JURNAL TEKNIK KOMPUTER AMIK BSI

DESAIN ALGORITHMA OPERASI PERKALIAN MATRIKS MENGGUNAKAN METODE FLOWCHART

Rini Nuraini

***Abstract* — Understanding the problems of science in particular matrix multiplication operations, for some people considered difficult to understand, such as how the sequence of steps, the order logic, decision making, and the arithmetic process. Based tersebutlah, the author wishes to lift one on the operation matrix, ie a matrix multiplication operation in this article, to be made a flowchart and pseudocodenya method. The goal is to learn and understand the example problems by describing the sequence of logic, decision-making, and the process of arithmetic, using symbols, so it is easy to understand. The symbols are symbols in the flowchart, which is a tool or a means of showing the steps that must be taken to resolve the problem of computing a way to express it in a series of special graphic symbols. Pseudocode is a description of a computer programming algorithm that uses a simple structure of some programming language, but the language is only intended to be human readable. The difference lies in the way of delivery, pseudocode using words to describe an algorithm, while the flowchart using pictures. The main purpose of the pseudocode own use is to enable people to understand the principles of an algorithm. The conclusion of this article is to answer one question sample matrix operations in two ways, namely the method flowchart and pseudocode. To analyze the validity of these answers can be seen in the next article from the same author, the title *"Desk Check Method of Flowchart Operation Multiplication Matrix"*.**

Intisari* —** Memahami soal-soal *science* khususnya operasi perkalian matriks, bagi sebagian orang dianggap sulit untuk dipahami, seperti bagaimana urutan langkah-langkahnya, urutan logikanya, pengambilan keputusannya, dan proses aritmatikanya. Berdasarkan hal tersebutlah, penulis berkeinginan mengangkat salah satu soal operasi matriks, yaitu operasi perkalian matriks dalam artikel ini, untuk dibuatkan metode *flowchart* dan *pseudocode*nya. Tujuannya adalah untuk mempelajari dan memahami contoh soal tersebut dengan menggambarkan urutan logika, pengambilan keputusan, dan proses aritmatikanya, dengan menggunakan simbol, sehingga mudah dipahami. Simbol tersebut adalah simbol-simbol dalam flowchart, yang merupakan suatu alat atau sarana yang menunjukkan langkah-langkah yang harus dilaksanakan dalam menyelesaikan suatu permasalahan untuk komputasi dengan cara mengekspresikannya ke dalam serangkaian simbol-simbol grafis khusus. Pseudocode adalah deskripsi dari algoritma pemrograman komputer yang menggunakan struktur sederhana dari beberapa bahasa pemograman, tetapi bahasa tersebut hanya ditujukan agar dapat mudah dibaca manusia. Perbedaannya terletak pada cara penyampaiannya, *pseudocode* menggunakan kata-kata untuk menjelaskan suatu algoritma, sedangkan *flowchart* menggunakan gambar. Tujuan penggunaan utama dari *pseudocode* sendiri adalah untuk memudahkan manusia dalam memahami prinsip-prinsip dari suatu algoritma. Kesimpulan dari artikel ini adalah menjawab salah satu contoh soal operasi matriks dengan dua cara, yaitu metode *flowchart* dan *pseudocode*. Untuk menganalisa kebenaran dari jawaban-jawaban tersebut dapat dilihat pada artikel berikutnya dari penulis yang sama, dengan judul, "Desk Check Table* pada *Flowchart* Operasi Perkalian Matriks".**

Kata Kunci: Desain Algorithma, Matriks, *Flowchart, Pseudecode.*

Program Studi Teknik Komputer AMIK BSI Jakarta. Jln. RS Fatmawati No. 24 Pondok Labu Jakarta Selatan Telp (021)7500282/(021) 7500680 ; Fax (021) 7513790, e-mail: rini.rna@bsi.ac.id

I. PENDAHULUAN

Ada beberapa langkah dasar yang perlu untuk diikuti dalam pembuatan suatu algoritma, antara lain adalah: pernyataan masalah; membangun model dari suatu masalah; perancangan algoritma dari model; menguji kebenaran algoritma; implementasikan dengan suatu bahasa pemrograman seperti C, Java, dan lain-lain; dokumentasi; dan analisa kompleksitas algoritma seperti analisa *output* dengan menggunakan *desk chek table*, *space complexity*, dan *time* complexity.

Pembuatan algoritma mempunyai banyak keuntungan diantaranya: pembuatan atau penulisan algoritma tidak tergantung pada bahasa pemrograman apapun, artinya penulisan algoritma independen dari bahasa pemrograman dan komputer yang melaksanakannya; notasi algoritma dapat diterjemahkan ke dalam berbagai bahasa pemrograman; apapun bahasa pemrogramannya, *output* yang akan dikeluarkan sama, karena algoritmanya sama.

Soal-soal *science* khususnya operasi perkalian matriks, bagi sebagian orang dianggap sulit untuk dipahami, seperti bagaimana urutan langkah-langkahnya, urutan logikanya, pengambilan keputusannya, dan proses aritmatikanya. Berdasarkan hal tersebut itu, penulis berkeinginan mengangkat salah satu soal Aljabar Linier, yaitu Perkalian Matrik untuk dijadikan contoh soal dalam artikel ini, untuk dibuatkan metode *flowchart* dan *pseudocode*nya.

