Alt programlar

- Bir programda aynı tür hesaplama işlemi programın farklı yer(ler)inde birden fazla kullanılabilir. Aynı işlem adımlarını bir çok kez tekrarlamak, programdaki deyim sayısını arttıracağından hem programın yavaş işlemesini, özellikle programı(n çalışması açısından) izlemeyi güçleştirir. Bu tür tekrarlanan program parçaları, ALT PROGRAM adı altında (ana programdan) ayrı bir program olarak yazılabilir.
- Bir alt program, uygun bir "çağırma" deyimi ile ana programda kullanılabilir. Fortranda kullanılabilen alt programlar 3 sınıfa ayrılır :

1-Deyim fonksiyonları

2-FUNCTION alt programı

3-SUBROUTINE alt programı

Alt programlar- Deyim Fonksiyonları

Bunlar bir tek deyimle tanımlanırlar. Bir ana programda tanımlanıp kullanılabildiklerinden başka programlar tarafından çağrılamazlar. Tanımlama deyimlerinde olduğu gibi programın hemen ilk satırlarında/başlarında yer almalıdırlar. Genel yapıları şu şekildedir:

isim (a1, a2, ..., aN) = ifade

Burada isim, deyim fonksiyonunun ismi olup bir FORTRAN değişken ismidir. Parantez içindeki **a**; nicelikleri de birer FORTRAN ismidir/değişkenidir. Bu isimler virgüllerle birbirlerinden ayrılırlar. Bu değişkenlerin her biri farklı isim taşımalıdır. Fonksiyon çağrıldığında gerçek argümanlar/değerler/veriler bunların yerine geçer. Bunlar sabit ya da değişken bir dizinin elemanları olamazlar. Eşitliğin sağ tarafındaki **ifade** ise, hesaplanacak ifade olup aritmetik ya da karakter ifadesi olabilir. **ifade**' nin tipi ile ismi uyuşmalıdır.

Örnek 1 Argümanlarının karelerinin toplamını bulan bir deyim fonksiyonu,

SUMSQ(X,Y) = X * X + Y * Y

şeklinde olsun. Bu fonksiyon programda SUMSQ ismiyle çağrılır. O zaman X ve Y argümanları gerçek değerlerini alarak ifade hesaplanır ve sonuç, çağıran deyimin ismine aktarılır. Programın çıktısı 13.0000 şeklindedir

```
SUMSQ(X,Y) = X * X + Y * Y
SONUC = SUMSQ(X,Y)
WRITE( * , * ) SONUC
```

Bir parçacığın potansiyel enerjisi PE = m*g*h denklemi ile verilmektedir. Kinetik enejisi ise E = m*v*v / 2 denklemi ile verilebilir. Parçacığın toplam enejisini TE = PE + KE denklemine göre hesaplayan Fortran programı.

```
c234567
      real pe, ke, m, g, h
      pe(h)=m*g*h
      ke(h)=0.5*m*v2
      h=100.0
      h0=h
      g=10.0
      m=1.0
      dh=10
10    if(h.lt.0)goto 20
      v2=2.0*g*(h0-h)
      te=pe(h)+ke(h)
      write(*,15)h,pe(h),ke(h),te
15    format(1x,' h :',f6.1,3(2x,f6.1))
      h=h-dh
      goto 10
20    stop
      end
```

Bir kişinin ideal kilosunun hesaplanması

```
      PROGRAM ideal_kilo
      REAL kilo, boy, fark
      fark(kilo,boy) = kilo - 90. * ( boy - 1. )
      WRITE(*,100) ' Kilonuzu (kg) giriniz : '
      READ(*,*) kilo
      WRITE(*,100) ' ve boy (m) giriniz :'
      READ(*,*) boy
      WRITE(*,*) ' ideal kilonuzdan fark ',
     * fark(kilo,boy),' Kilogram kadardır.'
100   FORMAT(1X,A,$)
      END
```

Alt programlar - FUNCTION alt programı

FUNCTION alt programı, ayrık ve tam bir FORTRAN programı olup diğer ana programlar tarafından çağırılabilir. Genel formatı şu şekildedir :

```
FUNCTION isim (a1, a2 ,..., aN)
.......
isim = bir ifade
RETURN
END
```

- Uygulama kuralları ise şu şekilde sıralanabilir:

1- FUNCTION deyimi ilk deyim olmalıdır.

2- Çağırılabileceği bir isim taşımalıdır.

3- Argümanları varsa bunlar basit FORTRAN isimleri veya indisli değişken isimleri olabilir; ancak sabit ya da dizi elemanı olamaz.

4- Hiçbir argüman taşımayabilir. Ancak bu durumda da parantezler kullanılmalıdır.

5- İsim sonuçta bir değer almalıdır; çünkü alt programda hesaplanan sonuç, aynı isim altında ana programa geri döner.

