Fonksiyonel Programlama

Lisp Programlama Dili

İlk program: Ekrana merhaba yazmak.

```
(Defun MerhabaYaz()
"Merhaba"
)
```

Lisp programlama dilinde fonksiyon tanımlamaları **Defun** kelimesi ile yapılır. Ve her komut parantez içerisinde yazılmalıdır.

(**load** "C:/Users/HH/Desktop/PDP/Lisp/merhaba.lisp")

(**compile-file** "C:/Users/HH/Desktop/PDP/Lisp/merhaba.lisp")

Lisp Programlama Dilinde Matematiksel İşlemler

Örnek: Dört işlemi gerçekleştiren Lisp kodu

```
(Defun Hesap(x y)

;Sayıların toplamı
(print "Toplami")
(setq sonuc (Topla x y))
(print sonuc)

;Sayıların farkı
(print "Farki")
(setq sonuc (Cikar x y))
(print sonuc)

;Sayıların Çarpımı
(print "Carpimi")
(setq sonuc (Carp x y))
(print sonuc)

;Sayıların Bölümü
(print "Bolumu")
(Bol x y)
)

(Defun Topla(x y)
(+ x y)
)
(Defun Cikar(x y)
(- x y)
)

(Defun Carp(x y)
(* x y)
)

(Defun Bol(x y)
(/ x y)
)
```

Örnek: Kullanıcıdan değer alınması

```
(Defun Kare()
(print "X:")
(setq x (read))
(print "Y:")
(setq y (read))
(print "Sonuc:")
(+ (* x x) (* y y))
)
```

setq ile birden çok değişkene değer atanabilir. Örneğin (**setq** x 10 y 32) gibi.

(**incf** x)

(**decf** x)

(**rotatef** x y)

(#b001) komutu yazılan 0 ve 1 lerin ikili bir ifade olduğunu belirtir ve ekrana onun 10 tabanındaki halini yazar örneğin (#b001) ekrana 3 yazacaktır.

#c ifadesi yazılacak sayının karmaşık sayı olduğunu belirtir. Örneğin (+ #c(1 2) #c(5 2)) yazdığınızda ekrana iki karmaşık sayının toplamını yani #c(6 4) yazacaktır.

Lisp Programlama Dilinde Kontrol Blokları

Örnek: Girilen yaş değerinin 2 ile 18 arasında olup olmadığını kontrol ediyor.

```
(Defun Cocukmu()
( setq yas (read))
(if
(and (> yas 2) (< yas 18)) T
nil
)
)
```

Yukarıdaki kod bloğunda **T** doğruyu ifade ederken **nil** yanlışı ifade etmektedir.

Örnek: Verilen üç sayıdan en büyüğünü bulan fonksiyon.

```
(Defun EnBuyuk(x y z)
    (if (and (> x y) (> x z)) x
        (if (and (> y x) (> y z)) y
            (if (and (> z x) (> z y)) z
                "esitlik var"
            )
        )
    )
)
```

Lisp Programlama Dilinde Döngüler

- For Döngüsü: Aşağıya ya da yukarı doğru saydırılabilir.

Örnek: 1 den 10'a kadar ekrana tam sayıları yazmak.

```
(Defun Artan()
    (loop for i from 1 to 10 do
        (print i)
    )
)
```

Örnek: Girilen sayıya kadar tersten tek sayıları yazdırmak

```
(Defun Tek()
    (setq sayi (read))
    (loop for i from sayi downto 1 do
        (if (/= (mod i 2) 0) (print i))
    )
)
```

Yukarıdaki kod bloğunda **/=** ifadesi eşit değil operatörüdür. Yani diğer programlama dillerinde kullanılan != gibi bir operatördür. **mod** ifadesi ise pascalda olduğu gibi sayının modunu alır.