Tujuannya adalah untuk mempelajari dan memahami contoh soal tersebut dengan menggambarkan urutan logika, pengambilan keputusan, dan proses aritmatikanya, dengan menggunakan simbol, sehingga mudah dipahami. Biasanya, sesuatu yang dapat digambarkan dengan visual akan lebih mudah untuk dipahami. Simbol tersebut adalah simbol-simbol dalam flowchart, yang merupakan tools atau alat atau suatu

sarana yang menunjukkan langkah-langkah yang harus dilaksanakan dalam menyelesaikan suatu permasalahan untuk komputasi dengan cara mengekspresikannya ke dalam serangkaian simbol-simbol grafis khusus. Flowchart juga merupakan salah satu ilmu di dunia komputasi.

Pseudocode adalah deskripsi dari algoritma pemrograman komputer yang menggunakan struktur sederhana dari beberapa bahasa pemograman tetapi bahasa tersebut hanya ditujukan agar dapat mudah dibaca manusia. Biasanya yang ditulis dari pseudocode adalah variabel dan *function*. Fungsi dari pseudocode sama dengan Flowchart. Perbedaannya terletak pada cara penyampaiannya. Pseudocode menggunakan kata-kata untuk menjelaskan suatu algoritma, sedangkan Flowchart menggunakan gambar.

Pada penulisan artikel ini, penulis sudah secara langsung mengkonversikan pseudecode ke dalam salah satu bahasa pemrograman komputer, yaitu Bahasa C. Bahasa C tersebut dapat secara langsung diujicoba di komputer, untuk melihat hasil dari kebenaran aplikasi program ini, dengan tujuan tersebutlah, penulis mengkonversikannya ke dalam Bahasa C, supaya dapat diketahui kebenaran dari aplikasi program perkalian matrik ini.

II. KAJIAN LITERATUR

1. Operation Multiplication Matrix

Matriks adalah himpunan skcalar (bilangan riil atau kompleks) yang disusun/dijajarkan secara empat persegi panjang (menurut baris-baris dan kolom-kolom). Skalar-skalar itu disebut elemen matriks [4].

Matriks diberi nama dengan huruf besar A, B, P, C, dan lain-lain. Secara lengkap ditulis matriks A = (aij), artinya suatu matriks A yang elemen-elemennya aji di mana indeks i menyatakan baris ke-i matriks j menyatakan kolom ke-j dari elemen tersebut.

Sebuah matriks A = (aij), i = 1, 2, …, m dan j = 1, 2, …, n ; yang mana berarti bahwa banyaknya baris = m serta banyaknya kolom = n.

Dua buah matriks A = (aij) dan B = (bij) dikatakan sama A = B, bila ukurannya sama (m x n) dan berlaku aij = bij untuk setiap I dan I (I = 1, 2, … , m); j = 1, 2, … , n).

2. Algorithma

Algoritma berasal dari nama seorang Ilmuwan Arab yang bernama Abu Ja‟far Muhammad Ibnu Musa Al Khuwarizmi penulis buku berjudul Al Jabar Wal Muqabala (Buku Pemugaran dan Pengurangan). Kata Al Khuwarizmi dibaca orang barat menjadi Algorism yang kemudian lambat laun menjadi Algorithm diserap dalam bahasa Indonesia menjadi Algoritma. Algoritma dapat diartikan urutan langkah-langkah (instruksi-instruksi/aksi-aksi) terbatas untuk menyelesaikan suatu masalah.

Syarat-Syarat Algoritma, yaitu [7]:

1. Finiteness (Keterbatasan)
   A.lgoritma harus berakhir setelah melakukan sejumlah langkah proses.
2. Definiteness (Kepastian)
   Setiap langkah algoritma harus didefinisikan dengan tepat dan tidak menimbulkan makna ganda
3. Input (Masukan)
   Sebuah algoritma memiliki nol atau lebih masukan (input) yang diberikan kepada algoritma sebelum dijalankan.
4. Output (Keluaran)
   Setiap algoritma memberikan satu atau beberapa hasil keluaran.
5. Effectiveness (Efektivitas)
   Langkah-langkah algoritma dikerjakan dalam waktu yang “wajar”.

Suatu Algoritma dapat terdiri dari tiga struktur dasar, yaitu runtunan, pemilihan dan pengulangan. Berikut Penjelasan ringkas dari tiga struktur tersebut:

1. Runtunan
   Runtunan yaitu satu atau lebih instruksi yang dikerjakan secara berurutan sesuai dengan urutan penulisannya. Urutan dari instruksi menentukan hasil akhir dari suatu algoritma. Bila urutan penulisan berubah maka mungkin juga hasil akhirnya berubah.
2. Pemilihan
   Pemilihan yaitu instruksi yang dikerjakan dengan kondisi tertentu. Kondisi adalah persyaratan yang dapat bernilai benar atau salah. Instruksi hanya dilaksanakan apabila kondisi bernilai benar, sebaliknya apabila salah maka instruksi tidak akan dilaksankan. Pernyataaan kondisi menggunakan statemen If (jika) dan Then (maka).
3. Pengulangan
   Pengulangan merupakan pengulangan sejumlah aksi yang sama sebanyak jumlah yang ditentukan atau sesuai dengan kondisi yang diinginkan. Beberapa statemen pengulangan yaitu:
   a) For … To ... Do / For ... Downto ... Do
   b) While … Do
   c) Repeat ... Until

Algoritma dapat ditulis dengan cara berikut:

1. Menggunakan bahasa natural
2. Menggunakan kode semu (pseudo-code)
   Teknik penulisan yang mendekati bahasa pemrograman tertentu
3. Menggunakan diagram alir (flowchart)
   Teknik penyajian dengan menggunakan simbol-simbol.

