ANALISIS ALGORITMA DEFINITE INTEGRATION MENGGUNAKAN METODE DESK CHECK

Rini Nuraini[1)]

[1)]Staf Pegajar AMIK BSI JAKARTA
Jl. R.S. Fatmawati No.24 – Pondok Labu – Jakarta Selatan,
email: rininooraini@yahoo.com

ABSTRACT

We usually experience difficulties when determining output of a pseudocode algorithm, especially in repetition control structure or nested loop algorithm. They are usually complicated. We got difficulties to guess the logic or specifiy the output. For instance, what is the end result of the looping? Is it true or false? Where is the start and finish point of the looping? Where is the boundary of looping body? Should the looping continue or stop? Did looping occure during a boolean result or condition is true? Did looping occure until condition is true? In addition to this, because of limitation human mind to remember or to do the looping, we need some tools to analyze it. Based on the mentioned problems, author seek a design or method, easy way to analyze variables either input or output variables along with the algorithm logic in a algorithm psuedocode. This is the author's purpose of designing table, which is called desk check. In this case, function of the desk check is documenting the history of several changes from a number of analyzed variables, either input or output variables along with logic by using table. The purpose of this article is analyze the answer of Algorithm Design of Definite Integration by Using Flowchart Method, as well as verification of example problem candidate, namely definite integration.

Key Words: *Algorithm Analysis, Definite Integration, Desk Check, Looping*

1. PENDAHULUAN

Ada beberapa langkah dasar yang perlu untuk diikuti dalam pembuatan suatu algoritma, antara lain adalah: pernyataan masalah; membangun model dari suatu masalah; merancangan 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 computer yang melaksanakannya; notasi algoritma dapat diterjemahkan ke dalam berbagai bahasa pemrograman; apapun bahasa pemrogramannya, output yang akan dikeluarkan sama, karena algoritmanya sama.

Permasalahan atau soal-soal *science* khususnya kalkulus, 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 kalkulus, yaitu Integral Tentu 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. 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. 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.

2. TINJAUAN PUSTAKA

1. Kalkulus: Integral Tentu

Jr., Frank Ayres, et.al. (1999:130): notasi sigma, huruf besar Yunani S melambangkan penjumlahan berulang. Contoh:

a. $\sum_{j=1}^{5} j = 1 + 2 + 3 + 4 + 5 = 15$

b. $\sum_{i=0}^{3} (2i + 1) = 1 + 3 + 5 + 7$

Secara umum, jika f adalah fungsi terdefinisi pada bilangan bulat dan jika n dan k adalah bilangan bulat sedemikian rupa sehingga n ≥ k, maka:

$$\sum_{j=k}^{n} f(j) = f(k) + f(k+1) + \cdots + f(n)$$

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 menurut Donald E. Knuth, dalam http://www.akmi-baturaja.ac.id/wpcontent/uploads/2012/07/Logika-dan-Algoritma.pdf, yaitu:

1. Finiteness (Keterbatasan)
   Algoritma 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. METODE PENELITIAN

1. Flowchart

Menurut Edhy Sutanta (2004, hal. 28), 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. Manfaat yang akan diperoleh bila menggunakan flowchart dalam pemecahan masalah komputasi:

1. Terbiasa berfikir secara sistematis dan terstruktur
2. Mudah mengecek dan menemukan bagian-bagian prosedur yang tidak valid dan bertele-tele
3. Prosedur akan mudah dikembangkan

2. *Repetition Control Stuctures*

Robertson, Lesiey Anne (2004:54), *outline:*

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

Ngoen, Thompshon Susabda (2009:127-148), bahasa C menyediakan tiga instruksi untuk melakukan proses pengulangan: for, while, dan do while. Ketiga intruksi ini memiliki karakteristik masing-masing.

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.

