoh kecil serta pengerjaan soal SPOJ yang lain yang menggunakan pendekatan yang sama. Pengujian dan Evaluasi dari algoritma penyelesaian studi kasus soal PERIOD1 akan dilakukan terus-menerus hingga mendapatkan hasil Accepted dari Online Judge SPOJ. Luaran dari tahapan ini adalah status Accepted dari SPOJ.

5. Penyusunan buku
   Pada tahapan ini penulis menyusun hasil pengerjaan tugas akhir mengikuti format penulisan tugas akhir.

1.6. Sistematika Penulisan

Sistematika laporan tugas akhir yang akan digunakan adalah sebagai berikut:

1. Bagian awal, meliputi halaman depan, halaman pengesahan, abstrak, kata pengantar, daftar isi, daftar gambar, dan daftar tabel.
2. Bagian inti, meliputi pendahuluan, tinjauan pustaka, metodologi, hasil dan pembahasan, dan kesimpulan dan saran.
3. Bagian akhir, meliputi daftar pustaka, lampiran-lampiran, dan biodata penulis.

BAB II
DASAR TEORI

Pada bab ini akan dijelaskan mengenai dasar teori yang menjadi dasar pengerjaan tugas akhir ini.

2.1. Deskripsi Permasalahan

Permasalahan yang mendasari pengerjaan Tugas Akhir ini adalah permasalahan SPOJ (Sphere Online Judge) Klasik Periodic function, trip 1, dengan kode soal "PERIOD1". Dinyatakan bahwa semisal ada sebuah f dinyatakan sebagai fungsi 3-periodik dengan f(0) = f(3) = f(6) = f(9) = 4. f(1) = f(4) = f(7) = f(10) = -6 dan f(2) = f(5) = f(8) = f(11) = 7. Dengan notasi yang lebih mudah maka kita bisa mendefinisikan f dengan $\lceil$ 4, -6, 7 $\rceil$. Untuk 2 fungsi periodik dengan periode yang integral, perbandingan antar periodenya pasti bilangan rasional juga. Maka dari itu dapat dipastikan bahwa penjumlahan fungsinya akan menghasilkan fungsi yang periodik juga. Himpunan semua fungsi yang demikian berada dalam ruang vektor R. Tujuan dari permasalahan ini adalah mencari dimensi dari ruang vektor tersebut dimana periodenya dibatasi oleh integer N[1].

Format masukan
Di baris pertama sebuah integer T, yang menyatakan jumlah kasus uji, T baris berikutnya berisi sebuah integer N. Pikirkan seluruh keluarga dari fungsi dengan n sebagai periodenya untuk n dalam $\lceil$1..N$\rceil$.

Format keluaran
Rank dari keluarga fungsi tersebut.

Contoh masukan

```
3
2
3
4
```

Contoh keluaran

```
2
4
6
```

Gambar 2.1: Contoh Masukan dan Keluaran soal PERIOD1

2.2. Deskripsi Umum

Pada subbab ini akan dijelaskan mengenai deskripsi-deskripsi umum yang terdapat pada tugas akhir ini.

2.2.1. Fungsi Periodik

2.2.1.1. Definisi Fungsi Periodik

Sebuah fungsi periodik[2] dalam tugas akhir ini adalah sebuah fungsi dimana fungsi tersebut memiliki nilai P yang disebut periode yang memenuhi definisi f(x+P) = f(x). Nilai dari P dapat dipastikan sebuah nilai yang integral. Fungsi periodik dalam tugas akhir ini dituliskan sebagai $[A_1,A_2,...,A_P]$. Contohnya adalah sebuah fungsi periodik dengan periode 3 yaitu $[4,6,-7]$ yang berarti f(0) = f(3) = f(6) = f(9) bernilai 4, f(1) = f(4) = f(7) = f(10) bernilai 6 dan f(2) = f(5) = f(8) = f(11) bernilai -7.

2.2.1.2. Operasi Fungsi Periodik

Fungsi periodik dengan periode yang integral memiliki sifat penjumlahan dan perkalian. Penjumlahan berarti jumlah dari 2 buah fungsi periodik adalah fungsi periodik juga. Sedangkan perkalian sebuah fungsi periodik dengan skalar menghasilkan sebuah fungsi periodik dengan periode yang sama. Contohnya adalah sebuah fungsi 3-periodik dan fungsi 2-periodik jika dijumlahkan akan menghasilkan fungsi 6-periodik. Hal ini karena jumlah 2 fungsi periodik akan menghasilkan fungsi periodik dengan periode yang adalah angka terkecil yang dapat dibagi oleh ke 2 nilai periode yang ada. Karena 2 dan 3 habis membagi angka 6 maka periode fungsi periodik yang dihasilkan adalah 6. Fungsi 3-periodik tersebut misalnya adalah $\left[4, 6, -7\right]$ dan fungsi 2-periodiknya adalah $\left[1,2\right]$ maka hasil penjumlahan 2 fungsi periodik ini adalah $\left[4,6,-7,4,6,-7\right]$ + $\left[1,2,1,2,1,2\right]$ = $\left[5,8,-6,6,7,-5\right]$ yang artinya adalah fungsi 6-periodik.

2.2.1.3. Representasi Fungsi Periodik dengan Penjumlahan Vektor

Pada tugas akhir ini sebuah fungsi periodik dapat direpresentasikan dalam bentuk penjumlahan vektor[6]. Untuk sebuah fungsi n-periodik maka besar vektornya adalah n dimana n adalah periode. Contohnya, sebuah fungsi 3-periodik $\left[4,6,-7\right]$ memiliki representasi sebagai penjumlahan 3 vektor sebagai berikut:

$$4\begin{bmatrix}1\\0\\0\end{bmatrix} +6\begin{bmatrix}0\\1\\0\end{bmatrix} -7\begin{bmatrix}0\\0\\1\end{bmatrix} = \begin{bmatrix}4\\6\\-7\end{bmatrix}$$

Kemudian dapat juga direpresentasikan sebagai matriks

$$\begin{bmatrix}1 & 0 & 0\\0 & 1 & 0\\0 & 0 & 1\end{bmatrix}$$

Dengan mengabaikan skalar dari masing-masing vektor setiap fungsi n-periodik dapat direpresentasikan sebagai n x n matriks A dimana $A_{i,j}$ = 1 untuk setiap $i = j$. Contohnya untuk fungsi 5-periodik maka representasi matriksnya adalah

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

2.2.2. Vektor

2.2.2.1. Definisi Vektor

Vektor dalam tugas ini adalah sekumpulan angka yang direpresentasikan dengan $[A_1,A_2,...,A_P]$ sebagai sebuah kolom atau baris dalam sebuah matriks[7].

2.2.2.2. Ruang Vektor

Ruang Vektor[10] dalam dalam tugas ini direpresentasikan sebagai sebuah matriks A yang didapat dengan menggabungkan beberapa fungsi periodik yang telah direpresentasikan dalam bentuk matriks juga.
Contohnya : 2 buah fungsi periodik terdiri dari fungsi 2-periodik [3,-5] yaitu

$$3 \begin{bmatrix} 1 \\ 0 \end{bmatrix} - 5 \begin{bmatrix} 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 3 \\ -5 \end{bmatrix}$$

dan fungsi 3-periodik [4,6,-7] yaitu

$$4 \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} + 6 \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} - 7 \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 4 \\ 6 \\ -7 \end{bmatrix}$$

Representasi matriksnya adalah

$$\begin{bmatrix}1 & 0\\0 & 1\end{bmatrix} \text{ dan } \begin{bmatrix}1 & 0 & 0\\0 & 1 & 0\\0 & 0 & 1\end{bmatrix}$$

Untuk menggabungkan menjadi sebuah ruang vektor[10] maka tiap matriks di perpanjang ke bawah hingga jumlah baris pada matriks tersebut adalah benilai angka terkecil yang dapat dibagi oleh ke 2 periode yang digabungkan. Sehingga karena 6 adalah angka terkecil yang dapat dibagi oleh 2 dan 3 maka matriks fungsi 2-periodik dan 3-periodik menjadi

$$\begin{bmatrix}1 & 0\\0 & 1\\1 & 0\\0 & 1\\1 & 0\\0 & 1\end{bmatrix} \text{ dan } \begin{bmatrix}1 & 0 & 0\\0 & 1 & 0\\0 & 0 & 1\\1 & 0 & 0\\0 & 1 & 0\\0 & 0 & 1\end{bmatrix}$$

sehingga matriks gabungannya menjadi

$$\begin{bmatrix}1 & 0 & 1 & 0 & 0\\0 & 1 & 0 & 1 & 0\\1 & 0 & 0 & 0 & 1\\0 & 1 & 1 & 0 & 0\\1 & 0 & 0 & 1 & 0\\0 & 1 & 0 & 0 & 1\end{bmatrix}$$

2.2.2.3. Bebas Linear

Sebuah vektor dalam ruang vektor disebut bebas linear[8] apabila tiap vektor kolom tidak dapat ditulis sebagai kombinasi linear[11] dari vektor-vektor yang lain. Contohnya fungsi 3-periodik dan 1-periodik dapat digabungkan dengan cara yang telah dijelaskan sebelumnya menjadi sebuah matriks.

$$\begin{bmatrix} 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \end{bmatrix}$$

Matriks tersebut vektor-vektornya (vektor kolom) tidak bersifat bebas linear karena kolom pertama yaitu vektor

$$\begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$$

dapat dihasilkan dari penjumlahan 3 vektor yang lain yaitu

$$\begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$$

2.2.2.4. Dimensi

Dimensi[12] dari sebuah ruang vektor adalah jumlah maksimum vektor yang saling bebas linear. Dalam hal ini dimensi dari penjumlahan fungsi 1-periodik dengan 3-periodik adalah memiliki dimensi 3.

2.2.3. Matriks

2.2.3.1. Definisi Matriks

Matriks [7] adalah susunan bilangan, simbol, atau ekspresi, yang disusun dalam baris dan kolom sehingga membentuk suatu bangun persegi. Contohnya adalah matriks dari fungsi 2-periodik yaitu

$$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

2.2.3.2. Rank Matriks

Rank Matriks[9] dari sebuah matriks A adalah dimensi dari ruang vektor yang dihasilkan oleh vektor kolom /vektor baris dari matriks tersebut, atau dengan kata lain jumlah maksimal vektor kolom/baris yang saling bebas linear dari matriks A.