3. Array

Peubah atau variable hanya dapat menyimpan sebuah nilai saja. Peubah tidak dapat menyimpan beberapa buah nilai yang bertipe sejenis sekaligus. Sementara dalam kebutuhan pemrograma, seringkali kita diharuskan atau dibutuhkan mengolah sekumpulan data yang bertipe sama dalam yang bersamaan, misalnya dalam kasus menampung hasil ujian 100 orang mahasiswa, table harga-harga barang di swalayan, dan lain sebagainya. Dikarenakan setiap elemen data bertipe sama, maka elemen tersebut, cukup diacu dengan satu nama peubah, dan untuk membedakan elemen data yang satu dengan elemen data yang lainnya, maka elemen diacu dengan menggunakan

indeks (*subscript*). Misalnya jika data nilai ujian dilambangkan dengan peubah A, maka indeksnya A_i menyatakan nilai ujian mahasiswa yang kepi.

Dalam matematika, statistik, atau bidang eksakta lainnya, sering ditemui besaran yang menggunakan nama peubah berindeks seperti: u_1, u_2, u_3, u_4, u_5, u_6, u_7, u_8, u_9, u_{10}; a_1, a_2, a_3, … , a_n; $v_k \geq 0$, untuk k = 0, 1, 2, … , n; dan sebagainya. Besaran-besaran tersebut adalah sekumpulan nilai yang bertipe sama. Nama peubah yang menyatakan kumpulan nilai itu masing-masing adalah u, a, dam v. Nilai tertentu di dalam kumpulan peubah tersebut diacu dengan menggunakan indeksnya, misalnya u_3, a_8, a_k, atau v_j, dan lain-lain.

Dalam kegiatan pemrograman, sekumpulan data yang bertipe sama perlu disimpan sementara di dalam memori komputer untuk sewaktu-waktu dimanipulasi. Misalnya jika hendak menghitung nilia rata-rata kumpulan data nilai ujian, dengn rumus:

$$\text{Rata-rata} = (a_1 + a_2 + \ldots + a_n) / n = \sum_{i=1}^{n} ai$$

Sekumpulan data yang bertipe sama disimpan secara berurutan di dalam memori komputer, setiap elemen data diacu dengan menggunakan indeks. Indeks menyatakan posisi data relative di dalam kumpulannya. Struktur penyimpanan data seperti ini dinamankan larik (array). Nama lain untuk larik adalah table, vector, atau peubah majemuk (satu peubah mempunyai banyak elemen).

Larik adalah struktur data yang menyimpan sekumpulan elemen yang bertipe sama, setiap elemen diakses langsung melalui indeksnya. Indeks larik haruslah tipe data yang menyatakan keterurutan, misalnya integer atau karakter [1].

III. METODE PENELITIAN

1. Flowchart

Flowchart dapat diartikan sebagai suatu alat atau sarana yang menunjukkan langkah-langkah yang harus dilaksanakan dalam menyelesaikan suatu permasalahan untuk komputasi dengan cara mengekspresikannya ke dalam serangkaian simbol-simbol grafis khusus [5]. Manfaat yang akan diperoleh bila menggunakan flowchart dalam pemecahan masalah komputasi:

a. Terbiasa berfikir secara sistematis dan terstruktur
b. Mudah mengecek dan menemukan bagian-bagian prosedur yang tidak valid dan bertele-tele
c. Prosedur akan mudah dikembangkan

2. *Repetition Control Stuctures*

Outline [3]:

a. Repetition Using The DOWHILE Structure

The format is:
DOWHILE condition p is true
statement block
ENDDO

b. Repetition Using the REPEAT … UNTIL Structure

The format of the REPEAT … UNTIL structure is:
REPEAT
statement
statement
…
UNTIL condition is true

c. Counted Repetition

Counted repetition occurs when the exact number of loop iterations is know in advance. The execution of the loop is controlled by a loop index, and instead of using DOWHILE, or REPEAT … UNTIL, the simple keyword DO is used as follows:
DO loop_index = initial_value to finel_value
statement block
ENDDO

3. Struktur Kendali Pengulangan

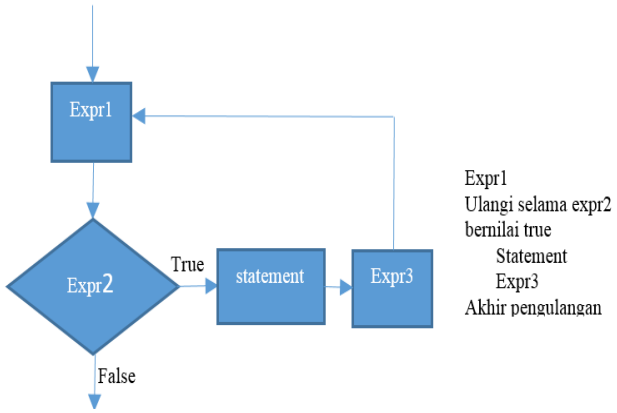
Bahasa C menyediakan tiga instruksi untuk melakukan proses pengulangan: for, while, dan do while. Ketiga intruksi ini memiliki karakteristik masing-masing [2].