Örnek: Faktöriyel Lisp kodu

```
(Defun Fakt(x)
    (setq sonuc 1);
    (loop until (< x 1) do
        (setq sonuc (* sonuc x))
        (setq x (- x 1))
    )
    sonuc
)
```

Örnek: Fibonacci hesabını özyineleme olmadan hesaplama

```
(Defun Fib(x)
    (setq onceki -1)
    (setq sonuc 1)
    (setq toplam 0)
    (setq i 0)
    (loop while (<= i x) do
        (setq toplam (+ sonuc onceki))
        (setq onceki sonuc)
        (setq sonuc toplam)
        (setq i (+ i 1))
    )
    sonuc
)
```

- Fonksiyonlarda opsiyonel parametre tanımlama

```
(Defun fonk(x &optional y z)
 (list x y z)
)
```

```
(Defun fonk(&key x y z)
 (list x y z)
)
```

Yukarıdaki fonksiyonda şöyle bir çağrım yapıldığında (fonk :x 3 :z 5) ilk parametreye 3, ikinci parametre opsiyonel geçilir ve üçüncü parametreye 5 değeri verilir.

Aynı if içerisinde birden çok komut çalıştırılmak isteniyorsa hepsi aynı paranteze alınır ve başlarına **progn** komutu getirilir.

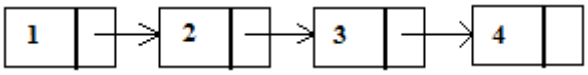
Örnek:

```
(Defun NickAta(&optional isim)
    (if isim
        (setq nick (concatenate 'string isim "123"))
        (progn (setq isim "")
               (setq nick (concatenate 'string isim "123"))
        )
    )
    nick
)
```

Listeler:

Listeler lisp programlama dilinde büyük rol oynar. Örneğin 4 elemanlı A isminde bir liste tanımlamak için,

(**defparameter** *A* (list 1 2 3 4))



(**first** *A*) komutu listenin ilk elemanını getirecektir, yani ekrana 1 yazacaktır.

(**rest** *A*) komutu listenin o andaki konumundan sonrasını getirecektir, yani ekrana (2 3 4) yazacaktır.

İki farklı liste **append** komutu ile birleştirilebilir.

(append *A* *B*) komutu A listesinin sonuna B listesini ekleyecektir.

Bir listeyi **reverse** komutu ile tersine çevirebilirsiniz.

(**setq** *A* (**reverse** *A*))

Listede bir eleman aramak için **find** komutu kullanılır. Eğer listede aranan eleman yoksa nil döndürecektir. Var ise elemanı döndürecektir.

(**find** 4 *A*)

Listedeki bir elemanın konumunu döndürmek için position komutu kullanılır. Yukarıdaki liste örneği düşünüldüğünde aşağıdaki kod ekrana 1 getirecektir. Çünkü 2 elemanı 1. index'te.

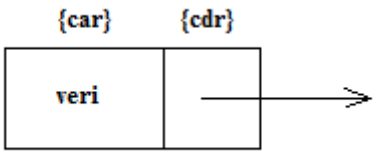
(**position** 2 *A*)

Listedeki bir elemanı silmek için **delete** komutu kullanılır. Yukarıdaki liste örneği düşünüldüğünde aşağıdaki kod 4'ü silecektir.

(**delete** 4 *A*)

İki listenin eşit olup olmadığını karşılaştırmak için **eq** komutu kullanılır. Eşit ise T, değil ise nil dönecektir.

(**eq** *a* *b*)



Bir liste düğümünde verinin olduğu bölüm lisp programlama dilinde **car** ile ifade edilir. Diğer düğümü gösteren bölüm ise **cdr** ile ifade edilir.

Örnek: Listenin belirli bir konumuna eleman ekleme

```
(Defun Ekle(liste index x)
    (push x (cdr (nthcdr index liste)))
 liste
)
```

Yukarıdaki metot lisp'e tanıtıldıktan sonra bir liste tanımı yapılırsa,

(**defparameter** *p* (list 1 2 3 4))

daha sonra,

(Ekle *p* 1 10)

1. indeks'e 10 sayısını ekleyecektir. Listenin yeni görünümü (1 2 10 3 4) olacaktır.

Örnek: Listenin belirli bir konumundan eleman çıkarma

```
(Defun Cikar(liste index)
    (delete (first (subseq liste index)) liste)
 liste
)
```