Listas

Listas Simplemente enlazadas

Definicion de un nodo

```
Nodo = Registro
    Dato: <tipo>
    Prox: Puntero a Nodo
Fin Registro

Prim, p, a, q: puntero a Nodo;

valor: <tipo>
```

Búsqueda

```
LEER (valor)

p:= Prim
MIENTRAS (p <> nil) y (*p.dato <> valor) HACER
    p = *p.prox
FIN MIENTRAS

SI (p <> nil) ENTONCES
    Escribir ('Se encontro el elemento')
SINO
    Escribir ('No se encontro el elemento')
FIN SI
```

Carga Apilada

```
LEER (valor)

Prim:= nil
MIENTRAS (valor <> '*') HACER
```

```
    NUEVO (p)
    *p.Dato:= valor
    *p.Prox:= Prim
    Prim:= p
    LEER(valor)
FIN MIENTRAS
```

Carga Encolada

```
LEER (valor)

Prim:= nil
MIENTRAS (valor <> '*') HACER
    NUEVO (p)
    *p.Dato:= valor
    *p.Prox:= nil

    SI Prim = nil ENTONCES
        Prim = p
    SINO
        *a.Prox:= p
    FIN SI
    a:=p
    LEER(valor)
FIN MIENTRAS
```

Carga Ordenada

```
Prim:= nil
A:= nil;

LEER (valor)
NUEVO (p)
*p.dato:= valor
q:= Prim

MIENTRAS (q <> nil) y (*q.dato < valor) HACER
    a:= q
    q:= *q.prox
FIN MIENTRAS

SI (a = nil) ENTONCES
    *p.prox:= Prim
    Prim:= p
SINO
    *p.prox:= q
```

```
    *a.prox:= p
FIN SI
```

Eliminacion

```
LEER (valor)

q:= Prim

MIENTRAS (q <> nil) y (*q.dato <> valor) HACER
    a:= q
    q:= *q.prox
FIN MIENTRAS

SI (q = nil) ENTONCES
    Escribir ('Error, el valor no existe')
SINO
    SI (a = nil) ENTONCES
        Prim = *q.prox
    SINO
        *a.prox:= *q.prox
    FIN SI
    DISPONER(q)
FIN SI
```