

FORTRAN DASAR

FORTRAN 77

17 Feb 2001

Luknanto@tsipil.ugm.ac.id

1

Isi bab 3. Fortran Dasar

1. Jenis data + algoritma = program
2. Format program
3. Konstanta dan Variabel
4. Operasi aritmetika dan Fungsi
5. Pernyataan "assignment"
6. Input/Output
7. Komposisi program
8. Contoh: Keasaman dari Campuran
9. Metoda Numeris: kesalahan aritmetika

17 Feb 2001

Luknanto@tsipil.ugm.ac.id

2

3.1 Jenis data + algoritma = program

- dari penjelasan pada bab sebelumnya, tampak bahwa dua langkah pertama dari suatu penyelesaian masalah adalah:
 - jenis data, dan
 - algoritma
- setiap permasalahan di lapangan selalu menyangkut pemrosesan data dari berbagai macam jenis.

17 Feb 2001

Luknanto@tsipil.ugm.ac.id

3

3.1 ... Jenis data FORTRAN

- FORTRAN mengenal 6 jenis data:
 1. integer, untuk memproses bil. utuh
 2. real atau single precision, untuk memproses bil. pecah
 3. double precision, untuk memproses bil. pecah
 4. complex, untuk memproses bil. pecah/komplek
 5. character, untuk memproses string atau karakter
 6. logical, untuk memproses TRUE dan FALSE

17 Feb 2001

Luknanto@tsipil.ugm.ac.id

4

3.1 ... algoritma

- Bagian terpenting kedua setelah data adalah algoritma, yang dinyatakan dalam bahasa FORTRAN.
- Bagian-bagian dari sebuah Program FORTRAN
 - heading
 - bagian spesifikasi/definisi
 - bagian eksekusi
 - penutup

17 Feb 2001

Luknanto@tsipil.ugm.ac.id

5

3.1 ... bagian-bagian program

- Heading: biasanya berupa nama program, kemudian diikuti dengan dokumentasi dari program.
- Spesifikasi/definisi: mendefinisikan setiap variabel yang akan digunakan dalam program.
- Eksekusi: bagian program yang melakukan hitungan dari algoritma.
- Penutup: memberi batas secara fisik bahwa program telah selesai.

17 Feb 2001

Luknanto@tsipil.ugm.ac.id

6

3.1 contoh bagian-bagian program

```

C-----
C PROGRAM Peluruhan
C-----
C Program untuk menghitung massa zat radioaktif
C
C Input : MassaAwal, WaktuParuh, WaktuSelang
C Output: MassaAkhir
C-----
REAL MassaAwal, WaktuParuh, WaktuSelang, MassaAkhir
-----
PRINT *, 'Masukkan massa, waktu paruh, dt = '
READ *, MassaAwal, WaktuParuh, WaktuSelang
MassaAkhir = MassaAwal*0.5**(WaktuSelang/WaktuParuh)
PRINT*, 'Massa akhir = ', MassaAkhir
-----
STOP 'Selesai bozz'
END
    
```

heading

definisi

eksekusi

penutup

17 Feb 2001 Luknanto@tsipil.ugm.ac.id 7

3.2 Format Program

- Setiap posisi di setiap baris dalam program FORTRAN disebut kolom yang diberi nomor dari 1,2,3,...,80.
- Setiap pernyataan FORTRAN ada pada kolom 7 s/d 72.
- Karakter pada Kolom 73 dan seterusnya diabaikan oleh compiler.
- Jika sebuah pernyataan FORTRAN membutuhkan label, maka label (harus integer 1 s/d 99999) tersebut diletakkan pada kolom 1 s/d 5.
- Setiap pernyataan FORTRAN dapat diteruskan ke baris sambungan dengan diberi tanda pada kolom 6 dengan karakter non-zero atau non-space.

17 Feb 2001 Luknanto@tsipil.ugm.ac.id 8

3.2 contoh

```

C2345678911234567892123456789312345678941234567895123456789612345678971234567898
C-----
C PROGRAM Peluruhan
C-----
C Program untuk menghitung massa zat radioaktif
C
C Input : MassaAwal, WaktuParuh, WaktuSelang
C Output: MassaAkhir
C-----
REAL MassaAwal, WaktuParuh, WaktuSelang, MassaAkhir
-----
200 PRINT *, 'Masukkan massa, waktu paruh, dt = '
READ *, MassaAwal, WaktuParuh, WaktuSelang
MassaAkhir = MassaAwal*0.5**(WaktuSelang/
WaktuParuh)
PRINT*, 'Massa akhir = ', MassaAkhir
-----
STOP 'Selesai bozz'
END
    
```

diabaikan

pernyataan: 7-72

label: 1-5

baris sambungan: 6

17 Feb 2001 Luknanto@tsipil.ugm.ac.id 9

3.3 Konstanta dan Variabel

- Konstanta adalah suatu nilai yang tidak berubah selama eksekusi.
- Konstanta integer: bilangan utuh, sekelompok angka yang tidak mengandung tanda desimal atau koma; bilangan negatif harus dimulai dengan tanda negatif, tetapi tanda positif adalah optional, misal:
 - 0
 - 137
 - 2516
 - +17745
- tidak diperbolehkan:
 - 5.280 (tidak boleh, karena ada koma)
 - 16.0 (tidak boleh, karena ada tanda desimal)
 - 5 (tidak boleh, hanya satu tanda aljabar diperkenankan)
 - 7- (tidak boleh, tanda aljabar harus mendahului bilangan)

