STUDI AWAL PERANCANGAN TABEL ANALISA DAN *FLOWCHART* UNTUK METODE ALGORITMA *LOOPING*

Rini Nuraini, ST, M.Kom
Staf Pengajar AMIK BSI
Jl. R.S. Fatmawati No.24 - Pondok Labu – Jakarta Selatan
www.bsi.ac.id

ABSTRACT

Sometimes in determining the output from an application program having trouble, especially in the looping algorithm, which is usually considered to be complicated. There are a number of variables that must be analyzed along with the algorithm. Another issue is the limitations of human reason to remember all the computational results of all the variables change with the algorithms used in an application program. Under the conditions of these problems, the authors find the design as a method, how easy to analyze the variables in the algorithm and its application program, the authors designed a table that is the purpose of the authors table berinama flowchart analysis combined with the already known standards in the world of computing to analyze each application program.

Table analysis to analyze variables along with the documented history of any change of variables while the flowchart to help illustrate the limit and start looping, from any application. Programming language Borland C + + that will use the author in writing this article as a tool to prove the truth of which is the result of computing. The author took a simple application example, the goal is to facilitate the explanation, or convey the concept of the draft results of the analysis table.

Experience author in teaching the course Analysis and Design of Algorithms or materials relating to the algorithm, almost all students can apply the analysis is complete the design of the table along with the flowchart in each application program, either simple or difficult applications such as arithmetic applications, sorting, and searching. The result is very satisfactory, meaning that the results are similar to the results of computation, thus the student is in conformity with the logic of computer logic, the results are very satisfactory, pride, and happiness for the writer to deliver the student ketahap. This design can also serve to train and help facilitate the logic in teaching subjects related to the algorithm.

Keywords: Design tables analysis, flowcharts, algorithms looping, Borland C++.

1. PENDAHULUAN

Penulis sering sekali menemui mahasiswa mengalami kesulitan atau bahkan menghasilkan jawaban yang salah, dalam menentukan *output* dari program aplikasi sekalipun itu aplikasi sederhana. Struktur aplikasi bisa terdiri dari intruksi sequential, pemilihan kondisi (*decision*), *goto*, dan *looping*, juga ditambah dengan sejumlah variabel yang harus dianalisa satu persatu-satu.

Permasalahan lain, adalah keterbatasan nalar manusia untuk mengingat semua hasil komputasi dari semua perubahan setiap variabel beserta logika algoritma yang menyertainya dalam suatu program aplikasi. Berdasarkan kondisi permasalahan tersebut, penulis mencari rancangan atau metode, bagaimana cara mudah untuk menganalisa khususnya logika algoritma *looping* sesuai pembahasan pada setiap program aplikasi, inilah tujuan penulis merancang suatu tabel yang diberinama

tabel analisa dikombinasikan dengan *flowchart* yang sudah standar diketahui di dunia komputasi untuk menganalisa setiap program aplikasi.

Selanjutnya, tabel analisa difungsikan untuk menganalisa variabel beserta mendokumentasikan *history* dari setiap perubahan variabel sedangkan *flowchart* untuk membantu mengambarkan logika algoritma, misalkan batas akhir dan awal logika dari variabel yang harus dikerjakan berulang-ulang dari setiap aplikasi. Sebagai contoh melakukan pengulangan pencetakan dari 1 2 3 4 5 berarti awal pengulangan adalah 1 dan akhir pengulangan adalah 5, sedangkan perubahan varibael setiap terjadi pengulangan adalah 1, dan hanya terdiri dari satu varibel yang berubah, sangat sederhana.

Kondisi akan menjadi rumit, bila variabel yang akan berubah lebih dari 2, bahkan 3, atau mungkin 4 dan seterusnya, ditambah dengan berbagai struktur logika algoritma baik *decision*, *looping* dan sebagainya yang menyertainya, bahkan kombinasi diantaranya pada setiap aplikasinya.

Bahasa pemrograman Borland C++ yang akan penulis gunakan dalam penulisan artikel ini sebagai *tools* untuk membuktikan kebenaran sebagai hasil olah dari komputer yang penulis sebut sebagai hasil komputasi.

Penulis mengambil contoh aplikasi sederhana, tujuannya adalah untuk memudahkan penjelasan atau menyampaikan konsep dari rancangan tabel analisa tersebut, yang merupakan tujuan utama dari artikel ini. Walau pun demikian tabel ini sudah diuji coba pada saat penulis mengajar, pada aplikasi rumit, hasilnya dapat terbukti benar sesuai hasil komputasi, karena pada dasarnya aplikasi serumit apapun konsep logika tidak berubah atau berbeda dengan aplikasi sederhana, yaitu terdiri dari struktur logika algoritma *decision* dan *looping*, bisa jadi ditambah dengan intruksi *goto*. Hal yang menjadi rumit pada aplikasi lebih kepada banyaknya variabel yang dianalisa dan baik decision atau looping yang juga banyak digunakan dan bahkan bersarang (*nested*). Konsep *decision* dan *looping*, baik pada aplikasi sederhana ataupun rumit adalah sama, arti kata lain tidak ada perbedaan konsep.