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.

c. Instruksi Do While

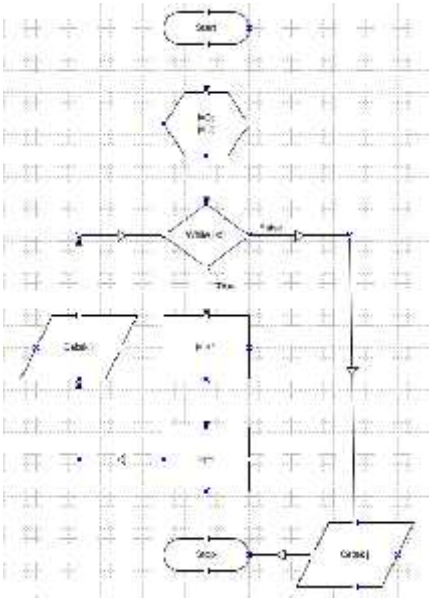
Instruksi *do while* ialah instruksi untuk melakukan proses pengulangan yang pemeriksaan syarat pengulangannya dilakukan pada akhir proses. Instruksi *do while* umumnya digunakan untuk melakukan proses pengulangan yang belum diketahui frekuensi pengulangannya tetapi pasti dikerjakan minimal satu kali.

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.

4. PEMBAHASAN DAN HASIL PENELITIAN

a. Flowchart Soal 1



Gambar 1

Penerapan metode-metode tersebut adalah dengan menjawab soal-soal dari materi Integral Tertentu diambil dari Schaum's Outline Of KALKULUS Edisi Keempat seperti tertulis di bawah ini:

1. Soal 1: $\sum_{j=1}^{5} j = 1 + 2 + 3 + 4 + 5 = 15$

b. Pseudecode Soal 1

```
i=0;
j=1;
while I < 5
        i = I + 1
        i = j + i
        cetak i
cetak j
```

c. Desk Check Table Soal 1

Tabel 1

I	J	While i<5	Hasil Boolean	I=i+1	j=j+i	Cetak i	Cetak j
0	0	0<5	T	1=0+1	1=0+1	1	
		1<5	T	2=1+1	3=1+2	2	
		2<5	T	3=2+1	6=3+3	3	
		3<5	T	4=3+1	10=6+4	4	
		4<5	T	5=4+1	15=10+5	5	
		5<5	F	-	-	-	15

d. Deskripsi Analisa Desk Check Table Soal 1

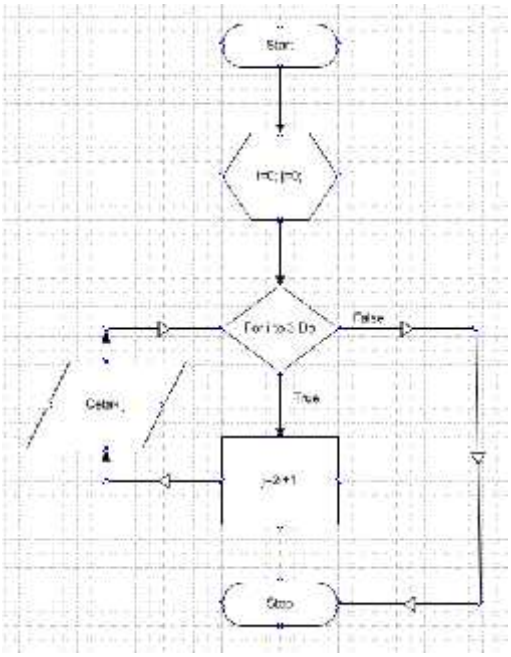
Pahami dahulu logika dari instruksi while, bahwa while akan berulang bila hasil pengecekan boolean menghasilkan nilai true atau bukan nol.

Perhatikan kolom Hasil Boolean pada Tabel 1, selama bernilai True (T) dilakukan pengulangan atas badan pengulangan yang terdiri dari instruksi: i=i+1, j=j+1, dan cetak i (perhatikan pula *flowchart* dan *pseudecode*-nya).

Pada saat bernilai False, maka akan berhenti berulang. Keluar dari kondisi berulang, ada instruksi untuk mencetak nilai dari variabel j (cetak j), terakhir tercetak adalah 15, berdasarkan konsep variabel, bahwa nilai variabel adalah nilai terakhir dari hasil komputasi. Perhatikan nilai variabel j berubah-rubah dari 1 berikutnya 3, 6, 10, dan terakhir 15. Variabel j diberi nilai awal 0 (j=0).