17 Feb 2001 Luknanto@tsipil.ugm.ac.id 10

3.3 ... real

- Jenis data real dikenal juga sebagai single precision.
- Konstanta real merupakan kelompok angka yang mengandung tanda desimal, namun koma tidak diijinkan, misal:
 - 1.234
 - .01546
 - +56473.
- tidak boleh:
 - 12,345 (tidak boleh, karena ada koma)
 - 63 (konstanta real harus menggunakan tanda desimal)

17 Feb 2001 Luknanto@tsipil.ugm.ac.id 11

3.3 ... notasi scientific

- Konstanta real dapat direpresentasikan dengan notasi scientific yang terdiri dari bilangan integer atau real diikuti dengan eksponen yang ditulis sebagai E dan bilangan integer dibelakangnya.
- Konstanta real 337.456 dapat ditulis sebagai 3.37456E2 yang artinya 3.37456×10^2 , atau dapat ditulis sbg 337456E-3, 33745.6E-2, 0.337456E3, 337.456E0

17 Feb 2001 Luknanto@tsipil.ugm.ac.id 12

3.3 ... double precision

- Untuk hitungan yang membutuhkan ketelitian tinggi, bilangan real tidak memadai, maka FORTRAN menyediakan konstanta double precision, yang panjang byte-nya biasanya 2x byte single precision.
- Notasi double precision sama dengan notasi ilmiah, hanya E diganti dengan D, misal: 3.141592652589D0, 1D-3, 0.234567D+05

17 Feb 2001

Luknanto@tsipil.ugm.ac.id

13

3.3 ... karakter

- Konstanta karakter dinamai string, yaitu urutan simbol yang diperoleh dari set karakter FORTRAN.
- Standar ANSI karakter FORTRAN dapat dilihat pada tabel berikut.
- Banyak compiler FORTRAN yang mengijinkan huruf kecil dan karakter khusus dalam konstanta ini.

17 Feb 2001

Luknanto@tsipil.ugm.ac.id

14

3.3 ... karakter standar ANSI

Karakter	Keterangan	Karakter	Keterangan
blank	blank atau spasi	*	bintang, kali
0, ..., 9	angka	+	tanda plus
A, ..., Z	huruf besar	-	tanda minus
a, ..., z	huruf kecil	/	garis miring
\$	tanda dollar	,	koma
'	kuote, tanda petik	.	titik
(kurung buka	:	titik dua
)	kurung tutup	=	sama dengan

17 Feb 2001

Luknanto@tsipil.ugm.ac.id

15

3.3 ... karakter ...

- Konstanta karakter harus berada didalam pasangan kuot tunggal:
'PDQ123-A' 'John Q. Doe'
- Jika kuot tunggal akan digunakan dalam konstanta, maka harus ditulis sebagai 2 buah kuot tunggal:
'Jum"at'
berarti konstanta kuot yang berisi 6 karakter yaitu J, u, m, ', a, t.

17 Feb 2001

Luknanto@tsipil.ugm.ac.id

16

3.3 Pengenal

- Pengenal adalah nama yang digunakan untuk program, konstanta, variabel, subprogram.
- Dalam FORTRAN, pengenal harus dimulai dengan huruf, kemudian dapat diikuti dengan huruf maupun angka, misalkan: Mass, Waktu1, Grav, Kecep2
- Tidak valid:
 - R2-D2 (tidak boleh, karena ada tanda minusnya)
 - 6meter (tidak boleh, karena dimulai dengan angka)
 - Luke Skywalker (tidak boleh, karena ada spasinya)
- Selalu gunakan pengenal yang mempunyai arti sedekat mungkin dengan yang direpresentasikan.

17 Feb 2001

Luknanto@tsipil.ugm.ac.id

17

3.3 Variabel ...

- Variabel adalah nama simbolis yang digunakan dalam suatu persamaan, misalkan
Luas = Panjang x Lebar
- Variabel termasuk pengenal, oleh karena itu harus mengikuti aturan yang digunakan pada pengenal.
- Compiler selalu memberikan alamat memori dimana suatu variabel dialokasikan dan nilai daripada variabel tersebut dimasukkan kedalam memori dengan alamat di atas.

17 Feb 2001

Luknanto@tsipil.ugm.ac.id

18

3.3 ... variabel ...

