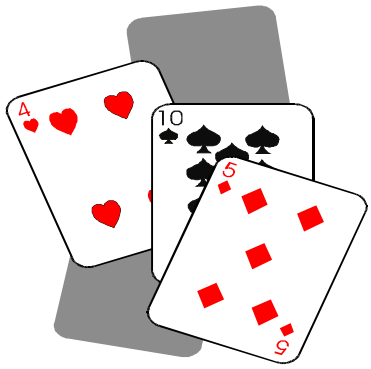
Programming in Fortran : Lecturer Notes

Avinanta T, ST



Introduction

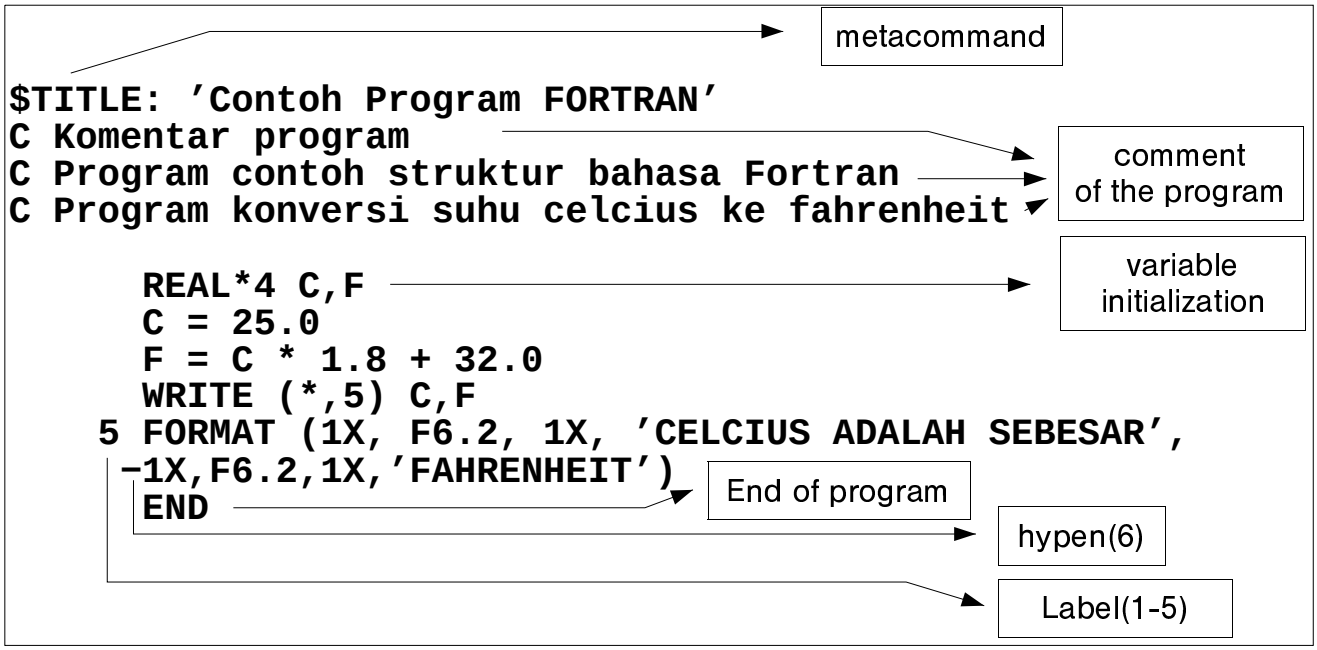
- stands for **Formula Translation**
- high level languange
- uses for numerical / mathematical computation

- History :
  - 1950. Developed by Jhon Backus − IBM
  - 1954. First reference by Programming Research Group (IBM)
  - 1957. Deployed in IBM 740 computer system
  - 1958. Fortran II was released.
  - 1962. Fortran IV was released.
  - 1966. Fortran 77 was released and standarized by ANSI (American National Standard Institute)