a. Instruksi For

Instruksi for digunakan untuk melakukan proses pengulangan yang frekuensi pengulangannya telah diketahui sebelum proses pengulangan dimulai.

for ([expression1]; [expression2]; [expression3])
statement;

Expression1 digunakan untuk melakukan proses awal atau inisialisasi, misalnya pemberian nilai awal kepada pencacah atau counter. Expression2 berupa ekspresi Boolean yang bila dikerjakan akan memberi nilai true (bukan nol) atau false (nol). Expression3 adalah instruksi pasca pengerjaan statement. Alur logika instruksi for ditunjukkan oleh **Gambar 1**.



Sumber: Sutanta (2004)

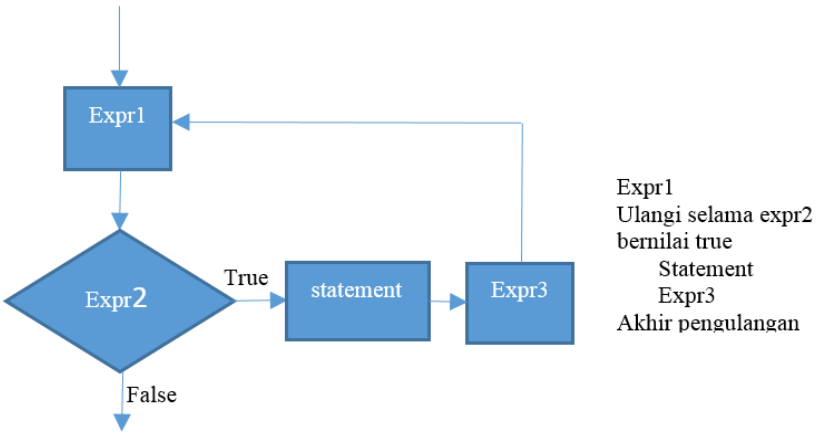
Gambar 1: Instruksi For

b. Instruksi While

Instruksi while ialah instruksi untuk melakukan proses pengulangan yang pemeriksaan syarat pengulangannya dilakukan pada awal proses. Instuksi while umumnya digunakan untuk melakukan proses pengulangan yang frekuensi pengulangannya belum diiketahui pada saat proses pengulangan dimulai.

while (expression) statement;

Expression berupa ekspresi Boolean dan berfungsi sebagai control pengulangan. Selama hasil evaluasi ekspresi ini memberikan nilai bukan nol maka statement dikerjakan berulang kali. Alur logika instruksi while ditunjukkan oleh Gambar 2.



Sumber: Sutanta (2004)

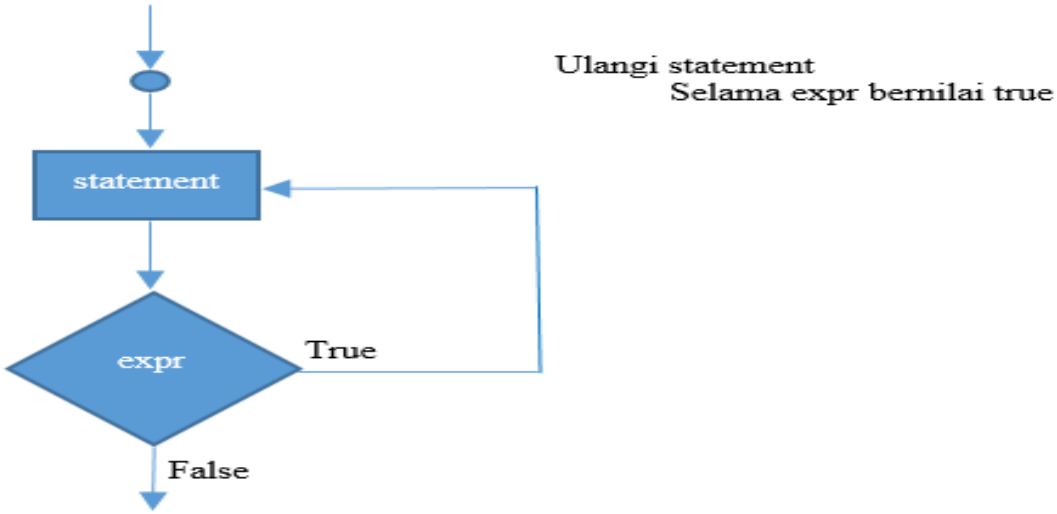
Gambar 2: Instruksi While

c. Instruksi Do While

Instruksi do while ialah instruksi untuk melakukan proses pengulangan yang pemeriksaan syarat pengulangannya dilakukan pada akhir proses. Instruksi do while umumnya digunakan untuk melakukan proses pengulangan yang belum diketahui frekuensi pengulangannya tetapi pasti dikerjakan minimal satu kali.

do statement while (expression);

Statement berupa sebuah instruksi atau beberapa instruksi yang dilingkup oleh {}. Expression berupa ekspresi Boolean dan berfungsi sebagai control pengulangan. Selama hasil evaluasi expression ini memberikan nilai bukan nol maka statement dikerjakan berulang kali. Alur logika instruksi do while ditunjukkan oleh **Gambar 3**.