- Dalam FORTRAN, variabel harus merupakan salah satu dari 6 jenis data yang telah dijelaskan di atas.
- Sebelum suatu variabel digunakan, harus didefinisikan di bagian depan program.
- Hal ini dikerjakan dengan "pernyataan jenis" dalam bentuk yang akan dijelaskan berikut ini.

17 Feb 2001

Luknanto@tsipil.ugm.ac.id

19

3.3 Pernyataan jenis data ...

- Bentuk:
 - spesifikasi-jenis list
 - dengan spesifikasi-jenis adalah salah satu dari:
 - INTEGER
 - REAL
 - DOUBLE PRECISION
 - COMPLEX
 - CHARACTER**n* (*n* adalah konstanta integer)
 - LOGICAL
 - dan list adalah daftar pengenalan yang dipisahkan oleh koma.
- Guna:
 - mendeklarasikan bahwa identifier yang ada dalam list mempunyai jenis yang disebut dalam spesifikasi-jenis.
- Contoh:
 - CHARACTER*50 Nama, Alamat
 - DOUBLE PRECISION Luas, Tekanan, Modulus

17 Feb 2001

Luknanto@tsipil.ugm.ac.id

20

3.3 ... jenis data implicit ...

- Jika suatu variabel tidak dideklarasikan jenis datanya, maka FORTRAN menganggap bahwa:
 - Setiap variabel yang dimulai dengan huruf I, J, K, L, M, N mempunyai jenis data INTEGER.
 - Setiap variabel yang tidak dimulai dengan 6 huruf di atas mempunyai jenis REAL.
- Jika diperlukan, maka peraturan di atas dapat dihilangkan dengan pernyataan IMPLICIT.

17 Feb 2001

Luknanto@tsipil.ugm.ac.id

21

3.3 Konstanta bernama

- Seringkali kita menggunakan suatu nilai konstanta dengan menyebut namanya, misalkan Gravitasi, Pi, e, dlsb.
- FORTRAN mengakomodasi ini dengan menggunakan pernyataan PARAMETER.
- Cara menggunakannya dijelaskan berikut ini.

17 Feb 2001

Luknanto@tsipil.ugm.ac.id

22

3.3 pernyataan konstanta bernama

- Bentuk:
 - PARAMETER (param₁ = const₁, param₂ = const₂, ..., param_n = const_n)
 - dengan
 - param₁, param₂, ..., param_n adalah identifier, dan
 - const₁, const₂, ..., const_n adalah konstanta atau ekspresi matematis yang menggunakan sembarang konstanta dan konstanta yang telah didefinisikan terlebih dahulu.
- Guna:
 - memberi nama konstanta const_i dengan nama param_i, agar mudah dikenali.
- Contoh:
 - PARAMETER (Gravitasi = 9.78, PI = 3.14, Grav2= 2.0*Gravitasi)

17 Feb 2001

Luknanto@tsipil.ugm.ac.id

23

3.3 Inisialisasi variabel

- Didalam FORTRAN semua variabel pada awalnya tidak diinisialisasi, artinya pada awal program semua nilai variabel tidak didefinisikan atau nilainya tidak diketahui.
- Oleh karena itu, suatu kebiasaan yang baik untuk menginisialisasi variabel, agar tidak terjadi hal yang tak terduga.
- Inisialisasi dapat dilakukan saat compiling dengan menggunakan perintah berikut ini.

17 Feb 2001

Luknanto@tsipil.ugm.ac.id

24

3.3 pernyataan inisialisasi variabel

- Bentuk:
DATA (list₁/data₁/, list₂/data₂/, ..., list_n/data_n/)
dengan
list, adalah list nama variabel dipisahkan oleh koma, dan data, adalah list konstanta yang digunakan untuk inisialisasi variabel-variabel yang ada dalam list.
- Guna:
inisialisasi setiap variabel dalam list, dengan nilai konstanta yang ada dalam data, pada waktu compile. Biasanya pernyataan DATA muncul setelah pernyataan PARAMETER.
- Contoh:
DATA W /2.0/, X, Y /1.0, 4.5/, Jedi/Luke Skywalker/
DATA Z1, Z2, Z3 /3 * 2.0/

3.4 Operasi aritmetika

- Dalam FORTRAN, tambah dan kurang dinyatakan dengan tanda (+) dan (-). Kali dan bagi dinyatakan dengan tanda (*) dan (/).
- Setiap perkalian harus digunakan tanda (*); jadi N kali 2 tidak boleh ditulis sebagai N2 atau 2N tetapi harus ditulis sebagai 2*N atau N*2.
- Pangkat dinyatakan dengan tanda (**) bukan (^); jadi b^2-4ac didalam FORTRAN harus ditulis sebagai: $b**2 - 4*a*c$

3.4 ... operasi aritmatika ...

Operator	Operasi
**	pangkat
*	kali
/	bagi
+	tambah, positif
-	kurang, negatif

3.4 ... operasi aritmetika ...