2.2.4. Teori Bilangan

Teori bilangan (number theory)[4] adalah teori yang mendasar dalam memahami sifat-sifat bilangan. Bilangan yang dimaksudkan adalah bilangan bulat (integer). Teori bilangan banyak digunakan untuk mempermudah perhitungan suatu persamaan atau mempercepat komputasi suatu persamaan.

2.2.4.1. Jenis Bilangan

Bilangan bulat adalah bilangan yang tidak memiliki pecahan desimal.

- **Contoh:** 8, 21, 8765, -34, 0.
- **Notasi:** Z

Yang berlawanan dengan bilangan bulat disebut bilangan riil, yaitu bilangan yang mempunyai pecahan desimal.

- **Contoh:** 8.0, 34.25, 0.02.
- **Notasi:** R

2.2.4.2. Sifat Pembagian pada Bilangan Bulat

Misalkan a dan b adalah dua buah bilangan bulat dengan syarat a $\neq$ 0.

- Kita menyatakan bahwa a habis membagi b (a divides b) jika terdapat bilangan bulat c sedemikian sehingga b = ac

- **Notasi :** a | b jika b = ac, c ϵ Z ,dan a $\neq$ 0
- **Contoh :** 4 | 12 karena 12 div 4 = 3 (bilangan bulat) atau 12 = 4 * 3.

2.2.5. Faktor Bilangan

Faktor[13] dari sebuah bilangan bulat adalah sebuah angka atau ekspresi yang membagi angka tersebut tanpa ada sisa.
Contoh : 2 habis membagi 6 maka 2 adalah faktor dari 6, demikian pula 1,3 dan 6 adalah juga faktor dari 6.

2.2.5.1. Faktor Persekutuan Terbesar

Misalkan a dan b adalah dua buah bilangan bulat tidak nol. Pembagi bersama terbesar[14] (PBB – greatest common divisor atau GCD) dari a dan b adalah bilangan bulat terbesar d sedemikian sehingga d|a dan d|b. Dalam hal ini kita nyatakan bahwa GCD(a,b) = d.

2.2.6. Bilangan Prima

Sebuah bilangan bulat positif n dinyatakan bilangan prima[15] jika n hanya memiliki 2 faktor yaitu 1 dan dirinya sendiri.
Contoh : 23 adalah bilangan prima karena ia hanya habis dibagi oleh 1 dan 23.

2.2.6.1. Teori Fundamental Aritmatika

Setiap bilangan bulat positif yang lebih besar atau sama dengan 2 dapat dinyatakan sebagai perkalian satu atau lebih bilangan prima[16]. Secara persamaan ditulis

$$n = p_1^{k_1} * p_2^{k_2} * .... * p_n^{k_n} \tag{2.1}$$

Contoh :

- 9 = 3 * 3 (2 buah faktor prima)
- 100 = 2 * 2 * 5 * 5 (4 buah faktor prima)
- 13 = 13 (1 buah faktor prima)

2.2.6.2. Mencari Bilangan Prima

Pencarian bilangan prima dapat dilakukan dengan berbagai macam metode. Salah satu metode yang sering digunakan adalah Sieve of Eratosthenes[17]. Metode Sieve of Eratosthenes selanjutnya akan disingkat menjadi SoE. Tahap pencarian bilangan prima kurang dari N dengan metode SoE adalah sebagai berikut:

1. Mendaftarkan seluruh bilangan dari 2 hingga N.
2. Pada tahap pertama p bernilai 2 yaitu bilangan prima terkecil.
3. Tandai angka mulai 2p,3p,4p dan seterusnya (dengan penambahan sebesar p) hingga N.
4. Cari angka terkecil yang lebih besar dari p dan belum ditandai pada daftar bilangan, dan pilih sebagai nilai p lalu ulangi langkah ke 3, jika tidak ada maka hentikan pencarian.
5. Bilangan yang tidak ditandai pada akhir pencarian adalah bilangan prima kurang dari N.

2.2.6.3. Relatif Prima

Dua buah bilangan bulat a dan b dikatakan relatif prima (coprime)[18] jika GCD(a, b) = 1.
Contoh:

- 20 dan 3 relatif prima sebab GCD(20, 3) = 1.
- 7 dan 11 relatif prima karena GCD(7, 11) = 1.
- 20 dan 5 tidak relatif prima sebab GCD(20, 5) = 5.

2.2.7. Fungsi Euler Totient

Fungsi Euler Totient adalah fungsi yang menghitung bilangan dari 1 hingga n yang relatif prima dengan bilangan n itu sendiri[5]. Atau dengan notasi matematika dapat dituliskan sebagai jumlah angka k dimana 1 $\leq$ k $\leq$ n, dimana GCD dari k dan n adalah 1 atau GCD(k,n) = 1.

2.2.7.1. Euler Product Formula (Perkalian Euler)

Euler Product Formula atau Perkalian Euler adalah sebuah fungsi matematika yaitu

$$\phi(n) = n \prod_{p|n}^{n} \tag{2.2}$$

dimana p adalah semua nilai dari 1 hingga n yang habis membagi n.

2.2.7.2. Sifat Multiplikatif pada fungsi Euler Totient

Sifat multiplikatif yaitu jika GCD(m,n) = 1 maka

$$\phi(n * m) = \phi(n) * \phi(m) \tag{2.3}$$

2.2.7.3. Nilai pemangkatan prima pada fungsi Euler Totient

Adalah cara perhitungan dari fungsi Euler Totient untuk p yang bernilai prima. Yaitu memenuhi persamaan:

$$\phi(p^k) = p^{k-1}(p-1) \tag{2.4}$$

2.2.7.4. Perhitungan Fungsi Euler Totient

Seperti sudah disampaikan sebelumnya, mengacu pada bagian 2.2.6.1, setiap bilangan bulat positif lebih dari 1, pasti merupakan perkalian bilangan prima dengan pangkat tertentu dengan bilangan prima yang lain[16]. Dan mengacu pada bagian 2.2.7.2 maka sebuah ϕ bilangan bulat adalah hasil perkalian antar ϕ yang lain. Sehingga dapat dituliskan sebagai berikut berdasarkan teori fundamental aritmatika:

$$\phi(n) = \phi(p_1^{k_1} * p_2^{k_2} * .... * p_n^{k_n}) \tag{2.5}$$

dimana p_i adalah bilangan prima dengan pangkat k_i, dan berdasarkan sifat multiplikatif pada fungsi euler totient maka

$$\phi(p_1^{k_1} * p_2^{k_2} * .... * p_n^{k_n}) = \phi(p_1^{k_1}) * \phi(p_2^{k_2}) * \cdots * \phi(p_n^{k_n}) \tag{2.6}$$

Untuk bilangan prima maka nilai euler totientnya adalah bilangan prima itu sendiri dikurangi dengan 1 dengan P adalah himpunan bilangan prima atau dengan notasi matematika dapat dituliskan sebagai:

$$\phi(p) = p - 1, p \in P \tag{2.7}$$

2.2.8. Penjumlahan dengan Pendekatan Faktor

Seperti telah dijelaskan sebelumnya pada bagian 2.2.7, fungsi euler totient adalah fungsi yang menghitung bilangan dari 1 hingga n yang relatif prima dengan bilangan n itu sendiri[5]. Dalam penjelasan lebih lanjut $\phi(n)$ adalah jumlah bilangan bulat positif yang memenuhi GCD(k,n) =1 untuk 1$\leq$ k$\leq$ n. Dengan demikian maka untuk jumlah semua pasangan a,b yang memenuhi 1$\leq$ a $\leq$ b $\leq$ n dan GCD(a,b)=1 adalah merupakan $\phi(n)$. Lebih lanjut, untuk sembarang bilangan a,b (1$\leq$ a $\leq$ b $\leq$ n) ada pasangan a,b yang memenuhi GCD(a,b) = p, untuk semua p dari 2 hingga n. Sehingga jumlah seluruh pasangan bilangan a,b yang dapat dibentuk dari kondisi 1$\leq$

a $\leq$ b $\leq$ n adalah penjumlahan dari pasangan a,b yang memenuhi GCD(a,b) = p, untuk semua p dari 1 hingga n. Selanjutnya jumlah pasangan bilangan a,b yang memenuhi GCD(a,b) = p akan ditulis dengan simbol $S_p(n)$ Contohnya adalah sebagai berikut:

Untuk N=3 maka ada beberapa pasangan a,b yang mungkin terbentuk yang memenuhi kondisi GCD(a,b) = p, untuk semua p dari 1 hingga n. Hal ini tergambarkan seperti pada tabel 2.1.

Tabel 2.1: Kiri: daftar n k, Kanan : daftar n, ϕ(n) dan penjumlahan N Euler Totient

a	b	GCD(a,b)
1	1	1
1	2	1
1	3	1
2	2	2
2	3	1
3	3	3

p	$S_p(3)$
1	4
2	1
3	1

Dari tabel 2.1 dapat diketahui bahwa untuk N=3 maka ada n pasang a,b dimana a bernilai 1, ada n-1 pasang a,b dimana a bernilai 2 hingga 1 pasang a,b dimana a bernilai n. Sehingga total semua pasangan a,b yang memenuhi 1$\leq$ a $\leq$ b $\leq$ n adalah x + (x-1) + (x-2) + . . . + 1. Mengikuti bentuk deret matematika maka rumusnya menjadi n * (n+1)/2. Selanjutnya sesuai yang tertera pada tabel, $S_p(3)$ dimana p = 1 menyatakan jumlah semua pasangan bilangan a,b yang memenuhi 1$\leq$ a $\leq$ b $\leq$ 3 dan memiliki GCD (a,b) =1. Hal ini menunjukan bahwa $S_p(n)$ untuk p=1 menyatakan jumlah dari n fungsi euler totient yang berurutan. $S_1(3)$ berarti adalah $\phi(1) + \phi(2) + \phi(3)$. Dengan pendekatan ini maka untuk mencari n nilai

dari fungsi euler totient, dapat disimpulkan sebuah persamaan:

$$S_1(n) = (\frac{n*(n+1)}{2}) - \sum_{i=2}^{N=n} S_i(n) \tag{2.8}$$

Selanjutnya dari hasil observasi dan nanti akan dibuktikan, maka $S_1(n/2)$ adalah menyatakan jumlah pasangan bilangan yang memenuhi GCD(a,b) = 2 , $S_1(n/3)$ adalah menyatakan jumlah pasangan bilangan yang memenuhi GCD(a,b) = 3 dan seterusnya hingga $S_1(n/n)$ yang menyatakan jumlah pasangan bilangan yang memenuhi GCD(a,b) = n. Dalam persamaan dapat dituliskan sebagai berikut

$$S_1(n) = (\frac{n*(n+1)}{2}) - (S_1(\frac{n}{2}) + S_1(\frac{n}{3}) + \cdots + S_1(\frac{n}{n})) \tag{2.9}$$

2.2.9. Contoh Kasus Penggunaan Euler Totient

Pada bagian ini akan diberikan sebuah contoh kasus, untuk dapat lebih menjelaskan mengenai definisi serta pemanfaatan dari penjumlahan fungsi Euler Totient serta dapat lebih menjelaskan bagian sebelumnya (bagian 2.2.8).

