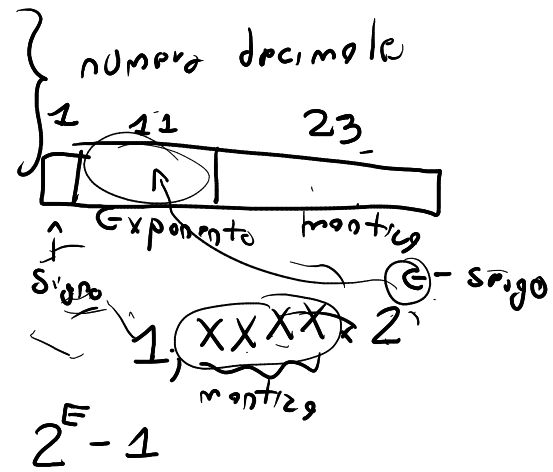


Tipos de datos {

- int \leftarrow 32 bits $- 2^{31}$ hasta $2^{31} - 1$
- long \leftarrow 64 bits $- 2^{63}$ hasta $2^{63} - 1$
-
- double \leftarrow 64 bits
- float \leftarrow 32 bits

\leftarrow 128 bits



boolean { 8 bits

String

collection caracteres

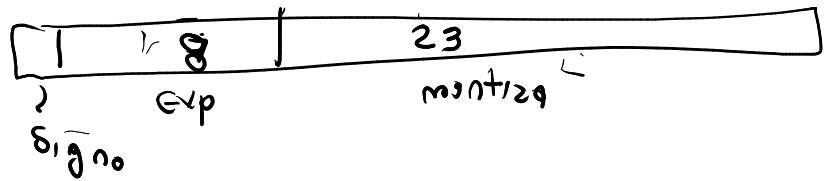
Codificación 7 bits, 8 bits } ASCII.
Normal, Extended

TAD

Implementación

Codifica en memoria

Float - 32 bits



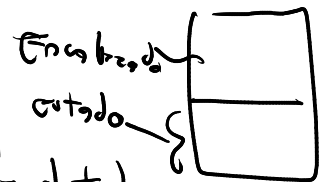
Interfaz

{ Lo que vemos cuando trabajamos
jamas el tipo de dato

32.5

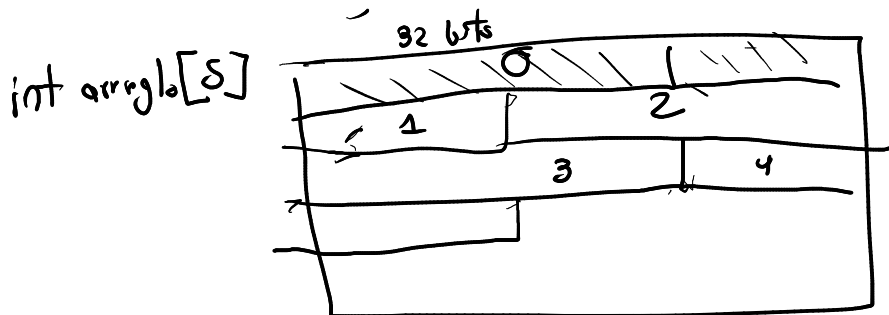
+ - * / ** % //

Arreglo / Array



Reserva de espacio de memoria (tipo de dato)

No se contienen los valores sino la dirección de memoria



RAM

`arreglo[0]`

`dir(arreglo) + 0`

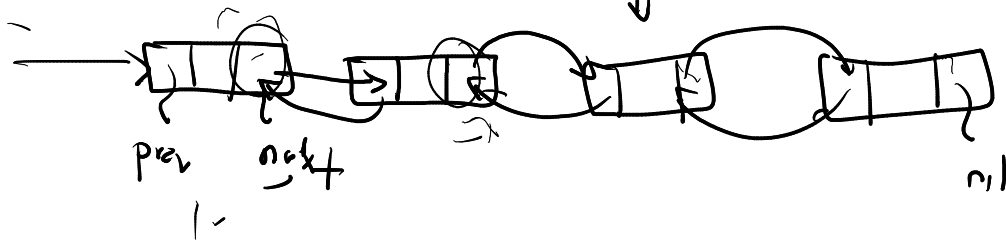
`arreglo[1]`

`dir(arreglo) + 32`

Facile de acceder

Puede de modificar su estructura (borrar/insertar)