- Jika dua konstanta dari jenis yang sama dioperasikan menggunakan (**, *, /, +, -), hasilnya adalah dari jenis yang sama.
- Jadi 9.0/4.0 hasilnya adalah 2.25, namun 9/4 hasilnya 2
- Serupa diatas, andaikan N = 2 dan X = 2.0, maka 1.0/N hasilnya 0.5, sedangkan 1/N hasilnya 0

3.4 operasi campur jenis

- Kadang tidak terhindarkan adanya operasi dari gabungan berbagai jenis data, maka perlu dijelaskan secara rinci hasil operasi aritmetika dari campuran pelbagai jenis data.
- Hasil operasi campur jenis ini disajikan dalam tabel berikut ini.

3.4 tabel aritmetika campur jenis

Operasi	Evaluasi	Hasil
integer op real	Integer dikonversi ke real, kemudian operasi dilakukan	real
integer op double precision	Integer dikonversi ke double precision, kemudian operasi dilakukan	double precision
real op double precision	Real dikonversi ke double precision, kemudian operasi dilakukan	double precision

3.4 contoh

- Perhatikan konversi tidak terjadi sampai dibutuhkan:
 - $1.0/4 \rightarrow 1.0/4.0 \rightarrow 0.25$
 - $3.0 + 8/5 \rightarrow 3.0 + 1 \rightarrow 3.0 + 1.0 \rightarrow 4.0$
 - $3 + 8.0/5 \rightarrow 3 + 8.0/5.0 \rightarrow 3 + 1.6 \rightarrow 3.0 + 1.6 \rightarrow 4.6$
 - $0.1 + 0.1D0 \rightarrow 0.999999940395355D-1 + 0.1D0 \rightarrow 0.199999994039535D0$
- Oleh karena itu menggunakan operasi campur seperti di atas dianggap sebagai kebiasaan jelek.

... sampai di sini ...

3.4 ... pangkat campur jenis ..

- Satu-satunya operasi yang diharuskan menggunakan campur jenis adalah pangkat yaitu real atau double precision dipangkatkan integer.
 - $2.0 ** 3 \rightarrow 2.0 * 2.0 * 2.0 \rightarrow 8.0$
 - $(-4.0) ** 2 \rightarrow (-4.0) * (-4.0) \rightarrow 16.0$
- Bandingkan jika pangkatnya berupa bilangan real
 - $2.0 ** 3.0$ akan dievaluasi sebagai $e^{3.0 * \ln(2.0)}$ hasilnya tidak akan tepat 8.0, karena error pembulatan.

3.4 ... pangkat campur jenis ...

- Konsekuensi logis dari cara menghitung real pangkat real tersebut, maka bilangan real negatif dipangkatkan bilangan real akan menghasilkan error, karena ln (bil. negatif) tidak terdefiniskan.
- Oleh karena itu jangan menggunakan pangkat real sebagai ganti pangkat bilangan integer.
 - Contoh: $b^2 - 4ac \rightarrow b ** 2 - 4.0*a*c$
- Sebaliknya pangkat real harus digunakan pada kasus yang tepat, misalkan untuk menghitung: $H^{3/2} \rightarrow H ** 1.5$

3.4 Prioritas aritmetika

Operasi aritmetika dilakukan dengan aturan sbb:

- Semua perpangkatan dilakukan pertama; dikerjakan urut dari kanan ke kiri.
- Semua perkalian dan pembagian dilakukan kemudian; dikerjakan urut dari kiri ke kanan.
- Semua penambahan dan pengurangan dilakukan terakhir; dikerjakan urut dari kiri ke kanan.

3.4 contoh prioritas

- Contoh:
 - $2 ** 3 ** 2 = 2 ** 9 = 512$
 - $10 - 8 - 2 = 2 - 2 = 0$
 - $10 / 5 * 2 = 2 * 2 = 4$
 - $2 + 4 / 2 = 2 + 2 = 4$
 - $2 + 4 ** 2 / 2 = 2 + 16 / 2 = 2 + 8 = 10$

3.4 modifikasi prioritas

- Prioritas ini dapat dimodifikasi menggunakan kurung.
Contoh:
 $(5 * (11 - 5) ** 2) * 4 + 9$
- Ekspresi $(11 - 5)$ dievaluasi pertama, menghasilkan
 $(5 * 6 ** 2) * 4 + 9$
- Kemudian $(5 * 6 ** 2)$ dievaluasi berdasarkan prioritas standar, menghasilkan
 $(5 * 36) * 4 + 9 \rightarrow 180 * 4 + 9 \rightarrow 720 + 9 \rightarrow 729$

17 Feb 2001

Luknanto@tsipil.ugm.ac.id

37

3.4 Fungsi

- Karena banyak sekali hitungan yang menggunakan akar kuadrat dari sebuah bilangan real, maka FORTRAN menyediakan fungsi untuk menghitung hal ini dengan cara:
SQRT(argumen)
dengan argumen adalah konstanta, variabel, ekspresi jenis real.
- Untuk menghitung akar dari 7, dalam FORTRAN harus ditulis
SQRT(7.0) bukan SQRT(7)
- Jika dibutuhkan argumen dapat dikonversikan terlebih dahulu dengan fungsi REAL(), contoh:
SQRT(REAL(NUM))
- Beberapa *fungsi intrinsic* yang disediakan FORTRAN disajikan berikut ini.