2.2.9.1. Definisi Permasalahan

Soal yang akan diberikan berasal dari Sphere Online Judge dengan judul Candy Prime (34484) dengan kode soal CANPR[19]. Berikut deskripsi dari soal tersebut.

Seorang anak memiliki N buah permen dengan masing-masing permen memiliki label yang berbeda yaitu A_i untuk i adalah dari 1 hingga N. Anak ini ingin membagikan sepasang permen dengan syarat setiap pasang permen memiliki label A_i dan A_j dimana i dan j memiliki GCD tertentu. Si anak memiliki sebuah bilangan L yang merupakan GCD yang ia harapkan. Tentukan jumlah pasangan permen yang dapat dibentuk yang memenuhi kondisi tersebut.

Diberikan N dan L tentukan jumlah pasangan yang mungkin. Untuk kasus ini maka pasangan permen (x,y) tidak sama dengan (y,x). Pada kasus ini diperbolehkan jika x dan y adalah bilangan yang sama. Gambar 2.2 dan 2.3 adalah contoh uji kasus. Dibaris pertama diberikan sebuah T yang merupakan jumlah uji kasus. T baris berikutnya berupa pasangan bilangan N dan L. N dan L memenuhi ($1 \leq$ N,L $\leq 10^6$). Untuk penjelasan uji kasus ke 2 pada gambar 2.2 dan

Contoh masukan

```
2
5 1
3 2
```

Gambar 2.2: Contoh Masukan CANPR

2.3, dari 1 hingga 3 yang dapat berpasangan dengan GCD=2 adalah 2,2 sehingga jumlah cara berpasangan yang ada adalah 1.

2.2.9.2. Strategi Penyelesaian

Pada bagian ini akan dijelaskan mengenai strategi peneyelesaian dari permasalahan SPOJ dengan kode soal CANPR.

1. Melakukan pre-komputasi Euler Totient.

Contoh keluaran

```
19
1
```

Gambar 2.3: Contoh Keluaran CANPR

2. Melakukan pre-komputasi Penjumlahan Euler Totient.
3. Mencari pasangan bilangan yang mungkin.

2.2.9.2.1. Melakukan pre-komputasi Euler Totient

Pada bagian ini akan dijelaskan mengenai pre-komputasi dari fungsi Euler Totient. Pre-komputasi ini melibatkan penggunaan ide dari metode pencarian prima yaitu Sieve of Eratosthenes seperti yang dijelaskan pada bagian 2.2.6.2. Jika pada Sieve of Eratosthenes[17] hasil bilangan prima kurang dari N didapatkan setelah semua proses pencarian berakhir, maka pada desain fungsi untuk perhitungan nilai fungsi Euler Totient ini, tiap bilangan prima yang ada langsung diproses untuk pencarian nilai euler totientnya. Dan untuk bilangan bukan prima yang adalah kelipatan dari bilangan prima yang sedang diproses maka dapat dicari nilai euler totientnya dengan menggunakan sifat multipikatif dari fungsi euler totient seperti pada bagian 2.2.7.3.

2.2.9.2.2. Melakukan pre-komputasi Penjumlahan Euler Totient

Pada bagian ini akan dijelaskan mengenai penjumlahan N buah fungsi Euler Totient. Hal ini dilakukan dengan tujuan mendapatkan semua pasangan bilangan yang memiliki GCD=1[14] dengan batas N buah bilangan. Penjumlahan dilakukan dengan gabungan metode iterasi dengan menjumlahkan jumlah yang sebelumnya dengan nilai euler totient saat ini dan dengan penjumlahan dengan pendekatan faktor seperti pada bagian 2.2.8.

2.2.9.2.3. Mencari pasangan bilangan yang mungkin

Pada bagian ini akan dijelaskan rumus yang mengacu pada pembahasan sebelumnya pada bagian 2.2.8. Dari rumus yang didapatkan melalui observasi lebih lanjut maka diketahui bahwa sebuah

penjumlahan N fungsi Euler Totient pertama dapat digunakan untuk mencari jumlah pasangan bilangan yang berada di antara 1 hingga N dan memliki GCD tertentu yang $\neq 1$. Hal ini kemudian digunakan untuk menghitung jumlah cara pasangan permen dapat dibagikan. Untuk hasil uji coba kebenaran akan dibahas lebih lanjut di bagian 5.3.

2.3. Strategi Penyelesaian

Pada subbab ini akan dipaparkan mengenai strategi penyelesaian dari permasalahan pada situs SPOJ dengan kode soal PERIOD1. Secara singkat penyelesaian masalah dari PERIOD1[1] terbagi menjadi beberapa tahapan yaitu:

1. Representasi Fungsi Periodik sebagai Penjumlahan Vektor.
2. Pengabungan Matriks untuk Membentuk Ruang Vektor.
3. Pencarian Rank Matriks pada Matriks.
4. Hubungan Antar Bebas Linear dengan Faktor Bilangan.
5. Pencarian Hasil dari Fungsi Euler Totient untuk nilai tertentu.
6. Penjumlahan Fungsi Euler Totient hingga N.
7. Optimasi Penjumlahan Fungsi Euler Totient.

Sebagai contoh pada subbab ini akan dicari nilai dari dimensi[12] pada ruang vektor dengan fungsi periodik terbesar adalah 4 atau N=4. Titik utama dari permasalahan ini adalah membawa alur berpikir dari fungsi periodik menuju representasi matriks lalu kepada teori bilangan yaitu fungsi euler totient dan penjumlahannya karena tidak memungkinkan menggunakan rumus perhitungan untuk membangun matriks yang menggabungkan periode dari fungsi dengan periode terbesar adalah $< 10^8$.

2.3.1. Representasi fungsi periodik sebagai penjumlahan vektor

Seperti dijelaskan sebelumnya di bagian 2.2.1.3, sebuah fungsi periodik dalam permasalahan PERIOD1 ini dapat direpresentasi-

kan sebagai penjumlahan 1 atau lebih vektor. Tiap fungsi periodik $\left[A_1,A_2,...,A_N\right]$ dapat dituliskan sebagai:

$$A_1 \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix} + A_2 \begin{bmatrix} 0 \\ 1 \\ \vdots \\ 0 \end{bmatrix} + \dots \; A_N \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 1 \end{bmatrix} = \begin{bmatrix} A_1 \\ A_2 \\ \vdots \\ A_N \end{bmatrix}$$

Sehingga matriksnya adalah sebagai berikut

$$\begin{bmatrix} 1 & 0 & \dots & 0 \\ 0 & 1 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 1 \end{bmatrix}$$

Untuk N=4 maka berarti ada 4 fungsi periodik yaitu dengan periode 1 hingga 4.
fungsi 1-periodik menjadi :

$$\begin{bmatrix} 1 \end{bmatrix}$$

fungsi 2-periodik menjadi :

$$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

fungsi 3-periodik menjadi :

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

fungsi 4-periodik menjadi :

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

2.3.2. Penggabungan matriks untuk membentuk ruang vektor

Seperti dijelaskan sebelumnya di bagian 2.2.2.2, sebuah fungsi periodik yang sudah direpresentasikan dalam bentuk matriks dapat digabungkan dengan matriks yang lain dengan periode yang berbeda untuk mendapatkan matriks yang dapat digunakan untuk merepresentasikan ruang vektor[10]. Dalam permasalahan soal SPOJ PERIOD1 ini, jumlah fungsi periodik yang perlu digabungkan dan direpresentasikan dengan matriks berjumlah N. N pada permasalahan ini mencakup 1 hingga kurang dari 10^8, sehingga untuk jumlah N kecil, sangat memungkinkan untuk membangun dan mengabungkan beberapa fungsi periodik dalam bentuk matriks, sedangkan untuk N yang besar, sangat tidak mungkin untuk dilakukan. Untuk contoh kasus N=4 maka pengabungan matriksnya menjadi:

$$\begin{bmatrix} 1 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \end{bmatrix}$$

Dapat dilihat bahwa baris dari matriks gabungan tersebut berjumlah bilangan terkecil yang merupakan kelipatan dari 1,2,3 dan 4 yaitu 12 sehingga masing-masing representasi fungsi periodik diperpanjang dengan pola yang sama sehingga memenuhi jumlah baris yang diharapkan yaitu 12.

2.3.3. Pencarian Rank Matriks pada Matriks

Seperti dijelaskan sebelumnya di bagian 2.2.3.2, rank matriks[9] adalah jumlah maksimal vektor kolom pada sebuah matriks yang saling bebas linear[8]. Pada permasalahan matriks biasa, pencarian rank dapat dilakukan dengan melakukan operasi matriks yaitu Gauss-Jordan[20] sehingga mendapatkan jumlah vektor kolom yang saling bebas linear. Tetapi dalam pengerjaan permasalahan PERIOD1 ini tidak dapat dilakukan karena perhitungan Gauss-Jordan melibatkan pembentukan matriks yang mana sudah dibahas sebelumnya tidak mungkin dilakukan. Hal yang penting dan akan dibahas pada bagian selanjutnya adalah mengenai keterkaitan antara sifat bebas linear pada matriks ini dengan faktor bilangan.