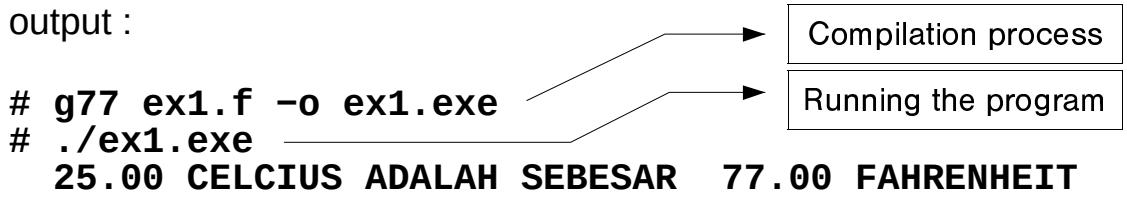
- Fortran compilers :
  - Microsoft Fortran (DOS)
  - (G77) GNU Fortran77 (UNIX − Linux)

- Further reading :
  - Jogiyanto H.M., Teori dan Aplikasi Program Komputer Bahasa Fortran, Andi Offset, Jogjakarta, 1989
  - Suryadi, M.T., Bahasa Fortran dan Analisis Numerik, Seri Diktat Kuliah, Penerbit Gunadarma, 1995
  - Husni Susanto, Belajar Fortran Lewat Basic, PT. Elex Media Komputindo, Jakarta, 1987

Fortran by Example



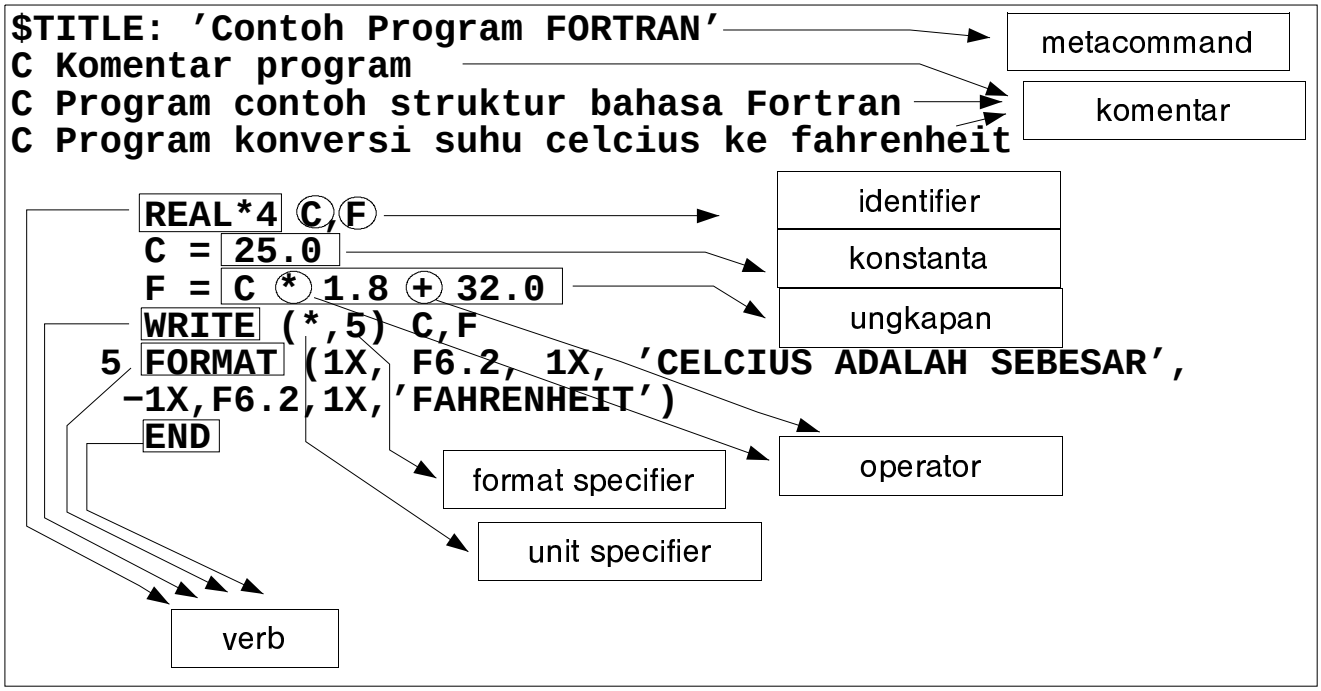
output :