6- Birden fazla RETURN deyimi içerebilir.

7- Son deyimi END olmalıdır.

8- İsminin tipi ile hesaplanan ifadenin tipi aynı olmalıdır. Eğer ifade karakter tipinde ise, ismi de karakter tipinde olmalıdır. Bu durumda karakter isminin uzunluğu, argümanlarının uzunlukları toplamından büyük veya ona eşit olmalıdır.

Function Alt programı

Örnek: İki gerçel sayının toplamını hesaplayan bir FUNCTION alt programı. Deyim fonksiyonlarını çağırma kuralları burada da geçerlidir

```
C A, B, X, Y, Z, U YEREL DEĞİŞKENLER.
      A = 2.0
      B = 3.0
      X = TOPLAM(A, B)
      Y = SQRT(A)
      Z = TOPLAM(3.0, 4.0)
      U = TOPLAM(Z, A)
      IF(TOPLAM(X,Y).GT.5.) GOTO 10
      WRITE( * , * ) A, B, X, Y, Z, U, TOPLAM(Z, U)
      END
C
      REAL FUNCTION TOPLAM(A,B)
C A VE B YEREL DEĞİŞKENLER
      TOPLAM = A + B
      RETURN
      END
```

Function Alt programı

Örnek: 3 tane reel sayı eğer bir üçgen oluşturuyorlarsa bu üçgenin alanını hesaplayan, eğer üçgen oluşturmuyorlarsa alanın değerini 0 olarak veren bir **FUNCTION** alt programı yazınız.

```
5       READ(*,*)X, Y, Z
        IF(X.LE.0.)STOP
        SQ = ALAN(X,Y,Z)
        IF(SQ.EQ.0.) THEN
        WRITE(*, *)'üçgen oluşmaz!'
        ELSE
        WRITE(*, *)'Alan = ', SQ
        ENDIF
        GOTO 5
        END
```

```
FUNCTION ALAN(A,B,C)
ALAN = 0.0
IF(A.GE.B+C) RETURN
IF(B.GE.A+C) RETURN
IF(C.GE.A+B) RETURN
S = 0.5*(A+B+C)
ALAN = SQRT(S*(S-A)*(S-B)*(S-C))
RETURN
END
```

Function Alt programı

```
C fonksiyon alt programının kullanılması
5       READ(*,*)X
        IF(X.EQ.0) STOP
        Y=F(X)
        WRITE(*,10)X, Y
10      FORMAT(' ', F3.1, F5.2)
        GOTO 5
        END
C
        FUNCTION F(X)
        F=2.*X
        RETURN
        END
```

SUBROUTINE Alt programı

SUBROUTINE, FUNCTION gibi ana programdan ayrık bir programdır. Genel yapısı şu şekildedir

```
SUBROUTINE isim (a1, a2,..., aN)
.....
RETURN
END
```

Uygulama kuralları ise şu şekildedir :

1- İlk deyim **SUBROUTINE** kelimesi ile başlamalıdır.

2- Bir isim taşımalıdır.

3- Argümanları varsa basit değişken veya indisli değişken isimleri olabilir , ancak sabit veya bir dizinin elemanları olamaz.

4- Bu alt programdaki argümanlar "sağır" argümanlardır ; eğer yazılmazlarsa alt programda hesaplama için gerekli değerler COMMON deyimi yardımıyla ana programdan alt programa aktarılır.

5- FUNCTION alt programının aksine ismi hiç bir değer taşımaz. Sadece argümanları sayısal değerler taşır. İsim, tam sayı veya reel sayı değişken ismi olabilir.

6- En az bir tane **RETURN** deyimi içermelidir.

7- Son deyimi **END** olmalıdır.

8- Ana programda CALL deyimi yardımıyla çağırılır. Bu deyimin genel yapısı şu şekildedir: CALL isim (a, , a., ,..., aN) Buradaki isim ile **SUBROUTINE** 'deki isim aynı olmalıdır. Ancak **CALL** 'daki argümanlar sabit, basit değişken, indisli değişken, dizi elemanı, aritmetiksel bir ifade veya bunların karışımı olabilir, CALL 'daki argümanlar ile SUBROUTINE 'deki argümanlar sıra , sayı ve tip olarak uyuşmalıdır.