2.3.4. Hubungan antar bebas linear dan faktor bilangan

Pada bagian 2.2.2.3, diketahui bersama bahwa vektor kolom bersifat bebas linear jika vektor tersebut tidak dapat dibentuk melalui penjumlahan maupun perkalian skalar dari vektor kolom yang lain pada matriks tersebut[8]. Dari observasi yang dilakukan maka dapat diketahui bahwa fungsi periodik dengan periode P_1 dan P_2 , dimana $P_1 \neq P_2$, vektor-vektornya selalu tidak saling bebas linear jika ada sebuah bilangan q yang membagi baik P_1 maupun P_2, atau dengan notasi matematika q|P_1 dan q|P_2. Contohnya adalah sebagai berikut:

Untuk N=4 maka untuk fungsi dengan periodenya bernilai 1 dan periodenya bernilai 2 maka ada bilangan q yang mana q|1 dan q|2 yaitu 1. Untuk fungsi dengan periodenya bernilai 1 dan periode-

nya bernilai 3 maka ada bilangan q yang mana q|1 dan q|3 yaitu 1. Untuk fungsi dengan periodenya bernilai 1 dan periodenya bernilai 4 maka ada bilangan q yang mana q|1 dan q|4 yaitu 1. Untuk fungsi dengan periodenya bernilai 2 dan periodenya bernilai 2 maka ada bilangan q yang mana q|2 dan q|2 yaitu 1 dan 2.Untuk fungsi dengan periodenya bernilai 2 dan periodenya bernilai 3 maka ada bilangan q yang mana q|2 dan q|3 yaitu 1. Untuk fungsi dengan periodenya bernilai 2 dan periodenya bernilai 4 maka ada bilangan q yang mana q|2 dan q|4 yaitu 1 dan 2. Untuk fungsi dengan periodenya bernilai 3 dan periodenya bernilai 3 maka ada bilangan q yang mana q|3 dan q|3 yaitu 1 dan 3. Untuk fungsi dengan periodenya bernilai 3 dan periodenya bernilai 4 maka ada bilangan q yang mana q|3 dan q|4 yaitu 1. Untuk fungsi dengan periodenya bernilai 4 dan periodenya bernilai 4 maka ada bilangan q yang mana q|4 dan q|4 yaitu 1, 2 dan 4.

Dengan kata lain setiap fungsi periodik dengan periode K, vektor penyusun matriksnya akan saling bebas linear dengan fungsi-fungsi dengan periode yang tidak memiliki faktor yang sama dengan K. Dan ini dikenal dalam teori bilangan sebagai sebuah fungsi yaitu fungsi Euler Totient[5].

2.3.5. Pencarian Hasil dari Fungsi Euler Totient untuk nilai tertentu

Mengacu pada bagian 2.2.7 maka dapat dihitung nilai dari fungsi Euler Totient dari n yang merupakan jumlah bilangan yang kurang dari n dan relatif prima terhadap n[5]. Yang mana berarti untuk fungsi periode dengan periode terbesar 4 atau N=4, maka perlu menghitung nilai dari fungsi euler totient dari i = 1 hingga i=4. Untuk i=1 maka berdasarkan rumus pada bagain 2.2.7.3 maka $\phi(1)$ adalah 1, $\phi(2)$ adalah 1, $\phi(3)$ adalah 2 dan $\phi(4)$ adalah 2. Berikut dicontohkan cara perhitungan $\phi(4)$ yaitu:

$$\phi(2^2) = 2^{2-1}(2-1)$$

$$\phi(4) = 2$$

2.3.6. Penjumlahan Fungsi Euler Totient hingga N

Dari observasi lebih lanjut maka diketahui bahwa N buah fungsi periodik yang digabungkan dalam bentuk matriks akan memiliki nilai dimensi yang adalah hasil penjumlahan N nilai yang didapat dari fungsi euler totient. Contohnya: Untuk fungsi dengan N=4, maka nilai rank matriks gabungannya adalah Euler(1) + Euler(2) + Euler(3) + Euler(4)= 4. Fungsi Euler(i) adalah fungsi mencari bilangan nilai euler totient untuk i. Sehingga untuk N=4 maka hasilnya adalah

$$\sum_{i=1}^{N=4} \phi(i)$$

Sebagai contoh untuk kasus uji N=4 maka:

$$rankmatriks = \phi(1) + \phi(2) + \phi(3) + \phi(4)$$

$$rankmatriks = 1 + 1 + 2 + 2$$

$$rankmatriks = 6$$

Sedangkan untuk perhitungan penjumlahan N yang lebih besar maka menggunakan rumus seperti pada bagian 2.2.8.

2.3.7. Optimasi Penjumlahan Fungsi Euler Totient

Bagian optimasi akan dijelaskan lebih lanjut pada bagian analisis dan desain pada bab 3 bagian 3.4.1.

[*Halaman ini sengaja dikosongkan*]

BAB III
ANALISIS DAN PERANCANGAN

Pada bab ini akan dijelaskan mengenai desain algoritma untuk menyelesaikan permasalahan SPOJ PERIOD1.

3.1. Desain Umum Sistem

Pada subbab ini akan dijelaskan mengenai gambaran secara umum dari algoritma yang dirancang.

Program akan diawali dengan menerima masukan berupa sebuah integer T sebagai jumlah uji coba. Pada setiap kasus uji coba, program akan menerima masukan berupa sebuah bilangan N yang menyatakan periode yang terbesar dalam ruang vektor tersebut. Setelah menerima masukan, masukan tersebut akan diolah untuk menghasilkan keluaran yang adalah menyatakan rank dari ruang vektor tesebut[1].

3.2. Desain Penghitungan Fungsi Nilai Euler Totient

Pada subbab ini akan dijelaskan mengenai pembuatan fungsi untuk menghitung nilai euler totient dari sebuah bilangan. Metode yang digunakan adalah berasal dari pengembangan metode SoE[17].

3.2.1. Metode Sieve atau Pengayakan

Mengacu pada penjelasan pada bagian 2.2.6.2, Pengayakan atau sieving adalah cara untuk menyaring nilai-nilai yang tidak termasuk, hingga yang tersisa adalah bilangan-bilangan yang diharapkan. Pada metode Sieve of Eratosthenes maka yang disaring adalah bilangan-bilangan yang adalah kelipatan dari bilangan prima, sedangkan bilangan yang tidak memiliki kelipatan atau faktor yang lebih kecil maka termasuk ke dalam himpunan bilangan prima[17].

3.2.2. Pengembangan metode Sieve of Eratosthenes

Pada bagian ini akan dijelaskan mengenai pengembangan metode Sieve of Eratosthenes untuk mencari nilai dari fungsi euler totient. Jika pada Sieve of Eratosthenes hasil bilangan prima kurang dari N didapatkan setelah semua proses pencarian berakhir[17], maka pada desain fungsi untuk perhitungan niilai fungsi Euler Totient ini, tiap bilangan prima yang ada langsung diproses untuk pencarian nilai Euler Totient nya. Dan untuk bilangan bukan prima yang adalah kelipatan dari bilangan prima yang sedang diproses maka dapat dicari nilai euler totientnya dengan menggunakan sifat multipikatif dari fungsi euler totient seperti pada bagian 2.2.7.2.

3.2.3. Desain fungsi computeTotient

Seperti dijelaskan sebelumnya dibagian 2.2.6.2 maka *pseudocode* dari fungsi pencarian nilai euler totient dari sebuah bilangan p adalah seperti pada gambar 3.1.

3.3. Desain fungsi sumEuler

Seperti dijelaskan sebelumnya di bagian 2.3.6, rank dari ruang vektor fungsi-fungsi periodik ini adalah penjumlahan N nilai fungsi euler totient. Gambar 3.2 ini menjelaskan mengenai desain fungsinya yang dilakukan sebagai pre komputasi terlebih dahulu untuk N yang memungkinkan yaitu 10^6.

3.4. Desain fungsi findSumEuler

Dalam bagian ini akan dijelaskan mengenai inti solusi dari permasalahan SPOJ PERIOD1[1]. Akan dijelaskan lagi mengenai batasan masalah. Seperti sudah dijelaskan sebelumnya, dibagian 2.3 disebutkan mengenai batasan masalah yaitu N dipastikan tidak akan lebih dari 10^8. Dengan nilai N yang sangat besar, selain tidak bisa diselesaikan dengan membangun sebuah matriks, maka pen-

```
computeTotient (N, A)
 1: for i < N do
 2:     A[i] ←i
 3: end for
 4: for p < N do
 5:     if A[p] == p then
 6:         A[p] ← p − 1
 7:         for i < N do
 8:             A[i] ← (A[i]/p) ∗ (p − 1)
 9:         end for
10:     end if
11: end for
```

Gambar 3.1: Pseudocode fungsi computeTotient

```
sumEuler (N,A)
1: computeTotient (N, A)
2: for i < N do
3:     A[i] ← A[i − 1] + A[i]
4: end for
```

Gambar 3.2: Pseudocode fungsi sumEuler

jumlahan sebanyak 10^8 kali dalam pengulangan akan melebihi batasan waktu pemrosesan pada sistem SPOJ. Sistem SPOJ diperkirakan memiliki batasan pengolahan 1 detik yaitu sebesar 10^6. Melihat kondisi ini maka dengan N lebih kecil dari 10^8 maka dimungkinan N lebih besar dari 10^7 dan ini melebihi batas waktu yang diizinkan pada permasalahan ini. Dibagian selanjutnya akan dijelaskan mengenai hasil observasi dari pendekatan faktor untuk penjumlahan seperti pada bagian 2.2.8 yaitu untuk menghitung penjumlahan euler totient untuk N lebih besar dari 10^6.