17 Feb 2001

Luknanto@tsipil.ugm.ac.id

38

3.4 sebagian fungsi intrinsic

Nama	Deskripsi	Jenis argumen	Jenis hasil
ABS (x)	Nilai absolut dari x	I, R, DP	sama
COS (x)	Cosinus dari x dalam radian	R, DP	sama
DBLE (x)	Konversi dari x ke double precision	I, R	DP
DPROD (x,y)	Perkalian double precision dari x * y	R	DP
EXP (x)	Evaluasi e ^x	R, DP	sama
INT (x)	Bagian integer dari x	R, DP	I
LOG (x)	Evaluasi ln (x)	R, DP	sama
MAX (x ₁ ,...,x _n)	Maksimum dari x ₁ ,...,x _n	I, R, DP	sama
MIN (x ₁ ,...,x _n)	Minimum dari x ₁ ,...,x _n	I, R, DP	sama
MOD (x,y)	Sisa dari x dibagi y	I, R, DP	sama
NINT (x)	Pembulatan x ke bilangan integer terdekat	R, DP	I
REAL (x)	Konversi dari x ke real	I, DP	R
SIN (x)	Sinus dari x dalam radian	R, DP	sama
SQRT (x)	Akar kuadrat dari x	R, DP	sama

17 Feb 2001

Luknanto@tsipil.ugm.ac.id

39

3.5 Pernyataan assignment

- Bentuk:
variabel = ekspresi
dengan
variabel adalah sebuah pengenal FORTRAN yang sah
ekspresi dapat berupa sebuah konstanta, atau variabel yang lain yang telah mempunyai nilai sebelumnya, atau sebuah rumus yang harus dihitung.
- Guna:
memberikan nilai evaluasi ekspresi kepada variabel.
- Contoh:
XKoord = 2.35
YKoord = SQRT(25.0)
XKoord = YKoord + 2.1*XKoord

17 Feb 2001

Luknanto@tsipil.ugm.ac.id

40

3.5 Assignment campur jenis

Jenis ekspresi	Jenis variabel	Hasil
real atau double precision	integer	nilai dari ekspresi dipotong menjadi bagian integernya dan diberikan kepada variabel
integer	real atau double precision	nilai dari ekspresi dikonversi menjadi bagian sebuah nilai real atau double precision dan diberikan kepada variabel
double precision	real	nilai dari ekspresi dipotong menjadi single precision dan diberikan kepada variabel
real	double precision	nilai dari ekspresi dikembangkan menjadi bagian sebuah nilai double precision dan diberikan kepada variabel

17 Feb 2001

Luknanto@tsipil.ugm.ac.id

41

3.5 ...assignment ...

- Pada setiap assignment, variabel yang akan diberi nilai harus selalu diletakkan di sisi kiri dari '='
- Contoh:
REAL X
INTEGER M, N

Pernyataan	Error
5 = N	Nama variabel harus di sebelah kiri '='
X + 3.5 = 4.26	Ekspresi numeris tidak boleh berada di sebelah kiri '='
N = 'Five'	Konstanta karakter tidak boleh diberikan kpd variabel numeris
N = '2' + '3'	'2' + '3' bukan ekspresi sah
M = N = 1	N = 1 bukan ekspresi sah

17 Feb 2001

Luknanto@tsipil.ugm.ac.id

42

3.5 bukan persamaan aljabar

- Pemrogram pemula kadang mencampurkan kedua assignment di bawah ini:

A = B
B = A

A	8.53
B	3.40

 $\rightarrow A = B \rightarrow$

A	3.40
B	3.40

A	8.53
B	3.40

 $\rightarrow B = A \rightarrow$

A	8.53
B	8.53

17 Feb 2001

Luknanto@tsipil.ugm.ac.id

43

3.5 tukar guling

- Pemrogram pemula bingung dalam operasi saling tukar nilai variabel A dan B:

REAL A, B, Z
Z = A
A = B
B = Z

A	8.53
B	3.40
Z	?

 $\rightarrow Z = A \rightarrow$

A	8.53
B	3.40
Z	8.53

 $\rightarrow A = B \rightarrow$

A	3.40
B	3.40
Z	8.53

 $\rightarrow B = Z \rightarrow$

A	3.40
B	8.53
Z	8.53

17 Feb 2001

Luknanto@tsipil.ugm.ac.id

44

3.5 jumlah berantai

- Pemrogram pemula bingung dalam menjumlah:

A = A + X

- Nilai A yang lama selalu diganti dengan nilai A yang baru

A	10.0
X	2.0

 $\rightarrow A = A + X \rightarrow$

A	12.0
X	2.0

 $\rightarrow A = A + X \rightarrow$

A	14.0
X	2.0

17 Feb 2001

Luknanto@tsipil.ugm.ac.id

45

3.6 Output

- Bentuk:

PRINT *, output-list

dengan

output-list dapat berupa sebuah ekspresi, atau list dari ekspresi yang dibatasi oleh koma. Setiap ekspresi dapat berupa suatu konstanta, variabel, atau rumus.