Sumber: Sutanta (2004)

Gambar 3: Instruksi Do While

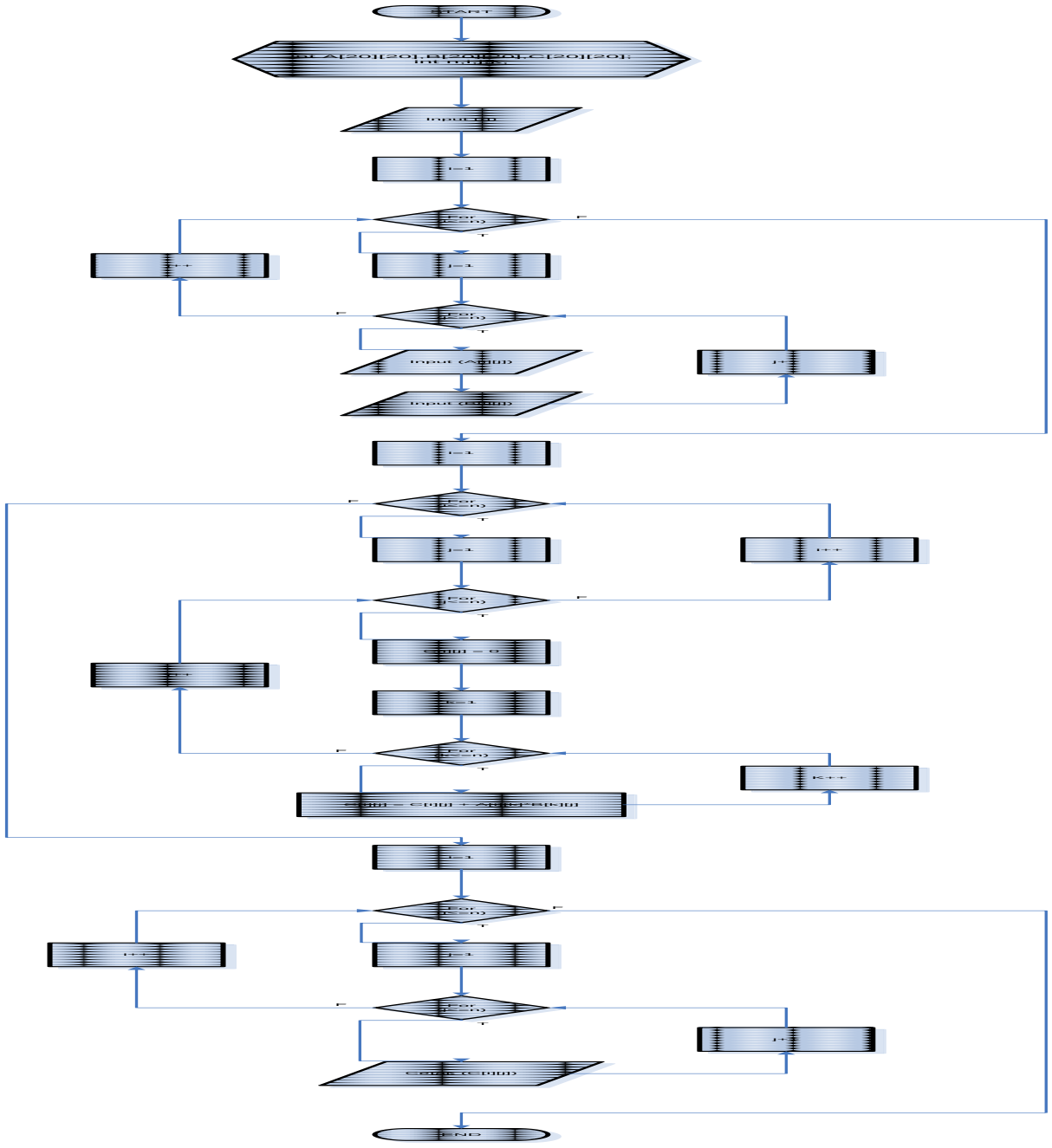
IV. HASIL DAN PEMBAHASAN

Penerapan metode-metode tersebut adalah dengan menjawab soal dari operasi perkalian matriks seperti tertulis di bawah ini:

$$\mathbf{A}\begin{vmatrix}1 & 2\\ 3 & 1\end{vmatrix} \; x \; \mathbf{B}\begin{vmatrix}2 & 2\\ 1 & 3\end{vmatrix} = \mathbf{C}\begin{vmatrix}4 & 8\\ 7 & 9\end{vmatrix}$$

Matriks A x Matriks B = Matriks C

a. Flowchart Operasi Perkalian Matriks



Sumber: Hasil Penelitian (2014)