Fokus pada pembahasan, yaitu logika algoritma *looping* terdiri dari 3 jenis: melakukan pengulangan selama kondisi benar (*while*); melakukan pengulangan selama kondisi salah atau arti kata lain melakukan pengulangan sampai kondisi benar (*do while*); dan melakukan pengulangan mulai kondisi start sampai kondisi stop (*for*).

Untuk keperluan *crosscheck* sebagai pembuktian atas analisa dengan tabel analisa dan *flowchart*, penulis menguji coba langsung pada komputer dari semua aplikasi program yang akan dianalisa dengan menggunakan bahasa pemrograman borland c++, selanjutnya membandingkan hasilnya, bila terjadi perbedaan, hasil komputasi dengan hasil tabel analisa, yang penulis anggap benar adalah hasil komputasi, karena komputer sudah sangat teruji kebenarannya dan kehandalannya, penulis mengulangi kembali untuk mengoreksi tabel analisa, biasanya memang masih ditemui ada kesalahan dalam menganalisa. Hal ini yang mengharuskan mahasiswa bila merancang tabel analisa untuk menggunakan pinsil, sehingga bisa dengan mudah menghapus atau menambah kolom, bahkan lebih baik lagi menggunakan aplikasi *excel*.

Pengamalaman penulis dalam mengajar matakuliah Analisa Perancangan Algoritma ataupun materi yang berkaitan dengan logika, hampir semua mahasiswa dapat menerapkan rancangan tabel analisa tersebut lengkap beserta *flowchart* pada setiap program aplikasi baik aplikasi sederhana atau sulit seperti aplikasi-aplikasi aritmatika, *sorting* dan *searching*, dan hasilnya sama dengan hasil komputasi, dengan demikian logika mahasiswa sudah sesuai dengan nalar komputer, hasil yang sangat memuaskan, kebanggan, dan kebahagiaan bagi penulis mengantakan mahasiswa ketahap tersebut.

Pengalaman lain yang penulis temui adalah bila tidak dibantu dengan *flowchart*, yaitu terjadinya kesalahan melakukan pengulangan atau tidak diketahuinya jumlah pengulangan pada aplikasi tersebut. Berdasarkan hal ini, kesimpulannya adalah harus dibuat pula *flowchart* selain tabel analisanya.

Berkaitan dengan pembuatan *flowchart*, penulis juga sekaligus ingin memberikan contoh yang benar sesuai kaidahnya, karena selama menulis memeriksa flowchart masih banyak ditemui kesalahan dalam membuat *flowchart*. Contoh kasus, pada penggunaan simbol *decision* hanya satu *output*-nya, padahal sesuai kaidahnya *output* dari *decision* ada dua, yaitu *true* dan *false*.

Rancangan ini juga dapat berfungsi untuk melatih nalar logika supaya sesuai dengan nalar logika komputer dalam menganalisa aplikasi komputasi dan dapat berfungsi untuk membantu memudahkan dalam menjelaskan pada saat mengajar matakuliah yang berkaitan dengan logika algoritma. Hal ini berdasarkan pengalaman dari penulis selama mengajar matakuliah yang berkaitan dengan logika algoritma.

Untuk memudahkan memahami pembahasan, ada beberapa hal teoritis yang perlu diketahui atau dipahami terlebih dahulu. Penulis mengangkat hal-hal tersebut sebagai berikut:

1.1. Pernyataan Looping Bahasa Pemrograman Borland C++

Menurut Abdul Kadir (2001, hal.190-204) pernyataan dari masing-masing *looping* dalam pemrograman Borland C++ adalah sebagai berikut:

a.Pernyataan While

Pernyataan while merupakan salah satu pernyataan yang berguna untuk memproses suatu pernyataan atau beberapa pernyataan beberapa kali. Bentuk pernyataan ini:

```
While (ungkapan)
    Pernyataan;
```

Bagian pernyataan tentu saja bisa berupa pernyataan majemuk, sehingga bentuknya menjadi seperti beikut:

```
While (ungkapan)
{
    Pernyataan1;
    Pernyataan2;
    …
    PernyataanN;
}
```

Bagian pernyataan yang mengikuti while akan dieksekusi selama ungkapan pada while bernilai benar (tidak sama dengan nol). Perlu diketahui, pengujian terhadap ungkapan pada while dilakukan sebelum bagian pernyataan. Oleh karena itu ada kemungkinan bagian pernyataan pada while tidak dijalankan sama sekali, yaitu kalau kondisi yang pertama kali bernilai salah.

b.Pernyataan Do-While

Pernyataan do_while juga berguna untuk mengulang proses. Formatnya:

```
Do
{
    Pernyataan1;
    Pernyataan2;
    …
    PernyataanN;
} while (ungkapan)
```