Dapat pula digunakan seperti:

PRINT *

Guna:

menyajikan nilai dari output-list, setiap perintah PRINT selalu menghasilkan satu baris baru. Jika tidak ada output-list, maka baris kosong disajikan.

- Contoh:

PRINT *, 'Koordinat: X = ', XKoord, ' dan Y =', YKoord
PRINT *

17 Feb 2001

Luknanto@tsipil.ugm.ac.id

46

3.6 Input

- Bentuk:

READ *, input-list

dengan

input-list dapat berupa sebuah variabel, atau list dari variabel yang dibatasi oleh koma.

Guna:

mentransfer nilai konstanta dari sumber luar (biasanya dari keyboard atau file) kedalam variabel yang terdapat dalam input-list.

Beberapa aturan yang harus diperhatikan:

- Data baris baru akan diproses setiap pernyataan READ di eksekusi.
- Jika terdapat **lebih sedikit** masukan pada satu baris data input dibandingkan variabel yang ada dalam input-list, baris-baris selanjutnya akan dibaca sampai seluruh variabel terisi.
- Jika terdapat **lebih banyak** masukan pada satu baris data input dibandingkan variabel yang ada dalam input-list, maka nilai-nilai sisanya diabaikan.

17 Feb 2001

Luknanto@tsipil.ugm.ac.id

47

3.6 Aturan Input

- Beberapa aturan yang harus diperhatikan:

- Masukan dalam setiap baris harus merupakan konstanta yang sesuai jenisnya dengan variabel terkait yang ada dalam input-list. (Namun demikian konstanta integer dapat diberikan kepada variabel real atau double precision, dan konstanta real dapat diberikan kepada variabel double precision dengan konversi jenis otomatis terjadi.)
- Masukan dalam data input harus dibatasi dengan koma atau satu atau lebih spasi.

- Contoh:

READ *, Tinggi, Kecep, Waktu

- Data yang disediakan dapat berupa:

- 100.0, 2.5, 6000
- 100.0 2.5 6000
- 100.0, 2.5, 6000

17 Feb 2001

Luknanto@tsipil.ugm.ac.id

48

3.7 Komposisi Program

- Komposisi program telah dijelaskan di depan.
- Heading program mempunyai bentuk:
PROGRAM nama
dengan nama adalah pengenalan sah FORTTRAN.
- Walaupun heading program tidak diharuskan, ini sebaiknya digunakan untuk membedakan dengan unit atau sub-program yang lain.

17 Feb 2001

Luknanto@tsipil.ugm.ac.id

49

3.8 Contoh Program ...

- Sebuah pabrik memproduksi suatu cetakan logam yang harus didinginkan dalam air, kemudian dibersihkan dalam larutan asam.
- Pada saat dibersihkan dalam larutan asam, sebagian air dari proses sebelumnya ikut cetakan logam dan kemudian terlarut dalam larutan asam tersebut, sehingga terjadi pengenceran asam.
- Pada saat diambil dari larutan asam, sebagian larutan ini ikut dalam bahan ini.
- Volume cairan tetap, karena jumlah yang masuk dan yang keluar dari cetakan logam tersebut adalah sama.

17 Feb 2001

Luknanto@tsipil.ugm.ac.id

50

3.8 ... Contoh Program ...

- Kita ingin merancang suatu program yang menghitung keasaman larutan pembersih, setelah digunakan untuk proses membersihkan cetakan logam tersebut.
- Program ini harus pula bisa menentukan pada saat larutan terlalu encer yaitu dengan membandingkan suatu nilai keasaman ijin.
- Input program berupa volume larutan asam, volume air yang ikut cetakan logam masuk kedalam larutan asam, jumlah cetakan yang dibersihkan, dan nilai ijin keasaman.
- Output berupa keasaman setelah sejumlah cetakan dibersihkan dan jumlah cetakan yang masih bisa diproses sebelum keasaman turun dibawah nilai ijin.

17 Feb 2001

Luknanto@tsipil.ugm.ac.id

51

3.8 ... analisis algoritma 1 ...

- Jika jumlah asam pada awalnya adalah A dan jumlah air adalah W , maka kadar larutan asam mula-mula adalah

$$\frac{A}{A+W}$$

- Pada saat cetakan logam diambil dari larutan asam, jumlah asam yang berada dalam larutan adalah

$$\left(\frac{A}{A+W}\right) \cdot A$$

17 Feb 2001

Luknanto@tsipil.ugm.ac.id

52

3.8 ... analisis algoritma 2 ...