Gambar 4: Flowchart Operasi Perkalian Matriks

b. Deskripsi Flowchart Operasi Perkalian Matriks

Perkalian matriks pada kasus ini adalah perkalian dua matriks, yaitu matriks A matriks B. Ordo matriks pada perkalian ini adalah 2 x 2, berarti dua kolom dan dua baris. Untuk flowchart yang penulis desain diawalali dengan start, selanjutnya yaitu memberikan kesempatan kepada user untuk memasukkan jumlah variabel array matriks A dan matriks B untuk menentukan jumlaha ordo pada matriks, yang ditampung oleh variabel n. Selanjutnya pengisian variabel lainnya yang berfungsi untuk mengontrol jalannya program matriks ini, yaitu pada variabel i, j, dan k, semuanya bertipe integer. Flowchart akan dijalankan berurutan, mulai *start* sampai *end*. Bila ada kondisi, akan diuji untuk melangkah ke instruksi berikutnya apakah menuju ke instruksi *True* (T) atau *False* (F), atau dapat dikatakan juga apakah *looping* atau keluar dari badan *looping*.

Pembahasan pada kasus ini, setelah start menuju instruksi Input n, di sini *user* diberikan kesempatan untuk memasukkan variabel n atau menentukan jumlah ordo. Berikan nilai 2 pada n sesuai soal pada kasus, yaitu perkalian matrik ordo 2 kali ordo 2. Variabel n berfungsi untuk mengontrol kondisi looping atau batas looping, yang akan digunakan pada kesempatan seperti instruksi For (i=<n) yang menguji nilai i, sebelumnya ada instruksi i=1, ini berarti memberikan nilai konstanta 1 pada variabel i. Pada kasus ini variabel i, berfungsi untuk menentukan jumlah baris, karena matriks ordo 2, maka barus hanya aka nada 2 atau maksimal <= 2.

Pada paragraph di atas, instruksi for (i<=n) pengujian nilai i, artinya apakah nilai i<=n atau i<=2, bila True maka langkah berikutnya ke instruksi j=1, artinya berikan nilai 1 pada variabel j. Variabel j selanjutnya berfungsi untuk menentukan jumlah kolom. Bila hasil pengujian False, maka keluar dari badan looping For (i<=n) dan menuju ke intruksi i=1 setelah badan looping For (i<=n).

Kondisi saat ini, berarti ada pada baris 1 dan kolom 1, berikutnya instruksi yang dijalankan adalah intruksi For (j<=n), apakah variabel j<=n. Bila hasil pengujian True maka instruksi berikut yang dijalankan adalah mengisi elemen pada **baris 1 dan kolom 1** pada matriks A dan matriks B, instruksinya adalah input A[i,j] dan input B[i,j], setelah itu ada intruksi j++, artinya variabel j otomatis bertambah 1, berarti sekarang variabel j=2, sementara posisi baris atau variabel i masih di 1 atau baris 1.

Setelah merubah variabel j=2, instruksi berikutnya adalah menuju ke intruksi For (j<=n) kembali. Terakhir variabel j=2, dengan kondisi 2<=2 berarti True, maka menuju kembali ke instruksi input A[i,j] dan input B[i,j], maka intruksi berikut yang dijalankan adalah mengisi elemen pada **baris 1 dan kolom 2** pada matriks A dan matriks B. Selanjutnya kembali lagi ke instruksi j++, artinya variabel j otomatis bertambah 1, berarti sekarang variabel j=3. Variabel j=3 diuji kembali atas intruksi For (j<=n), apakah 3<=2 hasilnya adalah False. Ingat variabel n=2, sesuai instruksi awal input n dan setelahnya atau tidak ada perubahan variabel n.

Keluar dari instruksi kondisi For (j<=n) menuju intruksi i++, berarti sekarang variabel i=2. Variabel i selanjutnya akan diuji oleh instruksi kondisi For (i<=n). Hasilnya adalah True, karena 2<=2. Langkah berikutnya menuju instruksi j=1, ini berarti kembali variabel j menjadi 1. Setelahnya menuju instruksi input A[i,j] dan input B[i,j], saat ini berarti mengisi elemen matriks **baris 2 kolom 1**. Setelah menuju instruksi j++ kembali, variabel j menjadi 2. Masuk kembali ke badan looping For (j<=n), mengisi kembali elemen matriks atas instruksi input A[i,j] dan input B[i,j], sekarang pada elemen **baris 2 kolom 2**.

Seperti pada penjelasan sebelumnya, setelah input A[i,j] dan input B[i,j], berarti menuju intruksi j++, selanjutnya variabel j diuji atas intruksi kondisi For (j<=n), 3<=2 hasilnya False. Hasilnya False berarti menuju instruksi i++, sekarang i=3, selanjutnya menuju instruksi kondisi For (i<=n), 3<=2 hasilnya False, maka sesuai desain algorithma, langkah berikutnya adalah menuju instruksi i=1, artinya memberikan nilai konstanta 1 pada i.

Instrusksi-instruksi tadi sudah dapat membuktikan pengisian elemen-elemen matriks A dan matriks B, mulai dari baris 1 kolom 1, baris 1 kolom 2, baris 2 kolom 1, dan terakhir baris 2 kolom 2, secara berurutan. Untuk memperjelas, sebagai kesimpulan pengisian elemen matriks penulis membuatkan tabel dari proses pengisian elemen matriks tersebut, yang dijelaskan pada Tabel 1.