Bagian pernyataan1 hingga pernyataanN dijalankan secara berulang sampai ungkapan bernilai salah (sama dengan nol). Namun berbeda dengan while, pengujian ungkapan dilakukan di belakang (setelah bagian pernyataan).

c.Pernyataan For

Pernyataan for juga berguna untuk mengulang pengeksekusian terhadap satu atau sejumlah pernyataan. Bentuknya sebagai berikut:

for (ungkapan1; ungkapan2; ungkapan3)
pernyataan;

Pernyataan di atas identik dengan:

```
ungkapan1;
while (ungkapan2)
    {
        pernyataan;
        ungkapan3;
    }
```

Jadi:
1. Ungkapan1 merupakan pernyataan inisialisasi sebelum masuk ke while
2. Ungkapan2 berlaku sebagai kondisi yang menentukan pengulangan terhadap pernyataan atau tidak
3. Ungkapan3 digunakan sebagai pengatur variabel yang digunakan di dalam ungkapan1

1.2. Definisi Flowchart

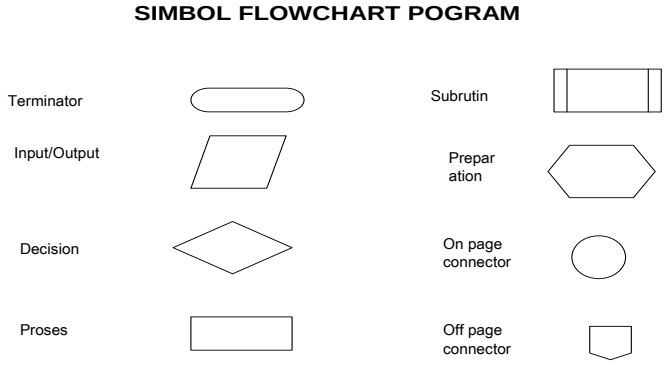
Flowchart biasanya digunakan sebagai salah satu tahapan untuk membangun aplikasi program. Tahapan-tahapan membangun aplikasi program tersebut adalah memahami, menggambarkan *flowchart*, penulisan *psedocode* (algoritma), dan penulisan *coding*. Pada artikel ini, pembuatan flowchart digunakan untuk membantu menganalisa *output* dari aplikasi program, yang dikombinasikan dengan tabel analisa.

Untuk memudahkan memahami logika *looping*, dibutuhkan alat bantu berupa *flowchart* (bagan alir), untuk menggambarkan logika dari program tersebut, yang juga berfungsi untuk menghindari sejak dini timbulnya kesalahan interpretasi bagi orang lain yang merupakan awal kegagalan suatu prosdur yang dikembangkan. Adapun simbol-simbol tersebut diberi nama *flowchart*.

Menurut Eddhy 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

Ir. Yanuar Supardi (2000, hal.150-153) mengatakan berikut adalah simbol-simbol flowchart program:

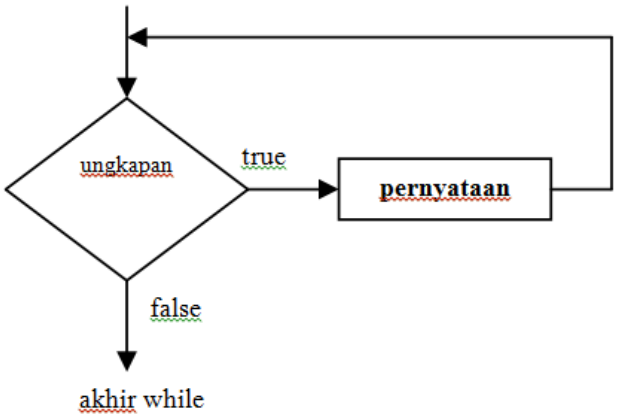


Gambar 1. simbol-simbol flowchart program

1.3. Flowchart Pada Pernyataan Looping Borland C++

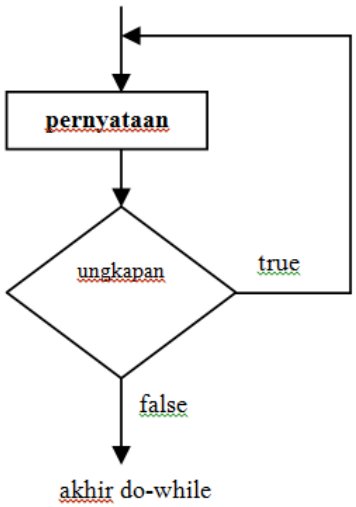
Masing-masing logika *looping* dapat digambarkan dengan menggunakan flowchart. Di bawah ini adalah masing-masing dari flowchart logika *looping* dalam Borland C++:

a.Flowchart Pernyataan Logika While



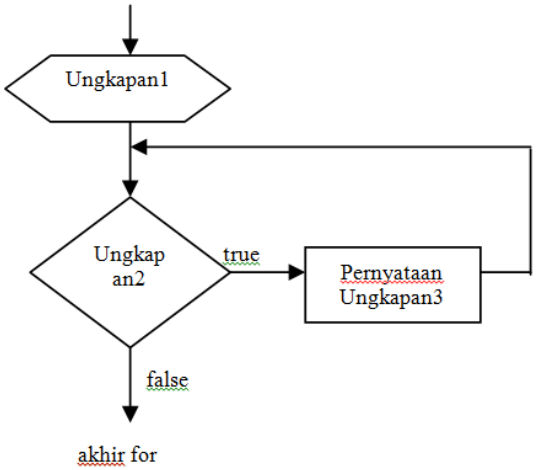
Gambar 2.Flowchart Pernyataan Logika While

b.Flowchart Pernyataan Logika Do-While



Gambar 3.Flowchart Pernyataan Logika Do-While

c.Flowchart Pernyataan Logika For



Gambar 4.Flowchart Pernyataan Logika For

1.4. Fungsi Perintah Input-Output Borland C++

Beberapa instruksi program yang digunakan dalam pembahasan ini adalah sebagai berikut:

a. scanf
Berfungsi untuk menerima *input* atau membaca data dari *keyboard*.

b. getche()
Berfungsi untuk menerima *input* dari keyboard sebanyak 1 (satu) *karakter* dan tidak membutuhkan tombol *enter* untuk mengakhiri proses *input*. Hasil *input* akan ditampilkan ke layar *monitor*.

c. getch()
Berfungi untuk menerima *input* dari *keyboard* sebanyak 1 (satu) *karakter* dan tidak membutuhkan tombol *enter* untuk mengakhiri proses *input*. Hasil *input* tidak akan ditampilkan ke layar *monitor*.

d. cin
Berfungsi untuk menerima *input* atau membaca data dari *keyboard*.

e. printf
Berfungsi untuk menampilkan data atau meletakkan suatu informasi ke layar *monitor*.

f. puts()
Berfungsi khusus untuk menampilkan data *string* ke layar *monitor*. *String* yang ditampilkan secara otomatis akan diakhiri dengan pindah baris.

g. putchar()
Berfungsi khusus untuk menampilkan sebuah *karakter* ke layar *monitor* tanpa diakhiri dengan perpindahan baris.

h. cout
Berfungsi untuk menampilkan data atau meletakkan suatu informasi ke layar *monitor*.

2. PEMBAHASAN

Suatu pernyataan *looping* digunakan dalam pembuatan program aplikasi, pedomannya adalah, bila diinginkan menjalankan suatu program secara berulang. Banyak sekali aplikasi yang bisa diimplementasikan dengan *looping*, berikut di bawah ini adalah aplikasi program

dengan menggunakan *looping*, masing-masing aplikasi program dilengkapi dengan *coding* program, flowchart dan tabel analisa.

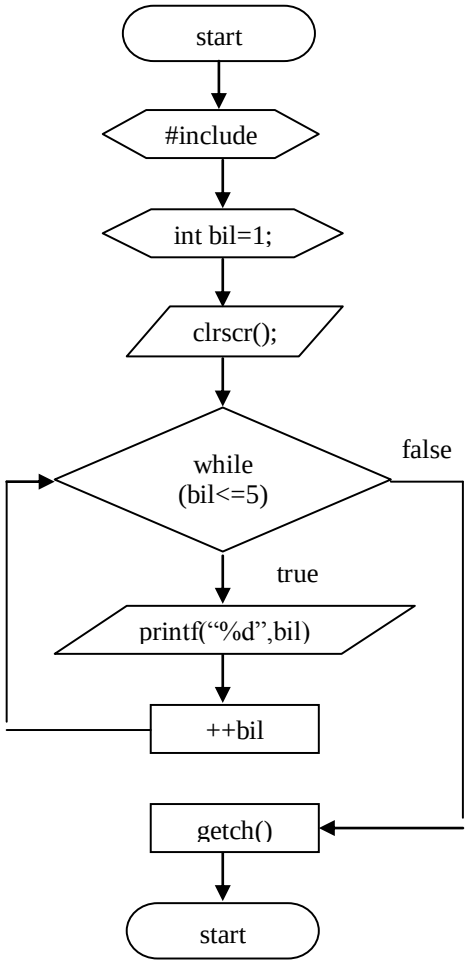
Pada pembahasan, penulis langsung mengaplikasikan pada contoh-contoh aplikasi program sederhana dari logika algoritma *looping*, hal yang disengaja, karena pada studi awal ini lebih fokus pada penyampaian konsep dari ramcangan tabel analisa. Satu contoh saja yang sedikit rumuit, yaitu nested for. Konsep *nested* ini juga berlaku untuk while dan do-while, bahkan kombinasi diantaranya. Selanjutnya, untuk contoh-contoh aplikasi program lebih rumit seperti aplikasi-aplikasi aritmatika (matrik, faktorial, polinomial, dan lain-lain), *sorting*, dan *searching* akan bahas pada artikel berikutnya dengan judul STUDI LANJUT PERANCANGAN TABEL ANALISA DAN *FLOWCHART* UNTUK METODE ALGORITMA BRUTE FORCE, selain berisi tabel analisa dan flowchart dari contoh aplikasi-aplikasi aritmatika seperti faktorial, matrik, dan lain-lain juga terdapat ketentuan-ketentuan dalam membuat tabel analisa.