Dari contoh di atas dapat dimengerti **struktur program Fortran**

- ditulis dalam fortran coding sheet 80 kolom
- dibagi menjadi 5 bagian kolom
  - **Kolom ke−1**
    Digunakan untuk komentar program (didahului dengan karakter "C" atau "c") atau metacommand (didahului dengan karakter "$").
  - **Kolom 1−5**
    Digunakan untuk penulisan label statement.
  - **Kolom 6**
    Digunakan untuk indikasi sambungan dari statement sebelumnya.
  - **Kolom 7−72**
    Digunakan untuk menulis statement program.
  - **Kolom 73−80**
    Tidak digunakan, dapat dimanfaatkan untuk komentar.

Elemen Program Fortran



- **metacommand (compiler directives)**
  sifatnya optional (tidak harus ada) dan digunakan untuk melakukan komunikasi ke compiler fortran
- **comments (komentar)**
  digunakan untuk memberi keterangan pada program sehingga memudahkan dokumentasi dan pembacaan program
- **statement**
  merupakan inti dari program yang berupa instruksi−instruksi. Statement dapat berupa :
  - *konstanta*
    nilai yang sudah pasti dan tidak akan berubah selama program berjalan
  - *operator*
    untuk melakukan operasi nilai antara variabel atau konstanta
  - *ungkapan/expression*
    pemberian nilai kepada suatu identifier
  - *identifier*
    memberi nama kepada suatu variable, array, function, atau subroutine

- *verb*
  instruksi kerja yang dimengerti oleh compiler fortran
- *unit specifier*
  menunjukkan divais (layar/printer/file) yang digunakan dalam operasi input dan output
- *format specifier*
  menunjukkan nomor label dimana terdapat statement format yang akan digunakan pada data input dan output

Data Types

Hal yang perlu diperhatikan sebelum memulai menulis statement−statement logika program adalah mendefinisikan **tipe data** semua variabel−variabel yang digunakan dalam program, walaupun sifatnya optional.

Tujuan :

- memberitahukan compiler untuk menyediakan ruang memori yang cukup
- mencegah terjadinya kesalahan pemasukan tipe data

Beberapa tipe data variabel dalam Fortran :

- *integer* (bilangan numerik − bulat)
  integer 2 byte : −32767 s/d 32767
  integer 4 byte : −2147483647 s/d 2147483647
  penamaan variabel integer diawali huruf I,J,K,L,M,N jika tidak didefinisikan terlebih dahulu
  contoh :

```
integer*2 int1,int2
integer*4 jnt,nnt
int1 = 0
int2 = 2048
jnt = 1000000
nnt = −9999999
```

- *real single precision* (pecahan ketepatan tunggal 4 byte)
  positif : 8.4E−37 s/d 3.37E+38
  negatif : −3.37E+38 s/d −8.43E−37
  penamaan variabel real tidak boleh diawali huruf I,J,K,L,M,N
  contoh :

```
real   vreal1
real*4 vreal2,vreal3
vreal1 = −1374.3
vreal2 = 12.34E+3
vreal3 = vreal1 + vreal2
```