Tabel 1. Proses Pengisian Elemen Matriks

Input (n)	i=1	For (i<=n)	HB	j=1	For (j<=n)	HB	Input (A[i][j])	Input (B[i][j])	j++	i++
2	i=1	1<=2	T	j=1	1<=2	T	A[1][1] = 1	B[1][1] = 2	1++	
					2<=2	T	A[1][2] = 2	B[1][2] = 2	2++	
					3<=2	F	–	–	–	1++
		2<=2	T	j=1	1<=2	T	A[2][1] =3	B[2][1] = 1	1++	
					2<=2	T	A[2][2] = 1	B[2][2] = 3	2++	
					3<=2	F	–	–	–	2++
		3<=2	F	–	–	–	–	–	–	–

Sumber: Hasil Penelitian (2014)

Berikutnya desain dibuat untuk melakukan operasi perkalian matriks matriks A dan matriks B, dan hasilnya ditampung oleh matriks C. Terakhir instruksi yang dikerjakan adalah instruksi i=1, artinya variabel i diberi nilai 1 kembali, untuk keperluan proses operasi perkalian matriks. Setelah instruksi i=1 dijalankan, berikutnya menguji varibel i atas instruksi For (i<=n), bila hasil True akan menuju ke perintah j=1, berikutnya pengujian pada variabel j dengan perintah For (j<=n), jika benar isi matriks C, C[i,j]=0, selanjutnya isi variabel k, k=1, selanjutnya pengujian pada variabel k, For (k<=n), jika benar isi matriks C, C[i,j]=C[i,j]+A[i,k]*B[k,j], selanjutnya menjalankan instruksi k++, kembali ke instruksi pengujian variabel k, For (k<=n). Jika diterapkan pada kasus, berarti variabel i berisi nilai 1, For (1<=1) hasilnya True, maka variabel berisi nilai 1, For (1<=1) hasilnya True, berikutnya C[1,1]=0, lalu isi variabel k dengan 1, For (1<=1) hasilnya True, lalu lakukan operasi matriks A dan B, simpan hasilnya pada matriks C, C[1,1]=C[1,1]+A[1,1]*B[1,1].

Setelah menyelesaikan operasi matriks looping pertama, berikutnya variabel k++, berarti sekarang k berisi 2, masih pada looping For (k<=n) berarti For (2<=2) hasilnya True, kembali memproses operasi matriks C[i,j]=C[i,j]+A[i,k]*B[k,j], untuk memudahkan penjelasan operasi perkalian matriks, penulis simpulkan dan jelaskan pada Tabel 2.

Tabel 2. Proses Looping dan Hasil Operasi Perkalian Matriks

i=1	For (i<=n)	HB	j=1	For (j<=n)	HB	C[i][j] = 0	k=1	For (k<=n)	HB	C[i][j] = C[i][j] + A[i][k]*B[k][j]	K++	j++	i++
i=1	1<=2	T	j=1	1<=2	T	C[1][1] = 0	k=1	1<=2	T	C[1][1] = 0+ 1*2 = 2	1++		
								2<=2	T	C[1][1] = 2 + 2*1 = 4	2++		
								3<=2	F	–	–	1++	
				2<=2	T	C[1][2] = 0	k=1	1<=2	T	C[1][2] = 0+ 1*2 = 2	1++		
								2<=2	T	C[1][2] = 2 + 2*3 = 8	2++		
								3<=2	F	–	–	2++	
				3<=2	–	–	–	–	–	–	–	–	1++
	2<=2	T	j=1	1<=2	T	C[2][1] = 0	k=1	1<=2	T	C[2][1] = 0+ 3*2 = 6	1++		
								2<=2	T	C[2][1] = 6 + 1*1 = 7	2++		
								3<=2	F	–	–	1++	
				2<=2	T	C[2][2] = 0	k=1	1<=2	T	C[2][2] = 0+ 3*2 = 6	1++		
								2<=2	T	C[2][2] = 6 + 1*3 = 9	2++		
								3<=2	F	–	–	2++	
				3<=2	–	–	–	–	–	–	–	–	2++
	3<=2	F	–	–	–	–	–	–	–	–	–	–	–

Sumber: Hasil Penelitian (2014)

Selesai melakukan operasi perkalian matriks, kembali variabel i diberi nilai 1, berikutnya adalah proses pencetakan hasil dari operasi perkalian matriks. Untuk memudahkan proses loopingnya, penulis menjelaskan dalam bentuk tabel, yaitu pada Tabel 3.