2.1.Aplikasi Program Looping While

Pembahasan aplikasi program *looping* untuk mencetak **1 2 3 4 5** secara berulang dengan Borland C++, berikut adalah *coding* programnya:

```
#include <stdio.h>
#include <iostream.h>
#include <conio.h>
main()
{
  int bil=1;
  clrscr();
  while (bil<=5)
  {
    printf(" %d",bil);
    ++bil;
  }
  getch();
}
```

a.Flowchart Aplikasi Program Looping While



Gambar 5.Flowchart Aplikasi Program Looping While

b.Tabel Analisa Aplikasi Program Looping While

Tabel 1.

bil= 1	**while (bil<=5)**	**Hasil Boolean**	**printf(" %d",bil)**	**++bil atau bil=bil+1**	**getch()**
1	1<=5	True	**1**	2=1+1	
	2<=5	True	**2**	3=2+1	
	3<=5	True	**3**	4=3+1	
	4<=5	True	**4**	5=4+1	
	5<=5	True	**5**	6=5+1	
	6<=5	False	-	-	proses

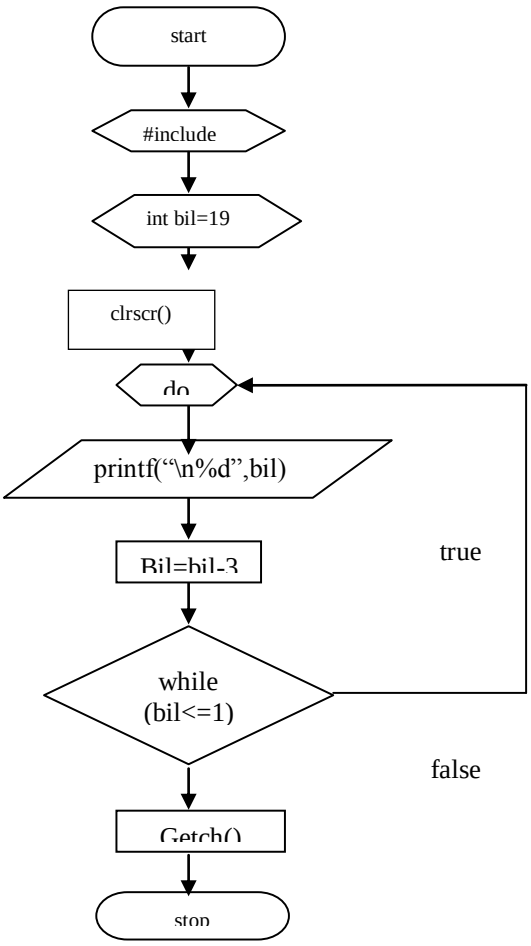
2.2.Aplikasi Program Looping Do-While

Pembahasan aplikasi program *looping* untuk mencetak angka secara berulang dengan Borland C++, berikut adalah *output* program dan *coding* programnya:

19
16
13
10
7
4
1

```
#include <stdio.h>
#include <conio.h>
#include "iostream.h"
main()
{
  int bil =19;
  clrscr();
  do
  {
    printf("\n %d ",bil);
    bil-=3;
   }
   while ( bil>=1 );
 getch();
}
```

a.Flowchart Aplikasi Program Looping Do-While



Gambar 6.a.Flowchart Aplikasi Program Looping Do-While

b.Tabel Analisa Aplikasi Program Looping Do-While

Tabel 2.

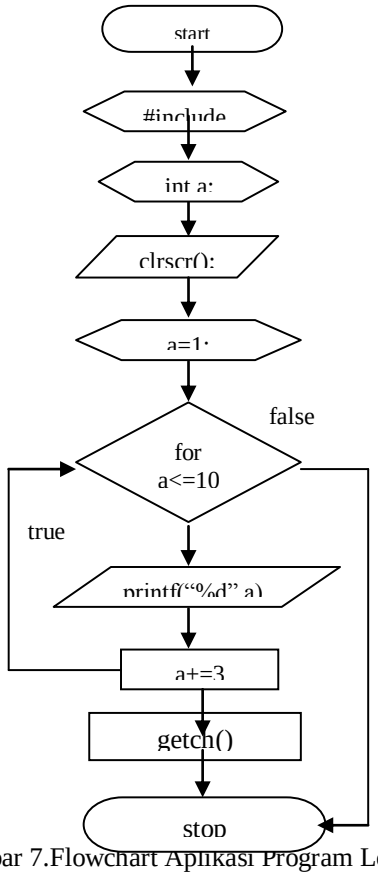
bil=19	Do	printf("\n%d",bil)	bil-=3 atau bil=bil-3	while (bil>=1)	Hasil Boolean	getch()
19		**19**	16=19-3	16>=1	True	
		16	13=16-3	13>=1	True	
		13	10=13-3	10>=1	True	
		10	7=10-3	7>=1	True	
		7	4=7-3	4>=1	True	
		4	1=4-3	1>=1	True	
		1	-2=1-3	-2>=1	False	Proses