3.4.1. Pendekatan Faktor untuk Penjumlahan Euler Totient

Dalam observasi mengenai rumus

$$S_1(n) = (\frac{n*(n+1)}{2}) - (S_1(\frac{n}{2}) + S_1(\frac{n}{3}) + \cdots + S_1(\frac{n}{n})) \quad (3.1)$$

terutama bagian ke duanya yaitu

$$(S_1(\frac{n}{2}) + S_1(\frac{n}{3}) + \cdots + S_1(\frac{n}{n})) \quad (3.2)$$

ditemukan sebuah pemanggilan nilai dari $S_1(\frac{n}{k_1})$ dan $S_1(\frac{n}{k_2})$ dari indeks yang sama, atau dengan kata lain $\frac{n}{k_1}$ dan $\frac{n}{k_2}$ bernilai sama. Untuk membuktikan contohnya adalah sebagai berikut. Untuk n=12 dengan menggunakan rumus

$$S_1(n) = (\frac{n*(n+1)}{2}) - (S_1(\frac{n}{2}) + S_1(\frac{n}{3}) + \cdots + S_1(\frac{n}{n}))$$

menghasilkan

$$S_1(12) = (\frac{12*(12+1)}{2}) - (S_1(\frac{12}{2}) + S_1(\frac{12}{3}) + \cdots + S_1(\frac{12}{12}))$$

Selanjutnya hasil pembagian akan disajikan dalam bentuk tabel 3.1

Tabel 3.1: Kiri: daftar $\frac{n}{k}$, Kanan : daftar n, $\phi(n)$ dan penjumlahan N Euler Totient

k_i	$\frac{n}{k_i}$
2	6
3	4
4	3
5	2
6	2
7	1
8	1
9	1
10	1
11	1
12	1

n	$\phi(n)$	$S_1(n)$
1	1	1
2	1	2
3	2	4
4	2	6
5	4	10
6	2	12
7	6	18
8	4	22
9	6	28
10	4	32
11	10	42

Dari observasi dari tabel kiri dari tabel 3.1, yang pertama dapat dilihat bahwa semakin k meningkat, maka jumlah $\frac{n}{k_i}$ yang memiliki hasil yang sama pun meningkat. Pada saat k_i=2 maka hanya ada 1 $\frac{n}{k_i}$ yang bernilai 6. Begitu pula saat k_i=3 maupun 4, $\frac{n}{k_i}$ yang bernilai sama hanya ada 1. Tetapi untuk k_i=5 atau 6, ada 2 nilai yang sama untuk $\frac{n}{k_i}$. Dan untuk $k_i > 6$ maka $\frac{n}{k_i}$ bernilai 1.

Observasi ini menujukan bahwa untuk semua $k_i > \frac{n}{2}$ pasti memiliki $\frac{n}{k_i}$ =1 untuk semua k_i-nya. Observasi lebih lanjut menunjukan bahwa k_1 dan k_2 dapat saling ditukar jika $\frac{n}{k_1} = k_2$ untuk $k_1 \neq k_2$ dan $k_1 \mid n$. Sehingga jumlah k_i yang dibutuhkan hanya $\sqrt{(n)}$ karena untuk tiap k_i akan didapatkan juga k_j yang adalah $\frac{n}{k_i}$. Selanjutnya jumlah dari tiap $\frac{n}{k_i}$ yang sama memiliki jumlah tertentu yang membentuk sebuah pola. Yaitu jumlahnya adalah selisih antar pembagian 2 nilai k_i yang berdekatan. Contohnya pada n=12 ini, saat k_i=2 maka hasil $\frac{n}{k_i}$ adalah 6, sedangkan saat k_i=3 maka hasil

$\frac{n}{k_i}$ adalah 4, maka jumlah $\frac{n}{2}$ yang bernilai 2 adalah 6-4 yaitu 2.
Dengan rumus dapat dituiskan sebagai berikut:
untuk setiap pasang $k_i \mid n$, $k_j \mid n$ dan $\frac{n}{k_i}$=k_j tetapi k_j-$k_i \neq 1$ dan $a = \frac{n}{k_i}$ berlaku

$$\sum_{i=2}^{N=n} S_1(\frac{n}{i}) = \sum_{k_i=2}^{N=\sqrt{(n)}} S_1(k_i) * (\frac{n}{k_i} - \frac{n}{k_{i+1}}) + S_1(\frac{n}{k_i}) \quad (3.3)$$

untuk $k_i \neq \sqrt{(n)}$ dan

$$\sum_{i=2}^{N=n} S_1(\frac{n}{i}) = \sum_{k_i=2}^{N=\sqrt{(n)}} S_1(k_i) * (\frac{n}{k_i} - \frac{n}{k_{i+1}}) \quad (3.4)$$

untuk $k_i = \sqrt{(n)}$
Dan untuk $\frac{n}{k_i}$ =1 ada sejumlah $\frac{n}{2}$ untuk n bilangan genap dan $\frac{n}{2} + 1$ untuk n bilangan ganjil. Pada gambar 3.3 disajikan pseudocode untuk fungsi findSumEuler ini.

3.5. Desain fungsi Utama

Pada gambar 3.4 akan dijelaskan mengenai desain dari fungsi utama untuk penyelesaian masalah SPOJ PERIOD1. Secara umum terbagi menjadi 2 macam pengerjaan. Yang pertama yaitu untuk N yang dimasukan $< 10^6$ dan yang kedua yaitu untuk N $\geq 10^6$.

findSumEuler (N, A)

1: $sum \leftarrow 0$
2: $i \leftarrow 2$
3: **for** $i < \sqrt{(}N)$ **do**
4: $\quad sum = sum + A[i] * (\frac{N}{i} - \frac{N}{i+1})$
5: $\quad$ **if** $\frac{n}{i} \neq i$ **then**
6: $\quad\quad sum = sum + A[\frac{N}{i}]$
7: $\quad$ **end if**
8: **end for**
9: **if** $2 \mid N$ **then**
10: $\quad sum = sum + \frac{N}{2}$
11: **else**
12: $\quad sum = sum + \frac{N}{2} + 1$
13: **end if**
14: **return** sum

Gambar 3.3: Pseudocode fungsi findSumEuler

Main

1: $A[] \leftarrow 0$
2: $batas \leftarrow batas$
3: sumEuler(batas, A)
4: $TC \leftarrow testcase$
5: **for** $i < TC$ **do**
6: $n \leftarrow inputan$
7: **if** $n < batas$ **then**
8: Print $A[n]$
9: **else**
10: Print $(\frac{n*(n+1)}{2}) - findsumEuler(n, A)$
11: **end if**
12: **end for**

Gambar 3.4: Pseudocode fungsi main

BAB IV
IMPLEMENTASI

4.1. Lingkungan implementasi

Lingkungan implementasi dan pengembangan yang dilakukan adalah sebagai berikut:

1. Perangkat Keras
   (a) Processor Intel® Core™ i5-7400 CPU @ 3.00GHz (4 CPUs), 3.0GHz
   (b) Random Access Memory 8192MB
2. Perangkat Lunak
   (a) Sistem Operasi Windows 10 Pro 64-bit
   (b) IDE Dev C++
   (c) Bahasa Pemrograman C++
   (d) MinGW 64-bit

4.2. Implementasi Program Utama

Subbab ini menjelaskan implementasi program utama. Program ini merupakan program yang digunakan untuk menyelesaikan permasalahan PERIOD1.

4.2.1. Penggunaan Library dan Define

Program ini menggunakan library standar C++ seperti yang ditunjukkan pada kode sumber. Untuk variabel MAKS menyimpan N terbesar yang perlu dicari terlebih dahulu (pre-komputasi) dengan menggunakan fungsi Euler Totient. Sedangkan MAXX adalah variabel yang menyimpan N terbesar yang diizinkan pada permasalahan ini. Untuk mempersingkat maka tipe data long long didefinisikan dengan ll. Lalu dibutuhkan array dengan nama arr dengan ukuran

Kode Sumber 4.1: Implementasi libary dan definie PERIOD1

```
#include <bits/stdc++.h>
#define MAKS 1000000
#define MAXX 100000000
#define ll long long

using namespace std;

ll arr[MAXX/2];
```

[MAXX/2]; untuk menyimpan sebagian dari nilai sumasi MAXX/2 Euler Totient. Kode Sumber 4.1 menyatakan contoh dari implementasi libary dan define untuk penyelesaian permasalahan PERIOD1.

4.2.2. Implementasi Fungsi Main

Subbab ini menjabarkan implementasi fungsi *main* yang dijelaskan pada subbab 3.5. Implementasi fungsi ini dapat dilihat pada kode sumber 4.2.

Pada bagian awal program memanggil fungsi sumEuler sebagai fungsi pre-komputasi sumasi N fungsi Euler Totient yang mana N lebih kecil dari MAKS. Variabel tcase adalah menerima inputan jumlah testcase. Untuk tiap testcase, scan n yang adalah inputan user sebagai periode terbesar dalam ruang vektor tersebut. Tiap n yang diterima diperiksa apakah lebih kecil dari MAKS atau tidak. Jika lebih kecil maka dapat langsung menampilkan hasil prekomputasi sumasi fungsi Euler Totient yang sudah dilakukan. Jika lebih besar maka dilakukan pencarian dengan menjalankan rumus pada bagian 3.4.1 yang memanggil fungsi findSumEuler.

Kode Sumber 4.2: Implementasi fungsi Main PERIOD1

```
int main()
{
	gen();
	ll n;
	int tcase;
	scanf("%d",&tcase);
	while(tcase--)
	{
		scanf("%lld",&n);
		if(n<MAKS)
		printf("%lld\n",arr[n]);
		else
		{
			printf("%lld\n",n*(n+1)/2-
			    ↪ find(n));
		}
	}
}
```

Kode Sumber 4.3: Implementasi fungsi computeTotient PERIOD1

```
void computeTotient(int n)
{

   for (int i=1; i<n; i++)
      arr[i] = i;

   for (int p=2; p<n; p++)
   {
      if (arr[p] == p)
      {
         arr[p] = p-1;
         for (int i = 2*p; i<n; i += p)
         {
           arr[i] = (arr[i]/p) * (p-1);
         }
      }
   }
}
```

4.2.3. Implementasi Fungsi computeTotient

Subbab ini menjabarkan implementasi fungsi *computeTotient* yang dijelaskan pada subbab 3.2.3. Implementasi fungsi ini dapat dilihat pada kode sumber 4.3.

Pada bagian awal fungsi yaitu mengisi array dengan 1 hingga n. Kemudian melakukan iterasi sebagai implementasi dari pengembangan metode Sieve of Eratosthenes serta memanfaatkan sifat multiplikatif dari fungsi Euler Totient seperti pada bagian 3.2.3 dan lebih jelas dapat dilihat pada kode sumber 4.3.

Kode Sumber 4.4: Implementasi fungsi sumEuler PERIOD1

```
void gen()
{
	arr[1]=1;
	computeTotient(MAKS);
	for(int x=2;x<MAKS;x++)
	{
		arr[x]=arr[x-1]+arr[x];
	}
}
```

4.2.4. Implementasi Fungsi sumEuler

Subbab ini menjabarkan implementasi fungsi *sumEuler* yang dijelaskan pada subbab 3.3. Implementasi fungsi ini dapat dilihat pada kode sumber 4.4.

Pada bagian awal fungsi yaitu didefinisikan bahwa jumlah fungsi Euler Totient adalah 1. Kemudian melakukan pemanggilan fungsi computeTotient untuk n<MAKS. Lalu menjumlahkan dan menyimpan tiap penjumlahan i nilai Euler Totient dalam array yang sama yaitu array arr.