Tabel 3. Proses Pencetakan Hasil Operasi Perkalian Matriks

i=1	For (i<=n)	HB	j=1	For (j<=n)	HB	Cetak (C[i][j])	j++	i++
i=1	i<=2	T	j=1	1<=2	T	4	1++	
				2<=2	T	8	2++	
				3<=2	F	–	–	1++
	2<=2	T	j=1	1<=2	T	7	1++	
				2<=2	T	9	2++	
				3<=2	F	–	–	2++

Sumber: Hasil Penelitian (2014)

c. Pseudecode Operasi Perkalian Matriks

Untuk keperluan pembuatan aplikasi dari program matriks, penulis membuatkan pseudocode dari operasi matriks tersebut, yang selanjutnya dapat ditranslate ke bahasa pemorgraman computer, seperti C++, Visual Basic, Java atau Bahasa pemrograman lainnya.

```
A[2,2}, B[2,2], C[2,2] = 0;
n, i, j, k = 0;
write ('Masukkan Ukuran Matriks: ');
read n;
j=1;
for i <= n;
      { j=1;
        for j <= n;
                  { write ('Matriks A: '); read A[i,j];
                    write ('Matriks B: '); read B[i,j]; }}
i=1;
for i <= n;
        { j=1;
        for j <= n;
                  { C[i,j] = 0;
                  k=1;
                  for k <= n;
                            { C[i,j] = C[i,j] + A[i,k] * B[k,j]; }}}
write ('Hasil Matriks: ');
i=1;
for I <= n;
        { j=1;
        for j <= n;
                  { write ('Matriks C: ');
                    write C[i,j]; }}
```

d. Konversi Pseudecode Operasi Perkalian Matriks pada Bahasa C

Bahasa pemrograman computer menggunakan Bahasas C untuk aplikasi perkalian matriks adalah sebagai berikut:

```
        int A[2][2],B[2][2],C[2][2];
        int n;
        int i,j,k;
        printf("Masukkan ukuran matriks : ");scanf("%d",&n);
        for (i=1;i<=n;i++){
                  for (j=1;j<=n;j++){
                            printf("A[%d][%d]                    :
",i,j);scanf("%d",&A[i][j]);
                            printf("B[%d][%d]                    :
",i,j);scanf("%d",&B[i][j]); }
                  //printf("x"); }
//printf("x");
        for (i=1;i<=n;i++){
                  for (j=1;j<=n;j++){
                            C[i][j]=0;
                            for (k=1;k<=n;k++){
                                      C[i][j]      =      C[i][j]      +
A[i][k]*B[k][j]; }}}
        printf("\nHASIL\n");
        for (i=1;i<=n;i++){
                  for (j=1;j<=n;j++){
        printf("C[%d][%d] : %d\n",i,j,C[i][j]); }}
        getch();
        return 0; }
```

V. KESIMPULAN

Desain algoritma dengan flowchart yang penulis buat adalah salah satu dari desain algorithma yang dapat dibuat untuk operasi perkalian matriks beracuan pada efisien algoritma seperti penentuan variabel n, j, k atau variabel array A[i,j], B[i,j] dan C[i,j]; penentuan kondisi looping, dan histori memori dapat penulis jelaskan dan pertanggungjawabkan pada tabel-tabel pembahasan dalam mendeskripsikan desain algorithma menggunakan metode flowchart.

REFERENSI

[1] Munir, Rinaldi. Algoritma & Pemrograman. Dalam Bahasa Pascal dam C. Bandung: Penerbit Informatika Bandung. 2011.
[2] Ngoen, Thompson Susabda. ALGORITMA DAN STRUKTUR DATA Bahasa C. Jakarta: Penerbit Mitra Wacana Media. Edisi Pertama. 2009.
[3] Robertson, Lesiey Anne. *Simple Program Design*. *A Step-by-Step Approach*. *Fourth Edition*. Hongkong: *Course Technology*. 2004.
[4] Suryadi, H.S., dkk.. Teori Dan Soal Pendahuluan Aljabar Linier. Serial Matematika. Jakarta: Ghalia Indonesia. 1990
[5] Sutanta, Edhy. Algoritma Teknik Penyelesaian Permasalahan Untuk Komputasi. Yogyakarta.: Graha Ilmu. 2004.
[6] Yahya, Yusuf, dkk. Matematika Dasar Untuk Perguruan Tinggi. Serial Matematika & Komputer ASKI. Cetakan Keduabelas. Bogor: Ghalia IKAPI. 2005.
[7] http://www.akmi-baturaja.ac.id/wp-content/uploads/2012/07/Logika-dan-Algoritma.pdf

Rini Nuraini, S.T., M.Kom. Tahun 1997 lulus dari Program Strata Satu (S1) dengan program studi Teknik Komputer (S.T.) Universitas YARSI Jakarta dan Tahun 2010 Program Strata Dua (S2) dengan program studi Ilmu Komputer (M.Kom). Menjadi Dosen sejak tahun 2004 hingga saat ini, di beberapa lembaga pendidikan perguruan tinggi swasta di Jakarta dan Karawang, pada program studi Teknik Informatika, Sistem Informasi, Manajemen Informatika, Teknik komputer, dan Akuntansi Komputer, untuk matakuliah yang berkaitan dengan komputasi, baik *software* ataupun *hardware*, diantaranya: Struktur Data, Intelegensia Semu, Manajemen Sains, Elektronika Dasar, dll. Sudah tersertifikasi dosen tahun 2011 dengan Jabatan Fungsional Akademik Lektor di AMIK BSI Jakarta. Pernah mendapatkan hibah dikti "Penelitian Dosen Muda" tahun 2009. Tulisan yang pernah dipublikasikan diantaranya: *Algorithm Design Of Definite Integration By Using Flowchart Method; Algorithm Analysis Of Definite Integration By Using Desk Check Method; dll*