True atau *False*, berdasarkan hasil pengujian pada *while*, dengan urutan intruksi sebagai berikut: Apakah 0<5? Hasil Boolean True; Apakah 1<5? Hasil Boolean True; Apakah 2<5? Hasil Boolean True; Apakah 3<5? Hasil Boolean True; Apakah 4<5? Hasil Boolean True; Apakah 5<5? Hasil Boolean False, pengulangan berhenti. Variabel i berubah, dari 0, berikutnya 1, 2, 3, 4, 5 karena ada instruksi di dalam badan pengulangan, yaitu: i = i + 1. Variabel i diberi nilai awal 0 (i=0).

a. Flowchart Soal 2



Gambar 2

2. Soal 2: $\sum_{i=0}^{3}(2i+1) = 1 + 3 + 5 + 7$

b. Pseudecode Soal 2

```
i=0;
j=0;
for i to 3 do
        j = 2i + 1
        cetak j
```

c. Desk Check Table Soal 2

Tabel 2

j	I	for 0 to 3 do	Hasil Boolean	j=2.i+1	Cetak j
0	0	0 to 3	T	1=2.0+1	1
		1 to 3	T	3=2.1+1	3
		2 to 3	T	5=2.2+1	5
		3 to 3	T	7=2.3+1	7
		4 to 3	F	-	-

d. Deskripsi Analisa Desk Check Table Soal 2

Pahami dahulu logika dari instruksi for, bahwa for digunakan untuk melakukan proses pengulangan yang frekuensi pengulangannya telah diketahui sebelum proses pengulangan dimulai, jumlah pengulangannya sudah dapat ditebak, yaitu berdasarkan inisial awal sampai inisial akhir, berarti diawali oleh inisial awal dan diakhiri oleh inisial akhir. Pengulangan for, sama halnya dengan while, yaitu selama hasil pengecekan Boolean bernilai *True* (T) atau bukan bernilai nol, proses berulang.

Perhatikan kolom Hasil Boolean pada Tabel 2, selama bernilai True dilakukan pengulangan atas badan pengulangan yang terdiri dari: j=2.i+1, dan cetak j (perhatikan pula *flowchart* dan *pseudecodenya*).

Pada saat bernilai *False*, maka akan berhenti berulang. Keluar dari kondisi berulang, tidak ada instruksi yang harus dilakukan lagi. Untuk mencetak nilai dari variabel j sudah dilakukan saat pengulangan, sehingga *output*nya 1, 3, 5, dan 7. Berbeda dengan *while*, yang memiliki intruksi untuk mengubah nilai variabel i, karena pada for variabel i berubah otomatis, defaultnya adalah berubah atau bertambah 1, jadi perubahan variabel i adalah 0 berikutnya 1, 2, dan 3. Variabel i diberi nilai awal 0 (i=0).

True atau *False*, berdasarkan hasil pengujian pada for, dengan urutan intruksi sebagai berikut: Apakah niali 0 diantara 0 dan 3? Hasil Boolean *True*; Apakah nilai 1 diantara 0 dan 3? Hasil Boolean *True*; Apakah nilai 2 diantara 0 dan 3? Hasil Boolean *True*; Apakah nilai 3 diantara 0 dan 3? Hasil Boolean True; Apakah nilai 4 diantara 0 dan 3? Hasil Boolean False, pengulangan berhenti.

Perubahan variabel i berdampak pada instruksi j=2.i+1, sehingga hasilnya perubahan variabel j adalah seperti dijelaskan pada Tabel 2.

5.DAFTAR PUSTAKA:

[1] Jr., Frank Ayres, et.al. (1999). *Schaum's Outlines of Theory and Problems of Calculus. Fourth Edition*. Kalkulus. Edisi Keempat. Alih Bahasa: Nur Danarjaya, M.Sc. Editor: Amalia Safitri, S.TP., M.Si. Jakarta: Penerbit Airlangga.

[2] Ngoen, Thompson Susabda. (2009). *ALGORITMA DAN STRUKTUR DATA Bahasa C*. Jakarta: Penerbit Mitra Wacana Media. Edisi Pertama.

[3] Robertson, Lesiey Anne. 2004. *Simple Program Design*. *A Step-by-Step Approach*. *Fourth Edition*. Hongkong: *Course Technology*.

[4] Sutanta, Edhy. (2004), *Algoritma Teknik Penyelesaian Permasalahan Untuk Komputasi*. Graha Ilmu. Yogyakarta.

[5] http://www.akmi-baturaja.ac.id/wp-content/uploads/2012/07/Logika-dan-Algoritma.pdf