4.2.5. Implementasi Fungsi findSumEuler

Subbab ini menjabarkan implementasi fungsi *findSumEuler* yang dijelaskan pada subbab 3.4. Implementasi fungsi ini dapat dilihat pada kode sumber 4.5.

Pada awal fungsi menginisalisasi nilai sum yang akan dikembalikan atau menjadi return value dari fungsi findSumEuler. Fungsi ini melakukan iterasi sebanyak $\sqrt{N}$ lalu mengimplementasikan rumus yang sudah dibuat pada bagian 3.4.1.

Kode Sumber 4.5: Implementasi findSumEuler PERIOD1

```
long long find(long long n)
{
	long long sum=0;
	for(long long x=2;x*x<=n;x++)
	{
		sum+=arr[x]*(n/x-n/(x+1));
		if(n/x!=x){
			if(arr[n/x]!=0){
				sum+=arr[n/x];
			}
			else
			{
				arr[n/x]=(n/x*(n/x
				↪ +1)/2)-find(n/
				↪ x);
				sum+=arr[n/x];
			}
		}
	}
	if(n%2==0)
	sum+=(n/2);
	else
	sum+=((n/2)+1);
	return sum;
}
```

BAB V
UJI COBA DAN PEMBAHASAN

Pada babi ini akan dijelaskan mengenai hasil uji coba serta pembahasan dari solusi permasalahan SPOJ PERIOD1 dan soal pendukung mengenai fungsi Euler Totient yaitu CANPR seperti dijelaskan pada bagian 2.2.9.

5.1. Lingkungan Uji Coba

Uji coba dilakukan pada perangkat dengan spesifikasi berikut.

1. Perangkat Keras
   (a) Processor Intel® Core™ i5-7400 CPU @ 3.00GHz (4 CPUs), 3.0GHz
   (b) Random Access Memory 8192MB
2. Perangkat Lunak
   (a) Sistem Operasi Windows 10 Pro 64-bit

5.2. Skenario Uji Coba

Subbab ini akan menjelaskan hasil pengujian program penyelesaian permasalahan. Ada 3 bagian dari pengujian yang akan digunakan:

1. Uji Coba Contoh Kasus CANPR
2. Uji Coba Contoh Kecil
3. Uji Coba Kebenaran

5.3. Uji Coba Contoh Kasus CANPR

Pada bagian ini akan dijelaskan secara singkat mengenai desain, implementasi dan uji coba dari contoh kasus CANPR[19] pada bagian 2.2.9 yang menjadi dasar penggunaan penjumlahan dari fungsi Euler Totient.

5.3.1. Desain solusi permasalahan CANPR

Pada bagian ini akan dijelaskan mengenai desain dan implementasi sederhana dari solusi permasalahan SPOJ CANPR.

5.3.1.1. Desain fungsi Utama CANPR

Desain fungsi utama yaitu dimulai dengan menjalankan fungsi pre-komputasi sumEuler lalu dilanjutkan dengan menerima inputan berupa T yaitu jumlah kasus uji dilanjutkan dengan T baris berikutnya berisi masing-masing N dan L sebagai jumlah permen dan angka GCD yang diharapkan. Ada beberapa kondisi N dan L yang bisa mempersingkat yaitu jika:

1. L > N maka cetak 0 karena GCD tidak mungkin lebih besar dari pada N
2. L = N maka cetak 1 karena hanya ada 1 pasang GCD yang adalah sebesar N yaitu pasangan (N,N)
3. L < N maka hitung dengan menggunakan rumus yang akan dijelaskan selanjutnya

Untuk L<N maka berdasarkan bahasan pada bagian 2.2.9.2.3 menggunakan hasil observasi pada bagian 2.2.8, pasangan bilangan yang mungkin dapat ditemukan dengan rumus

$$2 * S_1(\frac{N}{L}) - 1 \tag{5.1}$$

Dikali dengan 2 karena pasangan (x,y) berbeda dengan (y,x) sehingga untuk mendapatkan pasangan yang merupakan kebalikan, diperlukan pengalian dengan 2. Dikurangi dengan 1 adalah karena hasil

Main

1: $A[] \leftarrow 0$
2: $batas \leftarrow batas$
3: sumEuler(batas,A)
4: $TC \leftarrow testcase$
5: **for** $i < TC$ **do**
6: $n \leftarrow N$
7: $l \leftarrow L$
8: **if** $n < l$ **then**
9: Print 0
10: **else if** $n = l$ **then**
11: Print 1
12: **else**
13: Print $2 * A[\frac{n}{l}] - 1$
14: **end if**
15: **end for**

Gambar 5.1: Pseudocode fungsi Main CANPR

dari $S_1(\frac{N}{L})$ termasuk juga pasangan bilangan (x,x) yang mana pasangan kebalikannya adalah tetap sama, sehingga perlu dilakukan pengurangan dengan 1. Gambar 5.1 adalah pseudocode dari fungsi Utama solusi permasalahan CANPR ini.

5.3.1.2. Desain fungsi sumEuler CANPR

Desain fungsi sumEuler pada gambar 5.2 adalah menjumlahkan lalu menyimpan hasil penjumlahan tiap nilai fungsi Euler Totient dari 1 hingga N terbesar.

```
sumEuler (N,A)
1: computeTotient (N,A)
2: for i < N do
3:     A[i] ← A[i − 1] + A[i]
4: end for
```

Gambar 5.2: Pseudocode fungsi sumEuler CANPR

5.3.1.3. Desain fungsi computeTotient CANPR

Seperti dijelaskan sebelumnya dibagian 2.2.9.2.1 maka pseudocode dari fungsi pencarian nilai euler totient dari sebuah bilangan p adalah seperti pada gambar 5.3.

5.3.2. Implementasi solusi permasalahan CANPR

Pada bagian ini akan dijelaskan mengenai implementasi dari solusi sesuai desain pada bagian sebelumnya dari permasalahan CANPR.

5.3.2.1. Penggunaan Library dan Define CANPR

Program ini menggunakan library standar C++ seperti yang ditunjukkan pada kode sumber. Untuk variabel MAKS menyimpan N terbesar yang perlu dicari terlebih dahulu (pre-komputasi) dengan menggunakan fungsi Euler Totient. Untuk mempersingkat maka tipe data long long didefine dengan ll. Lalu dibutuhkan array dengan nama arr dengan ukuran $[\mathrm{MAXX}]$, untuk menyimpan sebagian dari nilai dari euler totient maupun penjumlahannya. Kode sumber 5.1 menunjukan contoh implementasinya pada program solusi.

computeTotient (N,A)

1: **for** $i < N$ **do**
2: $A[i] \leftarrow$i
3: **end for**
4: **for** $p < N$ **do**
5: **if** $A[p] == p$ **then**
6: $A[p] \leftarrow p - 1$
7: **for** $i < N$ **do**
8: $A[i] \leftarrow (A[i]/p) * (p - 1)$
9: **end for**
10: **end if**
11: **end for**

Gambar 5.3: Pseudocode fungsi computeTotient CANPR

Kode Sumber 5.1: Implementasi library dan define CANPR

```
#include <bits/stdc++.h>
#define MAKS 1000000
#define ll long long

using namespace std;

ll arr[1000005];
```

Kode Sumber 5.2: Implementasi fungsi Main CANPR

```
int main()
{
      sumEuler();
      ll n,l;
      int tcase;
      scanf("%d",&tcase);
      while(tcase--)
      {
            scanf("%lld␣%lld",&n,&l);
            if(l>n)
            printf("0\n");
            else
            if(l==n || n/l<=1)
            printf("1\n");
            else
            printf("%lld\n",2*arr[n/l]-1);
            }
      }
}
```

5.3.2.2. Implementasi Fungsi Main CANPR

Subbab ini menjabarkan implementasi fungsi *main* yang dijelaskan pada subbab 5.3.1.1. Implementasi fungsi ini dapat dilihat pada potongan kode seperti pada Kode sumber 5.2.

Pada bagian awal program memanggil fungsi sumEuler sebagai fungsi pre-komputasi penjumlahan N fungsi Euler Totient. Variabel tcase adalah menerima inputan jumlah testcase. Untuk tiap testcase scan N dan L yang adalah inputan jumlah permen dan bilangan yang disukai. Tiap n yang diterima diperiksa apakah lebih besar dari L. Jika lebih kecil maka dapat langsung menampilkan 0.

Kode Sumber 5.3: Implementasi computeTotient CANPR

```
void computeTotient(int n)
{

   for (int i=1; i<n; i++)
      arr[i] = i;

   for (int p=2; p<n; p++)
   {
      if (arr[p] == p)
      {
         arr[p] = p-1;
         for (int i = 2*p; i<n; i += p)
         {
            arr[i] = (arr[i]/p) * (p-1);
         }
      }
   }
}
```

Jika sama maka cetak 1. Jika lebih besar maka dilakukan perhitungan sesuai rumus yang sudah diberikan pada bagian 5.3.1.1.

5.3.2.3. Implementasi Fungsi computeTotient CANPR

Subbab ini menjabarkan implementasi fungsi *computeTotient* yang dijelaskan pada subbab 5.3.1.3. Implementasi fungsi ini dapat dilihat pada kode sumber 5.3.

Pada bagian awal fungsi yaitu mengisi array dengan 1 hingga n. Kemudian melakukan iterasi sebagai implementasi dari pengembangan metode Sieve of Erastosthenes serta memanfaatkan sifat multiplikatif dari fungsi Euler Totient seperti pada bagian 5.3.1.3.

Kode Sumber 5.4: Implementasi sumEuler CANPR

```
void sumEuler()
{
       arr[1]=1;
       computeTotient(MAKS);
       for(int x=2;x<=MAKS;x++)
       {
              arr[x]=arr[x-1]+arr[x];
       }
}
```

Kode sumber 5.3 memberikan gambaran mengenai contoh pengimplementasiannya.

5.3.2.4. Implementasi Fungsi sumEuler CANPR

Subbab ini menjabarkan implementasi fungsi *sumEuler* yang dijelaskan pada subbab 5.3.1.2. Implementasi fungsi ini dapat dilihat pada kode sumber 5.4.