Subroutine Alt programı

- **Örnek :** İki gerçel sayıyı toplayıp sonucunu ana programa aktaran bir alt program.

```
SUBROUTINE TOPLA(A,B,SONUC)
SONUC = A + B
RETURN
END
```

Subroutine Alt programı

- **Örnek** Bir üçgenin alanının **SUBROUTINE** tipi alt programla hesaplanması.

```
SUBROUTINE AREA(A,B,C,ALAN)
ALAN = 0.
IF(A.GE.B+C) RETURN
IF(B.GE.A+C) RETURN
IF(C.GE.A+B) RETURN
S = 0.5*(A+B+C)
ALAN = SQRT (S * (S - A) * (S - B) * (S - C))
RETURN
END
```

Subroutine Alt programı

- Önceki örnekteki üçgenin alanını bulma probleminin **SUBROUTINE** tipi alt programla yazılması.

```
5       READ( * , * ) X , Y , Z
        IF(X.LE.0.) STOP
        CALL AREA(X,Y,Z,SQ)
        IF(SQ.EQ.0.) THEN
        WRITE( * , * )' Bu uzunluklar üçgen oluşturmuyor ! '
        ELSE
        WRITE( * , * ) 'Üçgenin alanı = ', SQ
        ENDIF
        GOTO 5
        END
```

Subroutine Alt programı

```
c histogram
      DIMENSION n1(7)
      character*8 c1(7)
C ogrenci sayilari
      DATA n1/3,6,9,7,4,2,1/
      DATA c1/'100 - 95',' 94 - 90',
     .'89 - 85',
     .' 84 - 80',' 79 - 75',
     .' 74 - 70',' 69 - 65'/
C
      WRITE(*,10)
10    FORMAT(' Aralık    Öğrenci sayısı',/
     .       ' ------    --------------')
C
      do 15 i=1,7
15    write(*,20)c1(i), n1(i)
20    format(1x,  a8,'     ',i2)
c
      write(*,25)
25    format(//)
      CALL GRAFIK(c1, n1)
      STOP
      END
```

```
      SUBROUTINE GRAFIK(c1, n1)
      DIMENSION   n1(7)
      character*8 c1(7), c2*1
      c2='*'
      do 30 i=1,7
      write(*,35)c1(i)
35    format(1x, a8,\)
30    write(*,*)(char(219), j =
     *1, n1(i))
      return
      END
```

Eğik atış hareketi yapan bir cismin hareketinin incelenmesi.

```
      DIMENSION A1(1000), A2(1000)
C BASLANGIÇ YATAY HIZ
      X0=0.0
C BAŞLANGIÇ DÜŞEY HIZ
      Y0=0.0
C BAŞLANGIÇ HIZI
      V0=50.0
C YATAYLA YAPILAN AÇI (derece)
      TH=60.0
C YERÇEKİMİ İVMESİ (m/s^2)
      G=10.0
C BAŞLANGIÇ NOKTASINA UZAKLIK
      R=0.0
C GEÇEN SÜRE
      T=0.0
C ZAMAN ARALIĞI
     DT=0.5
     A=3.141593/180.
     VX0=V0*COS(A*TH)
     VX=VX0
     VY0=V0*SIN(A*TH)
```

```
      VY=VY0
      I=1
      V=(VX**2+VY**2)**0.5
      WRITE(*,5)
5     FORMAT(‘   T    X    Y    R    Vx   Vy    V’)
10    IF(T.GT.8.6) GOTO 20
      WRITE(*,15)T,X,Y,R,VX,VY,V
15    FORMAT(‘ ‘,7(F8.3,1X))
      T=T+DT
      X=V0X*T
      Y=Y0+VY0*T-0.5*G*T**2
      VX=VX0
      VY=VY0-G*T
      R=(X**2+Y**2)**0.5
      V=(VX**2+VY**2)**0.5
      A1(I)=Y
      A2(I)=T
      I=I+1
      GOTO 10
20    N=I-1
      CALL GRAFIK(A1,A2,N)
      STOP
      END
```

```
      SUBROUTINE GRAFIK(A1, A2, N)
      DIMENSION A1(50), A2(50)
      WRITE('------------------------>'
      DO 40 I=1,N
      C=' '
      WRITE(*,25)A2(I)
25    FORMAT(' ', F7.3,'|',\)
      DO 30 J=1,NEAREST(0.2*A1(I),0.5)
30    WRITE(*,35)C
35    FORMAT(A1)
40    WRITE(*,45)
45    FORMAT('*')
      RETURN
      END
```

BLOCK DATA Alt programı

- Bu altprogram yapısal olarak **FUNCTION** ve **SUBROUTINE** alt programlarından çok farklıdır.
- Amacı **DATA** deyimleri yardımıyla isimli **COMMON** devinimdeki değişkenlere başlangıç değerlerini aktarmaktır.
- **BLOCK DATA** deyimiyle başlar, **END** deyimiyle sona erer. Hiç bir uygulanabilir deyim içermez.
- Tanımlama deyimlerinin yanısıra **DIMENSION,** isimli **COMMON** ve **DATA** deyimlerini içerir.