- Ini berarti pada saat cetakan kedua dimasukkan kedalam larutan pembersih, kadar asamnya menjadi

$$\frac{\left(\frac{A}{A+W}\right) \cdot A}{A+W} = \left(\frac{A}{A+W}\right)^2$$

- Secara umum setelah cetakan ke n dibersihkan, kadar asam dalam larutan pembersih adalah:

$$\left(\frac{A}{A+W}\right)^n$$

17 Feb 2001

Luknanto@tsipil.ugm.ac.id

53

3.8 ... analisis algoritma 3 ...

- Jika kadar asam ijin adalah L , maka syarat larutan pembersih adalah

$$\left(\frac{A}{A+W}\right)^n > L$$

- Sehingga jumlah cetakan n yang mampu dibersihkan, sebelum kadar asam turun dibawah nilai ijin adalah:

$$n > \frac{\log L}{\log A - \log(A+W)}$$

17 Feb 2001

Luknanto@tsipil.ugm.ac.id

54

3.8 Algoritma

1. Masukkan nilai: Vasam, Vair, Ncetak
2. Hitung Vtotal = Vasam + Vair
3. Hitung Kons = (Vasam / Vtotal) ** Ncetak
4. Sajikan hasil Kons
5. Masukkan nilai: Ambang
6. Hitung Ncetak = 1 + LOG(Ambang) / (LOG(Vasam) - LOG(Vtotal))
7. Sajikan hasil Ncetak

17 Feb 2001

Luknanto@tsipil.ugm.ac.id

55

3.8 FORTRAN programnya

```

C2345678911234567892123456789312345678941234567895
PROGRAM Pembersih
C-----
      INTEGER Ncetak
      REAL Vasam, Vair, Vtotal, Ambang, Kons
      PRINT*, 'Berapa vol. asam, air, jml. cetakan? ='
      READ*, Vasam, Vair, Ncetak
      Vtotal = Vair + Vasam
      Kons = (Vasam / Vtotal) ** Ncetak
      PRINT*, 'Kons. setelah ', Ncetak, ' cetakan = ', Kons
      PRINT*, 'Berapa ambang ijin? = '
      READ*, Ambang
      Ncetak = 1 + LOG(Ambang)/(LOG(Vasam) - LOG(Vtotal))
      PRINT*, 'Jumlah cetakan yang dapat diproses lagi = ',
      1      Ncetak
      END
    
```

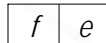
17 Feb 2001

Luknanto@tsipil.ugm.ac.id

56

3.9 Kesalahan aritmetika

- Bilangan real dalam komputer dapat dinyatakan sebagai $f \times 2^e$ dengan f adalah bagian pecahan (atau mantissa) dan eksponen e dalam binari.
- Didalam komputer dengan ukuran word tertentu, maka sebagian memori untuk menyimpan mantissa dan memori yang lain untuk menyimpan eksponen.



17 Feb 2001

Luknanto@tsipil.ugm.ac.id

57

3.9 Overflow dan underflow

- Overflow terjadi jika bagian eksponen dari suatu bilangan real terlalu besar untuk disimpan dalam memori.
- Untuk single precision: eksponen 8 bit akan membatasi kisaran bilangan diantara -10^{38} s/d $+10^{38}$.
- Underflow terjadi jika bagian eksponen dari suatu bilangan real terlalu kecil untuk disimpan dalam memori.
- Untuk single precision: eksponen 8 bit akan membatasi kisaran bilangan diantara -10^{-38} s/d $+10^{-38}$.

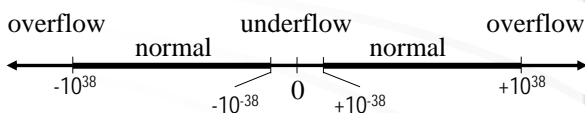
17 Feb 2001

Luknanto@tsipil.ugm.ac.id

58

3.9 ... over- dan under- flow

- Overflow dan underflow untuk bilangan single precision (eksponen 8 bit)



17 Feb 2001

Luknanto@tsipil.ugm.ac.id

59

3.9 Kesalahan pembulatan

- Tidak semua bilangan real dapat direpresentasikan secara lengkap oleh komputer karena terbatasnya bit dalam mantissa.
- Hanya sebagian dari bilangan real yang mampu disimpan secara tepat dalam memori.
- Bilangan real seperti $\pi = 3.141592653589\dots$ dan $1/3 = 0.33333333\dots$, yang tidak pernah berakhir \rightarrow tidak pernah dapat disimpan secara tepat didalam sejumlah bit yang terbatas.