Pada bagian awal fungsi yaitu didefinisikan bahwa jumlah 1 fungsi Euler Totient adalah 1. Kemudian melakukan pemanggilan fungsi computeTotient. Lalu menjumlahkan dan menyimpan tiap penjumlahan i nilai Euler Totient dalam array yang sama yaitu array arr. Kode sumber 5.4 memuat gambaran pengimplementasian fungsi sumEuler.

5.3.2.5. Uji Coba kebenaran CANPR

Berdasarkan gambar dibawah ini maka, desain dan implementasi solusi permasalahan CANPR yang diusulkan dinyatakan diterima oleh online judge dan mendapat verdict Accepted (Gambar

5.4).

Gambar 5.4: Umpan Balik Online Judge pada permasalahan CANPR

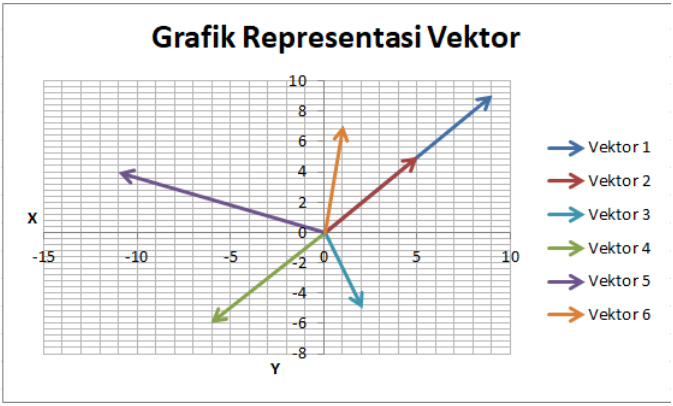
5.4. Uji Coba Contoh Kecil

Pada subbab ini akan dijelaskan dengan menggunakan contoh kecil dan akan dibahas dengan pengambaran visual lalu dibandingkan dengan hasil yang didapatkan melalui penjalanan program solusi yang sudah dibuat. Pembahasan yang pertama akan menggunakan pengambaran visual dari permasalahan ini sedangkan bagian ke-dua akan disajikan melalui perhitungan serta hasil yang didapat dari penjalanan program solusi. Contoh kecil yang akan digunakan adalah dengan N=2.

5.4.1. Pengujian secara Visual

Dengan N=2 maka menyatakan bahwa batas periode dari fungsi - fungsi periodik yang ada dalam ruang vektor yang diinginkan adalah 2. Hal ini berarti ada setidaknya 2 vektor yang bisa mewakili fungsi periodik didalam ruang vektor tersebut yaitu fungsi dengan periode sebesar 1 dan fungsi dengan periode sebesar 2. Selanjutnya fungsi periodik yang ada dibuat dalam representasi vektor seperti pada bagian 2.2.2.1 dan 2.2.1.3. Vektor tersebut kemudian dapat digambarkan seperti gambar 5.5.

Permasalahan yang diberikan adalah menghitung dimensi maksimum dari ruang vektor yang terbentuk dari fungsi dengan periode maksimum N. Maka dari itu. tiap vektor yang merepresentasikan tiap fungsi periodik perlu digambarkan agar dapat melihat rentangan[10] dari vektor yang dibentuk dan dapat mengetahui dimensinya.



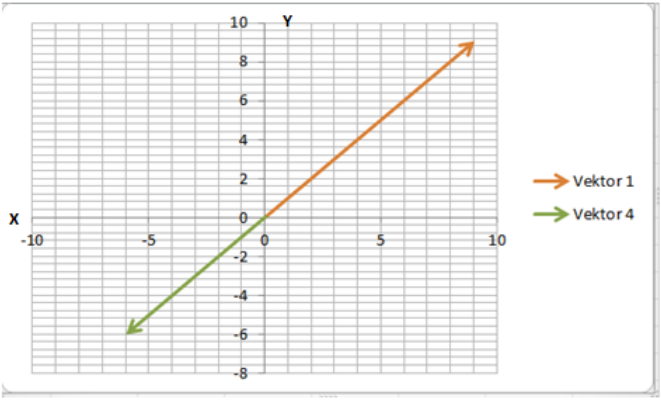
Gambar 5.5: Contoh vektor-vektor yang mungkin di ruang vektor dengan N=2

Tabel 5.1: Tabel koordinat contoh vektor

Nama Vektor	X_1	Y_1	X_2	Y_2
Vektor 1	0	0	9	9
Vektor 2	0	0	5	5
Vektor 3	0	0	2	-5
Vektor 4	0	0	-6	-6
Vektor 5	0	0	-11	4
Vektor 6	0	0	1	7

Ada 3 kombinasi yang mungkin dalam membentuk ruang vektor yaitu:

1. Gabungan antara 2 vektor yang merepresentasikan fungsi 1-periodik
2. Gabungan antara 2 vektor yang merepresentasikan fungsi 2-periodik

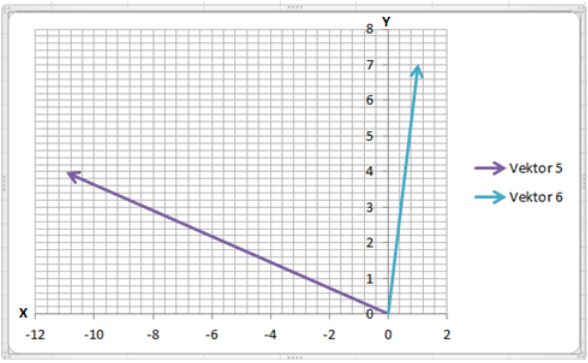


Gambar 5.6: Gabungan antara 2 vektor yang merepresentasikan fungsi 1-periodik

3. Gabungan antara 1 vektor yang merepresentasikan fungsi 1-periodik dengan 1 vektor yang merepresentasikan fungsi 2-periodik

Gabungan fungsi-fungsi tersebut seperti pada gambar (Gambar 5.6, 5.7 dan 5.8). Untuk mencari ruang vektor maka harus melakukan spanning atau perentangan yaitu, mencari daerah dari semua vektor yang dapat dibentuk oleh vektor-vektor yang ada di ruang vektor tersebut[21].

Dari gambar (Gambar 5.9, 5.10 dan 5.11) maka ketiga kondisi yang mungkin menyatakan semua kemungkinan yang terjadi dalalm batas N=2, sehingga dimensi terbesar yang terbentuk adalah 2 yaitu baik oleh fungsi periodik 1-2 atau yang terbentuk karena fungsi periodik 2-2. Sehingga rank matriks yang menyatakan dimensi dari vektor kolom penyusun ruang vektor tersebut adalah 2.



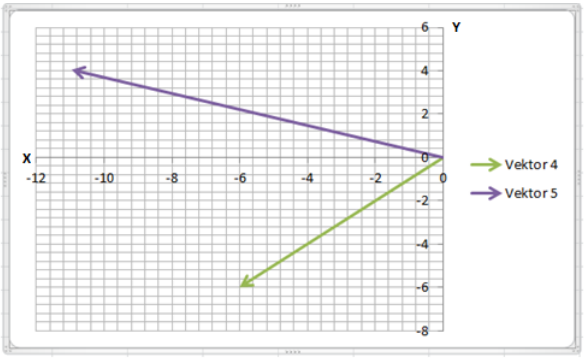
Gambar 5.7: Gabungan antara 2 vektor yang merepresentasikan fungsi 2-periodik

5.4.2. Pengujian dengan Program Solusi

Pengujian yang kedua adalah dengan menjalankan program solusi dan melihat hasilnya. Gambar 5.12 ini menyatakan bahwa hasil yang didapatkan adalah 2 yang mana didapatkan dari penjumlahan pre komputasi dari $\phi(1) + \phi(2)$.

5.5. Uji Coba Kebenaran

Kebenaran metode dan pengoptimalan diujikan dengan mengirim kode sumber terkait ke Sphere Online Judge. Berikut bukti hasil pengujian. (Gambar 5.13) Pengujian juga dilakukan dengan pengumpulan jawaban ke SPOJ sebanyak 20 kali dan gambar 5.14 adalah hasil submission serta pengujian waktunya. Dapat dilihat bahwa dalam 20 kali submission, rata-rata dari waktu yang dibutuhkan adalah 0.21 detik. (Gambar 5.14). Untuk kompleksitas dari program solusi adalah O($\sqrt{(n)}$) karena perulangan yang dilakukan di saat pecarian penjumlahan euler dengan menjalankan fungsi findSumEuler seperti pada gambar 3.3.

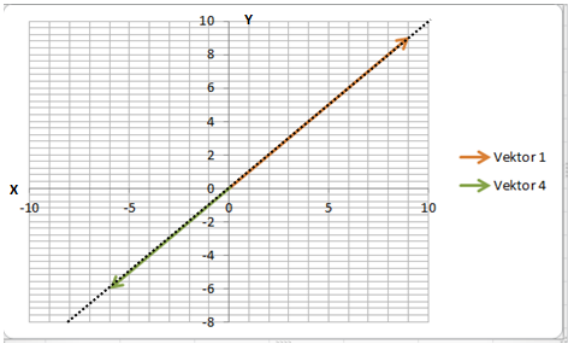


Gambar 5.8: Gabungan antara 1 vektor yang merepresentasikan fungsi 1-periodik dengan 1 vektor yang merepresentasikan fungsi 2-periodik

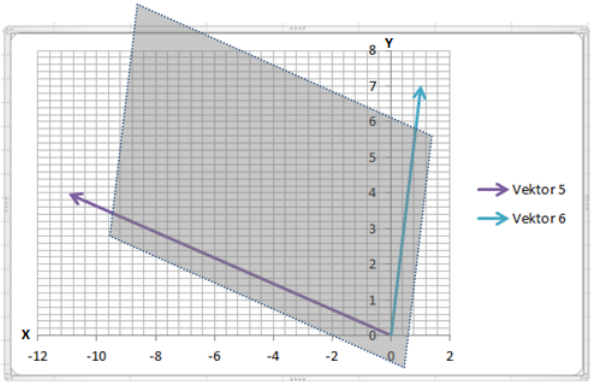
5.6. Kesimpulan

Dari hasil uji coba yang telah dilakukan maka dapat disimpulkan beberapa hal:

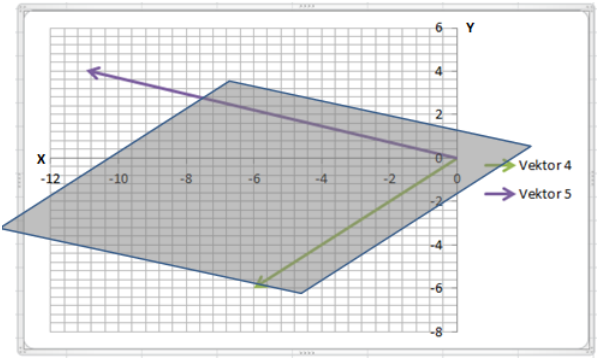
1. Pendekatan Penjumlahan Fungsi Euler Totient dapat diterapkan pada permasalahan SPOJ Candy Pair dengan kode soal CANPR.
2. Pembuktian kebenaran dengan menggunakan kasus kecil untuk permasalahan SPOJ Periodic Function, trip 1 sesuai dengan pendekatan manual dengan visualisasi.
3. Pendekatan Penjumlahan Fungsi Euler Totient dapat diterapkan pada permasalahan SPOJ Periodic Function, trip 1 dengan kode soal PERIOD1.
4. Penjumlahan Euler Totient dapat dioptimalkan sehingga memiliki kompleksitas menjadi O($\sqrt{(n)}$).