BLOCK DATA Alt programı

```
BLOCK DATA
COMMON /A1/ G,H,I
COMMON /A2/ P,R(2)
DATA G,H,I / 7.0,8.0,15 /
DATA P,R / 9.0,10.0,11.0/
END
```

BLOCK DATA Alt programı

```
BLOCK DATA
DIMENSION A(4),B(4)
REAL NUM(3)
CHARACTER C(4), D * 4
COMMON /AB/ A,B,NUM,K
COMMON /CHR/ C,D
DATA A / 4 * 3.0 /, B / 4 * 0.0 /
DATA NUM / 3.5,5.0,7.8 /, K / O /
DATA C /'A','B','C','D'/,D/ 'ABCD' /
END
```

COMMON deyimi

- Basit değişken ve dizilerin aynı bellek konumlarım paylaşmasını EQUFVALENCE deyimi sağlıyordu. COMMON deyimi ise ana ve alt programların ya da sadece alt programların aynı bellek alanını kullanmasını sağlar. İki farklı biçimi vardır :

- **i) İsimsiz COMMON - COMMON değişkenler**
- **ii) İsimli COMMON - COMMON /isim/ değişkenler**

COMMON deyimi

- COMMON A,B,C(3) veya DIMENSION C(3)
- COMMON A,B,C
- yazılabilir. İkincisinde DIMENSION ile COMMON deyimleri yerdeğişliremezler. Yukarıdaki deyimler, bellekte aşağıdaki yer ayırımını sağlar :

```
A
B
C(1)
C(2)
C(3)
```

- Bir programda ortak alan belirlendikten sonra tüm alt programlar veya fonksiyonlar, aynı COMMON deyimine sahip iseler, bu alandan yararlanabilirler

COMMON deyimi

```
C 1. program
COMMON A,B,C(3)
........
.........
CALL SUB1(K)
.........
END
SUBROUTINE SUB1(L)
   SUB1(L)
COMMON A,B,C(3)
...........
RETURN
END
```

```
C 2. program
COMMON A,B,C(3)
...........
...........
CALL SUB1(K)
...........
END
SUBROUTINE

COMMON X,Y,Z(3)
......
RETURN
END
```

COMMON deyimi

(Aşağıdakilerden her ikisi de yanlıştır).

```
COMMON A,B                          COMMON A,B
........                            ........
CALL SUB(A,K)                       CALL
  SUB(A,K)
END                                 END
SUBROUTINE SUB(A,L)                 SUBROUTINE
  SUB(A,L)
COMMON A,B                          COMMON A,Y
........                            ........
RETURN                              RETURN
END                                 END
```

COMMON deyimi

```
COMMON K
      COMMON A,B,C(3)
.............
      CALL ALT
      END
      SUBROUTINE ALT
      COMMON K
      COMMON A,B,C(K)
...........
      RETURN
      END
```

i) Alt programdaki argümanlar yazılmazsa COMMON deyimiyle veriler aktarılabilir. Ayrıca bu durumda sayısal olmayan boyut da tanımlanabilir.

ii) İsimsiz COMMON 'daki değişkenlere DATA deyimi yardımıyla önceden değerler verilemez.

iii) İsimli COMMON da ortaklaşa kullanılan bellek alanı yaratır; ancak tüm alt programlar bu alanı paylaşmak zorunda değildir. Ayrıca BLOCK DATA adı verilen özel bir alt program yardımıyla değişkenlerine sayısal değerler aktarılabilir. Aynı isme sahip ortak alan kullanımında tüm program birimlerindeki COMMON 'lar, aynı uzunluğa sahip değişkenler içermelidir

COMMON deyimi

i) İsimli ve isimsiz COMMON 'lar iç içe yazılabilir; ancak bu yazım şekli genellikle tercih edilmez :

```
COMMON A,B,C / ISIM / I,J,K
```

veya

```
COMMON / ISIM / I,J,K // A,B,C
```

yazılabilir. Ancak bunların yerine isimsiz ve isimli COMMON deyimlerini ayrı ayrı satırlara yazmak daha uygundur :

```
COMMON A,B,C
COMMON /ISIM/I, J, K
```

COMMON deyimi

ii) Aynı programda iki COMMON deyiminde aynı değişkenler kullanılamaz :

```
COMMON A,B,C
COMMON / XYZ / X(5),A,Z
```

İsimli ve isimsiz COMMON Deyimi

isimsiz COMMON	isimli COMMON
Programda sadece bir tek isimsiz COMMON alanı vardır. Bu alan birden fazla isimsiz COMMON tarafından kullanılabilir.	Herbiri kendi ismini ve boyutunu taşıyan bir çok isimli COMMON kullanılabilir.
Farklı program birimlerinde isimsiz COMMON, farklı büyüklüklerde olabilir.	İsimli COMMON deyimini kullanan her program birimi, aynı uzunluktaki alanı kullanmak zorundadır.
Değişkenlerine değer aktarılamaz.	Değişkenlerine değer aktarılabilir.