2.3.Aplikasi Program Looping For

Pembahasan aplikasi program *looping* untuk mencetak angka **1 4 7 10** secara berulang dengan Borland C++, berikut adalah *coding* programnya :

```
#include <stdio.h>
#include <conio.h>
#include "iostream.h"
main()
{
  int a;
  clrscr();
  for(a= 1; a<= 10; a+=3)
    printf(" %d",a);
  getch();
}
```

a.Flowchart Aplikasi Program Looping For



Gambar 7.Flowchart Aplikasi Program Looping For

b.Tabel Analisa Aplikasi Program Looping For

Tabel 3.

a=1	for a<=10	Hasil Boolean	printf(" %d",a)	a+=3 atau a=a+3	getch()
1	1<=10	True	**1**	4=1+3	
	4<=10	True	**4**	7=4+3	
	7<=10	True	**7**	10=7+3	
	10<=10	True	**10**	13=10+3	
	13<=10	False	-	-	Proses

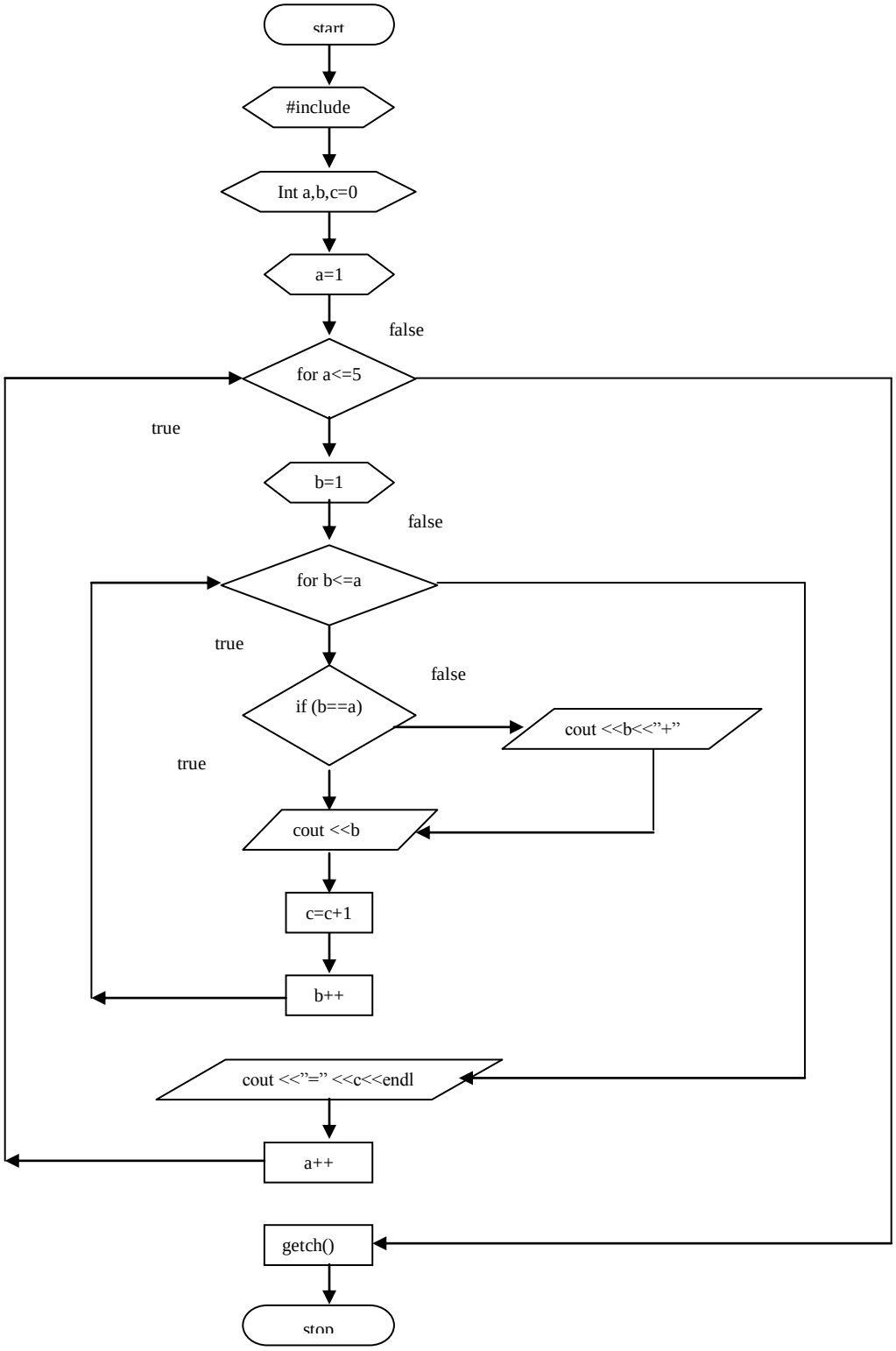
2.4.Aplikasi Program Looping Nested-For

Pembahasan aplikasi program *looping* untuk mencetak angka secara berulang dengan Borland C++, berikut adalah *output* program dan *coding* programnya:

1=1
1+2=3
1+2+3=6
1+2+3+4=10
1+2+3+4+5=15

```
#include <iostream.h>
#include <conio.h>
#include <stdio.h>
main()
{
  int a,b,c=0;
  for (a=1; a<=5; a++)
  {
    for (b=1; b<=a; b++)
    {
      if (b==a)
        cout <<b ;
      else
        cout<<b<<"+";
      c=c+1;
    }
    cout<<"="<<c<<endl;
  }
  getche();
}
```

a.Flowchart Aplikasi Program Looping Nested-For



Gambar 8.Flowchart Aplikasi Program Looping Nested-For

b.Tabel Analisa Aplikasi Program Looping Nested-For

Tabel 4. **Analisa Aplikasi Program Looping Nested-For**

a=1	**for a<= 5**	**HB**	**b=1**	**for b<= a**	**HB**	**if (b== a)**	**HB**	**cout< <b**	**cou t<< b<< "+"**	**c=c +1**	**b++**	**cout< <"="< <c<<e ndl**	**a++**	**getch()**
1	1<= 5	T	**1**	1<= 1	T	1==1	T	1	-	1=0 +1	2=1 +1			
				2<= 1	F	-	-	-	-	-	-	**=1**	2=1 +1	
	2<= 5	T	**1**	1<= 2	T	1==2	F	-	1+	2=1 +1	2=1 +1			
				2<= 2	T	2==2	T	2	-	3=2 +1	3=2 +1			
				3<= 2	F	-	-	-	-	-	-	**=3**	3=2 +1	
	3<= 5	T	**1**	1<= 3	T	1==3	F	-	1+	4=3 +1	2=1 +1			
				2<= 3	T	2==3	F	-	2+	5=4 +1	3=2 +1			
				3<= 3	T	3==3	T	3	-	6=5 +1	4=3 +1			
				4<= 3	F	-	-	-	1+	-	-	**=6**	4=3 +1	
	4<= 5	T	**1**	1<= 4	T	1==4	F	-	2+	7=6 +1	2=1 +1			
				2<= 4	T	2==4	F	-	3+	8=7 +1	3=2 +1			
				3<= 4	T	3==4	F	-	-	9=8 +1	4=3 +1			
				4<= 4	T	4==4	T	4	-	10= 9+1	5=4 +1			
				5<= 4	F	-	-	-	-	-	-	**=10**	5=4 +1	
	5<= 5	T	**1**	1<= 5	T	1==5	F	-	1+	11= 10+ 1	2=1 +1			
				2<= 5	T	2==5	F	-	2+	12= 11+ 1	3=2 +1			
				3<= 5	T	3==5	F	-	3+	13= 12+ 1	4=3 +1			
				4<= 5	T	4==5	F	-	4+	14= 13+ 1	5=4 +1			
				5<= 5	T	5==5	T	5	-	15= 14+ 1	6=5 +1			
				6<= 5	F	-	-	-	-	-	-	**=15**	6=5 +1	
	6<= 5	F	-	-	-	-	-	-	-	-	-	-	-	Proses

Keterangan : HB = Hasil Boolean
T = True
F = False

3. PENUTUP

3.1. Kesimpulan

Kesimpulan yang bisa diambil dari pembahasan ini adalah sebagai berikut:

1. Ada 3 (tiga) jenis logika looping dalam bahasa pemrograman Borland C++, yaitu: while, do-while dan for. Semua logika *looping* tersebut dapat digunakan untuk membuat aplikasi program yang berulang. Algoritma *looping* tersebut bisa bersarang, dalam arti dalam *looping* ada *looping*, seperti dalam pembahasan dicontohkan dengan *Nested For*, dalam algoritma for ada for. Untuk algoritma while atau do-while bisa saja terjadi bersarang pula, bahkan bisa saja kombinasi dari ketiga algoritma tersebut untuk mengatasi suatu kasus komputasi.

2. Langkah prosedur penyelesaian permasalahan dapat diekspresikan dengan serangkaian simbol grafis yang baku dan mudah digunakan, yaitu flowchart. Penggunaan flowchart akan menghidarkan sejak dini timbulnya kesalahan interpretasi bagi orang lain yang merupakan awal kegagalan suatu prosedur yang dikembangkan, juga untuk membantu mengetahui awal dan akhir suatu pengulangan.