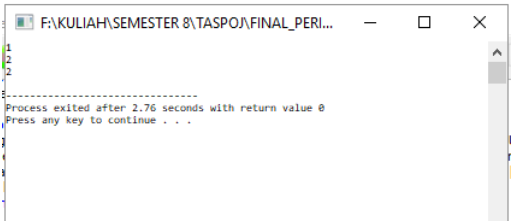
Gambar 5.9: Spanning antara 2 vektor yang merepresentasikan fungsi 1-periodik



Gambar 5.10: Spanning antara 2 vektor yang merepresentasikan fungsi 2-periodik



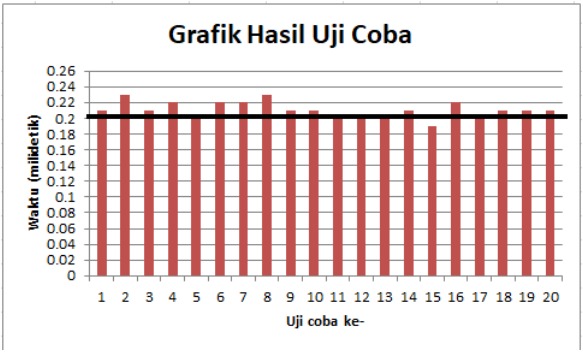
Gambar 5.11: Spanning antara 1 vektor yang merepresentasikan fungsi 1-periodik dengan 1 vektor yang merepresentasikan fungsi 2-periodik



Gambar 5.12: Penjalanan program solusi untuk N=2 dengan T=1

21548545	☐	2018-04-20 08:20:05	Periodic function, trip 1	**accepted** edit ideone it	0.21	384M	C++ 4.3.2

Gambar 5.13: Umpan Balik Online Judge dengan Metode Sumasi Euler Totient Function pada permasalahan PERIOD1



Gambar 5.14: Grafik rata-rata waktu pengerjaan untuk solusi permasalahan PERIOD1 pada SPOJ

BAB VI
KESIMPULAN

Berdasarkan penjabaran di bab-bab sebelumnya, dapat disimpulkan beberapa hal terkait penyelesaian permasalahan Periodic Function, trip 1 dengan kode soal PERIOD1

1. Metode Penjumlahan Fungsi Euler Totient dapat digunakan untuk penyelesaian permasalahan penentuan dimensi dari keluarga fungsi periodik.
2. Metode Optimasi Penjumlahan Fungsi Euler Totient dapat penyelesaikan permasalahan pada situs SPOJ Periodic Function, trip 1.
3. Penjumlahan dari fungsi-fungsi Euler Totient dapat dipercepat dengan menggunakan pola tertentu sehingga hanya memiliki kompleksitas O($\sqrt{(N)}$).

[*Halaman ini sengaja dikosongkan*]

DAFTAR PUSTAKA

[1] *Problem PERIOD1*. [Online]. Available: http://www.spoj.com/problems/PERIOD1/.

[2] E. S. o. A.A. Konyushkov, "Periodic function", *Encyclopedia of Mathematics*, [Online]. Available: http://www.encyclopediaofmath.org/index.php?title=Periodic_function&oldid=15049.

[3] H. Anton and C. Rorres, *Elementary Linear Algebra: Applications Version, 11th Edition*. Wiley Global Education, 2013, ISBN: 9781118879160. [Online]. Available: https://books.google.co.id/books?id=loRbAgAAQBAJ.

[4] A. Vazzana, M. Erickson, and D. Garth, *Introduction to Number Theory*, ser. Textbooks in Mathematics. Taylor & Francis, 2007, ISBN: 9781584889373. [Online]. Available: https://books.google.co.id/books?id=BeFIIvcBT80C.

[5] E. W. Weisstein, *Totient Function*. [Online]. Available: http://mathworld.wolfram.com/TotientFunction.html.

[6] H. Anton and C. Rorres, *Elementary Linear Algebra: Applications Version, 11th Edition*. Wiley Global Education, 2013, pp. 131–132, ISBN: 9781118879160. [Online]. Available: https://books.google.co.id/books?id=loRbAgAAQBAJ.

[7] ——, *Elementary Linear Algebra: Applications Version, 11th Edition*. Wiley Global Education, 2013, pp. 25–27, ISBN: 978 1118879160. [Online]. Available: https://books.google.co.id/books?id=loRbAgAAQBAJ.

[8] ——, *Elementary Linear Algebra: Applications Version, 11th Edition*. Wiley Global Education, 2013, pp. 202–211, ISBN: 9781118879160. [Online]. Available: https://books.google.co.id/books?id=loRbAgAAQBAJ.

[9] ——, *Elementary Linear Algebra: Applications Version, 11th Edition*. Wiley Global Education, 2013, pp. 248–258, ISBN: 9781118879160. [Online]. Available: https://books.google.co.id/books?id=loRbAgAAQBAJ.

[10] ——, *Elementary Linear Algebra: Applications Version, 11th Edition*. Wiley Global Education, 2013, pp. 183–190, ISBN: 9781118879160. [Online]. Available: https://books.google.co.id/books?id=loRbAgAAQBAJ.

[11] ——, *Elementary Linear Algebra: Applications Version, 11th Edition*. Wiley Global Education, 2013, pp. 32–33, ISBN: 978 1118879160. [Online]. Available: https://books.google.co.id/books?id=loRbAgAAQBAJ.

[12] ——, *Elementary Linear Algebra: Applications Version, 11th Edition*. Wiley Global Education, 2013, pp. 221–228, ISBN: 9781118879160. [Online]. Available: https://books.google.co.id/books?id=loRbAgAAQBAJ.

[13] M. Erickson and A. Vazzana, *Introduction to Number Theory*. Chapman & Hall/CRC, 2007, pp. 15–16, ISBN: 1584889373, 9781584889373.

[14] ——, *Introduction to Number Theory*. Chapman & Hall/CRC, 2007, pp. 21–25, ISBN: 1584889373, 9781584889373.

[15] R. Zazkis and P. Liljedahl, "Understanding Primes: The Role of Representation", *Journal for Research in Mathematics Education*, vol. 35, no. 3, pp. 164–186, 2004, ISSN: 00218251, 19452306. [Online]. Available: http://www.jstor.org/stable/30034911.

[16] E. Weisstein, "Making MathWorld", *The Mathematica Journal*, vol. 10, no. 3, 2007. DOI: 10.3888/tmj.10.3-3.

[17] S. Horsley, "ΚΟΣ ΚΙΝΟΝ ΕΡΑΤΟΣ Θ ΕΝΟΥ Σ . or, The Sieve of Eratosthenes. Being an Account of His Method of Finding All the Prime Numbers, by the Rev. Samuel Horsley, F. R. S.", *Philosophical Transactions (1683-1775)*, vol. 62, pp. 327–347, 1772, ISSN: 02607085. [Online]. Available: http://www.jstor.org/stable/106053.

[18] A. Varga, "Computation of coprime factorizations of rational matrices", *Linear Algebra and its Applications*, vol. 271, no. 1, pp. 83 –115, 1998, ISSN: 0024-3795. DOI: https://doi.org/10.1016/S0024-3795(97)00256-5. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0024379597002565.

[19] *Problem CANPR*. [Online]. Available: https://www.spoj.com/problems/CANPR/.

[20] H. Anton and C. Rorres, *Elementary Linear Algebra: Applications Version, 11th Edition*. Wiley Global Education, 2013, pp. 11–15, ISBN: 9781118879160. [Online]. Available: https://books.google.co.id/books?id=loRbAgAAQBAJ.

[21] ——, *Elementary Linear Algebra: Applications Version, 11th Edition*. Wiley Global Education, 2013, pp. 196–200, ISBN: 9781118879160. [Online]. Available: https://books.google.co.id/books?id=loRbAgAAQBAJ.

[*Halaman ini sengaja dikosongkan*]

BIODATA PENULIS

Penulis bernama Michael Dave, putra pertama dari dua bersaudara yang lahir pada tanggal 02 November 1995 di Surabaya. Penulis telah mengenyam pendidikan di Sekolah Dasar Kristen Gloria 2 tahun 2002 hingga 2008, Sekolah Menengah Pertama Kristen Gloria 1 pada tahun 2008 hingga 2011, dan Sekolah Menengah Atas Kristen Gloria 1 pada tahun 2011 hingga 2014. Pada masa penulisan, penulis sedang menempuh masa studi S1 di Institut Teknologi Sepuluh Nopember, Surabaya di Jurusan Teknik Informatika.

Selama masa studi, penulis memiliki ketertarikan yang dalam mengenai *Image Processing*, rancang bangun aplikasi *game*, dan rancang bangun aplikasi sistem informasi serta permasalahan optimasi dan keamanan jaringan. Keinginan penulis untuk mengajar dan membagikan ilmu juga mendorong penulis untuk menjadi pemberi tutor untuk beberapa matakuliah jurusan serta mengajar anak-anak smp dan sma melalui bimbingan privat diluar kampus.

Di luar kesibukan akademik, penulis juga terlibat aktif di gereja sebagai mentor kelompok kecil dan pengurus, serta di kampus melalui organisasi kerohanian Kristen di tingkat jurusan sebagai ketua. Penulis juga berkontribusi dalam berbagai kepanitiaan, terutama di tingkat jurusan melalui himpunan mahasiswa yang ada di tingkat jurusan.