Listas

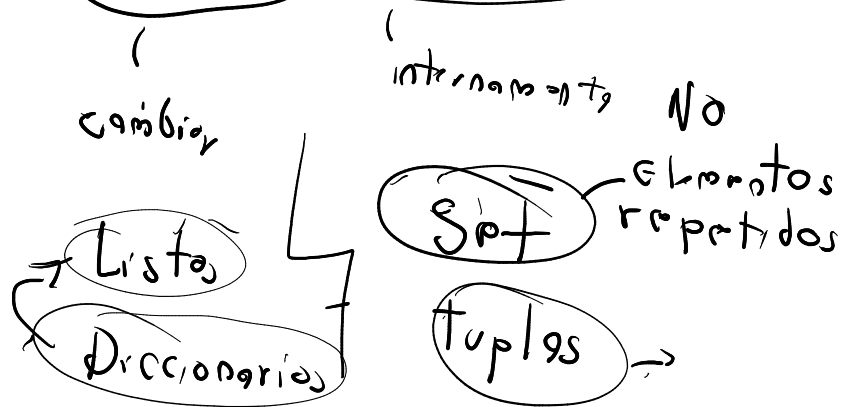


— Costoso de acceder

Valor = Se crea una nueva estructura `list[:3]`

In place : Modifica la estructura `append`
`sort`

Estructuras mutables y no mutables



Estructuras ordenadas (secuenciales) y estructuras no ordenada

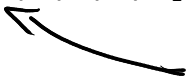
listas	[0]		Iterador
arrays	[1]		
tupla	[2]		
	⋮		

Paso de parámetros de estructuras: Listas, arreglos, colas, pilas, etc

```
def f(lista):  
    listaF = lista.copy()  
    listaF[2] = 30
```

```
listaA = [1,2,3,4,5]  
f(listaA)
```

Referencia



copy() } Python

clone() } Java

Pilas

LIFO (Ultimo en entrar primero en salir)

Push (coloca en la cabeza de la pila)

Pop (Retirar el primero de la pila)

```
pila.push(5)
```

```
pila.push(8)
```

```
pila.pop() //Retorna 8
```

Colas FIFO (Primero en entrar, primero en salir)

Enqueue (Encolar, colocar al final de la cola)

Dequeue (Desencolar, quierar el primero de la cola)

```
cola.enqueue(5)
```

```
cola.enqueue(8)
```

```
cola.dequeue() //Retorna 5
```

Recursión. Funciones que se llaman a sí mismas.

Caso base: Es donde la función tiene una respuesta inmediata

Caso recursivo: Es donde la función se llama a si misma, y la entrada cambia llevandola hacia el caso base

$$f_{9c}(n) = \begin{cases} 1 & n = 1 \\ n \times f_{9c}(n-1) & n > 1 \end{cases}$$

$$n \geq 1$$

Recursión de cabeza

```
def fac(n):  
    if n==1:  
        return 1  
    else:  
        return n*fac(n-1)
```

$$\begin{aligned} \text{fac}(10) &= 10 \times \text{fac}(9) \\ \text{fac}(9) &= 9 \times \text{fac}(8) \\ &\vdots \end{aligned}$$

Recursión de cola

```
def fac(n, res=1):  
    if n==1:  
        return 1  
    else: return fac(n-1, n*res)
```

tail recursion

$$\begin{aligned} \text{fac}(10) &= \text{fac}(9, 1 \times 10) \\ \text{fac}(9, 10) &= \text{fac}(8, 9 \times 10) \end{aligned}$$