- *real double precission* (pecahan ketepatan ganda 8 byte)
  positif : 4.19D−307 s/d 1.67D+308
  negatif : −1.67D+308 s/d −4.19D−307
  penamaan variabel real tidak boleh diawali huruf I,J,K,L,M,N
  contoh :

```
real*8 vreal1
double precision vreal2
vreal1 = 12345.490D+2
vreal2 = vreal1 / 389.29D+1
```

- *character* (karakter max 127 byte, panjang default : 4 byte)
  penamaan variabel karakter bebas
  contoh :

```
character namapendek
character*40 namapanjang
namapendek  = 'TIKA'
namapanjang = 'KARTIKA SARI SEMBRONOWATI'
```

- *logical*
  penamaan variabel bebas
  hanya bisa berisi salah satu dari 2 nilai −> benar / salah
  benar −> .TRUE.
  salah −> .FALSE.
  contoh :

```
logical   status1
logical*2 status2
logical*4 status3
status1 = .TRUE.
status2 = .FALSE.
status3 = status1 .AND. status2
```

Penamaan Variable :

- max. panjang 1320 karakter, tetapi compiler hanya mengenali 6 karakter pertama
- karakter pertama harus berupa huruf alphabet
- tidak boleh mengandung spasi / blank
- tidak boleh mengandung karakter khusus (selain huruf dan angka)
- mengikuti aturan penamaan setiap tipe data

Operator & Ungkapan

Operator merupakan suatu operasi yang dikenakan terhadap dua variable/konstanta atau lebih. Yang kemudian rangkaian operasi ini disebut dengan ungkapan.

- Operator aritmatika & ungkapan aritmatika

Operator Aritmatika	***Tujuan***	***Jenjang***
**	memangkatkan	1
*	Mengalikan	2
/	Membagi	2
+	Menambahkan	3
−	Mengurangkan	3

Contoh ungkapan Aritmatika:

```
g = a* b / c * d
h = (a * b)/c + d
i = a + (b * c) /d
j = (a**2+b**2) ** 0.5
k = ((−b+c)/(2*a)) ** (1/7)
```

- Operator hubungan & ungkapan hubungan (*relation*)

Operator hubungan	***Tujuan***
.LT.	lebih kecil dari (less than)
.LE.	lebih kecil sama dengan dari (less and equal than)
.EQ.	Sama dengan (equal)
.NE.	Tidak sama dengan (not equal)
.GT.	Lebih besar dari (greater than)
.GE.	Lebih besar sama dengan dari (greather and equal than)

Contoh ungkapan hubungan :

```
if ( A .LT. B ) then
   S = 'A lebih kecil dari B'
endif

if ( A .EQ. B ) then
   S = 'A sama dengan B'
endif

if ( B .GE. C ) then
   S = 'B lebih besar sama dengan C'
endif

if ( C .NE. D ) then
   S = 'C tidak sama dengan D'
endif
```

- Operator Logika

Operator Aritmatika	*Maksud*	*Jenjang*
.NOT.	Tidak	1
.AND.	Dan	2
.OR.	Atau	3

Contoh program :

```
C Contoh Program Operator & Ungkapan Logika
      LOGICAL S1, S2
      S1 = .TRUE.
      S2 = .FALSE.
      IF ( S1 .AND. S2 ) THEN
         WRITE (*,5)
      ELSE
         WRITE (*,10)
      ENDIF
    5 FORMAT (1X, 'LOGIKA S1 AND S2 ADALAH BENAR')
   10 FORMAT (1X, 'LOGIKA S1 AND S2 ADALAH SALAH')
      END
```

Output :

```
# g77 coba2.f -o coba2
# ./coba2
 LOGIKA S1 AND S2 ADALAH SALAH
```

- Ungkapan Karakter
  Memasukkan nilai karakter/string kepada suatu variabel/identifier.
  Contoh :

```
nama1 = 'Anastasia Yanti'
nama2 = 'Toshiro Shiba'
jenis = 'PRIA'
sex   = jenis
```