17 Feb 2001

Luknanto@tsipil.ugm.ac.id

60

3.9 ... kesalahan pembulatan

- Pada kenyataannya, dari setiap bilangan real dalam bentuk $0.d$ dengan d adalah sembarang bilangan, hanya 0.0 dan 0.5 yang dapat direpresentasikan secara eksak; 0.1, 0.2, 0.3, 0.4, 0.6, 0.7, 0.8, 0.9 tidak dapat.
- Hanya 4 dua-digit bilangan real $0.d_1d_2$ yang dapat direpresentasikan secara eksak yaitu 0.00, 0.25, 0.50, 0.75; sisanya 96 buah tidak dapat.

17 Feb 2001

Luknanto@tsipil.ugm.ac.id

61

3.9 ...kesalahan pembulatan

- Secara umum, bilangan real yang dapat direpresentasikan secara eksak dalam komputer mempunyai bentuk $(m/2^k)$ dengan m dan k adalah bilangan utuh.
- Kesalahan pembulatan bertambah besar dengan adanya operasi aritmetika.
- Sebagai contoh penambahan 3 bilangan real 0.4104, 1.0, 0.2204.
- Untuk kemudahan, diandaikan hitungan dilakukan dengan ketelitian 4 angka.

17 Feb 2001

Luknanto@tsipil.ugm.ac.id

62

3.9 ...kesalahan pembulatan

- Representasi ilmiahnya adalah 0.4104×10^0 , 0.1000×10^1 , 0.2204×10^0 .
- Langkah pertama dalam penambahan dua bilangan adalah "align" tanda desimal dengan menaikkan eksponen yang lebih kecil dan menggeser mantisa.
- Jadi penjumlahan dua bilangan: $0.0410 \times 10^1 + 0.1000 \times 10^1 = 0.1410 \times 10^1$
- Penambahan dengan bilangan ketiga membutuhkan penggeseran mantisa: $0.1410 \times 10^1 + 0.0220 \times 10^1 = 0.1630 \times 10^1$

17 Feb 2001

Luknanto@tsipil.ugm.ac.id

63

3.9 ...kesalahan pembulatan

- Jika pertambahan dilakukan terhadap dua bilangan yang terkecil terlebih dahulu, maka hasilnya: $0.4104 \times 10^0 + 0.2204 \times 10^0 = 0.6308 \times 10^0$
- Penambahan dengan bilangan ketiga membutuhkan penggeseran mantisa: $0.0631 \times 10^1 + 0.1000 \times 10^1 = 0.1631 \times 10^1$
- Hanya penambahan tiga bilangan real saja, menghasilkan hasil berbeda 1.631 dan 1.630. Jika operasi semacam ini dilakukan berulang-ulang, maka perbedaan semacam ini akan membesar.

17 Feb 2001

Luknanto@tsipil.ugm.ac.id

64

3.9 ... kesalahan pembulatan

- Banyak persamaan aljabar yang gagal jika diaplikasikan terhadap bilangan real. Contoh: terdapat dapat dicari 3 bilangan real A, B, dan C dimana persamaan di bawah ini tidak benar.
- $(A + B) + C$ dan $A + (B + C)$
- $(A * B) * C$ dan $A * (B * C)$
- $A * (B + C)$ dan $(A * B) + (A * C)$

17 Feb 2001

Luknanto@tsipil.ugm.ac.id

65

3.9 ... kesalahan pembulatan

- Salah satu contoh program FORTRAN berikut ini:

```
PROGRAM Error
REAL A, B, C

READ *, A, B
C = ((A + B)**2 - 2*A*B - B**2)/A**2
PRINT *, C
END
```

17 Feb 2001

Luknanto@tsipil.ugm.ac.id

66

3.9 ... kesalahan pembulatan

- Tabel di bawah ini menunjukkan hasil dari satu sistem komputer.

A	B	C
0.5	888.0	1.00000
0.1	888.0	-12.5000
0.05	888.0	-50.0000
0.003	888.0	-13888.9
0.001	888.0	-125000.0

17 Feb 2001

Luknanto@tsipil.ugm.ac.id

67

3.9 Kesalahan pemotongan

- Banyak masalah yang penyelesaiannya hanya diperoleh dengan proses tak terhingga.
- Contoh $e = 2.71828182845\dots$, dapat dihitung dari seri tak terhingga:

$$e = 1 + \frac{1}{1!} + \frac{1}{2!} + \frac{1}{3!} + \frac{1}{4!} + \dots$$

dengan $n! = 1 \times 2 \times 3 \times \dots \times n$.

17 Feb 2001

Luknanto@tsipil.ugm.ac.id

68

3.9 ...kesalahan pemotongan

- Jadi nilai e hanya dapat didekati. Selisih antara nilai pendekatan dengan nilai e sebenarnya inilah yang disebut kesalahan pemotongan.

Jumlah suku	Nilai pendekatan
2	2.0
3	2.6666666...
4	2.7083333...
5	2.7166666...
6	2.71805555...

17 Feb 2001

Luknanto@tsipil.ugm.ac.id

69

Sampai disini BOZZ

17 Feb 2001

Luknanto@tsipil.ugm.ac.id

70