3. Pembuatan flowchart dan tabel analisa ini, tidak bisa dipastikan jumlah simbol pada flowchart dan jumlah kolom pada tabel yang digunakan. Untuk masing-masing aplikasi program berbeda, ini tergantung dari jumlah algoritma dan jumlah variabel yang digunakan. Flowchart dan tabel analisa menyesuaikan dengan keadaan tersebut. Semakin banyak algoritma yang digunakan semakin panjang flowchart, begitu pula semakin banyak variabel juga mengakibatkan semakin banyak jumlah kolom pada tabel analisa yang digunakan. Hal ini tidak bisa dihindari, dan ini merupakan kelemahan dari rancangan tabel analisa yang penulis rancang ini. Studi lanjut pada penelitian ini, penulis dapat menemukan rancangan tabel analisa lebih sederhana dan ketentuan-ketentuan dalam membuat tabel analisa, yang selanjutnya sebagai standar baku dari tabel analisa ini. Pada studi awal ini, ketentuan membuat tabel secara tegas bahkan detail dari penulis belum ada, untuk sementara ini mengikuti contoh saja, sesuai yang penulis contohkan.

4. Flowchart dan tabel analisa adalah dua metode yang digunakan untuk menganalisa *output* dari program aplikasi, keduanya saling berkaitan. Pertama membuatkan flowchart dari aplikasi program tersebut untuk menganalisa algoritma dan berikutnya membuatkan tabel analisa untuk menganalisa variabel dengan mencatat *history* dari masing-masing variabel yang digunakan.

5. Intruksi *coding* yang dianalisa adalah hanya intruksi *input-output* saja, dengan mengesampingkan format tampilan. Pertimbangan penulis adalah hanya fokus pada logika program bukan sintak penulisan atau format penulisan. Untuk diketahui bahwa ada dua kesalahan dalam membuat program, yaitu kesalahan logika dan kesalahan sintak penulisan. Jadi lebih fokus pada mendiagnosa kesalahan logika, manfaat lainnya untuk menyederhanakan jumlah kolom.

3.2. Saran

Bila terjadi perbedaan hasil antara hasil komputasi dengan hasil tabel analisa, yang benar adalah hasil komputasi, karena komputer sudah sangat teruji kebenarannya dan kehandalannya. Arti kata lain, berarti hasil tabel analisa yang dibuat masih ada kesalahan, disarankan ulangi kembali untuk menganalisanya, periksa kembali tabel analisa sampai hasil tabel analisa sama dengan hasil komputasi. Sudah hal yang wajar dalam menganalisa ada kesalahan,

perlu ketelitian tinggi, dan perlu menganalisa ulang, karena manusia memeliki kelemahan kurang teliti dibanding komputer dan cepat lelah. Kedua hal tersebut teliti dan tanpa lelah adalah keunggulan dari komouter. Untuk hasil yang baik, pada prakteknya penulis menyarankan mahasiswa untuk menggunakan pinsil dalam membuat flowchart dan tabel analisa ini. Untuk hasil tabel analisa yang penulis sajikan dengan dua flowchart dan tabel analisa tersebut sudah teruji, Insya Allah benar hasilnya atau sama dengan hasil komputasi, karena sudah diuji coba selama penulis mengajarkan matakuliah Analisa dan Perancangan Algoritma.

DAFTAR PUSTAKA

Kadir, Abdul, Ir. 2001. Perograman C++ Membahas Pemrograman Berorientasi Objek Menggunakan Turbo C++ dan Borland C++. Penerbit Andi. Yogyakarta. Cetakan Ketiga.

Limanto, Susana dan Mulyono, Anton. 2002. Algoritma dan Pemrograman. Dinastindo. Jakarta.

Munir, Rinaldi. 2001. Algoritma & Pemrograman Dalam Bahasa Pascal Dan C. Penerbit Informatika Bandung. Cetakan ke lima. Jilid 1.

Supardi, Yuniar, Ir. 2000. Pascal Dan Flowchart Lewat Praktek. Dinastindo. Jakarta.

Sutanta, Edhy. 2004. Algoritma Teknik Penyelesaian Permasalahan Untuk Komputasi. Graha Ilmu. Yogyakarta.

Riwayat Hidup

Nama penulis adalah Rini Nuraini, ST, M.Kom. Pendidikan Strata Satu (S1) dengan konsentrasi pada jurusan Teknik Komputer dan pendidikan Strata Dua (S2) dengan konsentrasi pada Ilmu Komputer. Menjadi Staff Pengajar (Dosen) sejak tahun 1999 hingga saat ini, di beberapa lembaga pendidikan perguruan tinggi swasta di Jakarta dan Karawang, pada program studi Teknologi Informatika, Sistem Informasi, Manajemen Informatika, Teknik Komputer, dan Akuntansi Komputer untuk matakuliah yang berkaitan dengan komputasi, baik *software* ataupun *hardware*.