

Listas

Len(): longitud

List(): copiar lista

L.append() : unir un elemento

L.sort() : ordenar

Strange(j,k,d) : lista de enteros de sage

Range() : enteros de Python

.reverse() : dar la vuelta

L.extend([...]) : unir a la lista

Sum([...]) : sumar elementos de la lista

.count() : contar elementos

.index () : posición de un elemento

.pop(): borra por posiciones

.remove() :borrar elemento

.insert() : añadir en una posición concreta

k.digits(base=10) : crea la lista de los dígitos de k

Cadena

Str() : crea una cadena a partir de k

Frase.split(',') : separar la frase por ','

.join() : unir

.find() : el primer índice donde aparece la subcadena

.lower() : simplifica el texto (sin mayúsculas...)

Conjuntos

Set() : crear conjunto

C.difference(a) : elementos de C diferentes de a

A|B : unión

A%B / A.intersection(B) : intersección

A.add() : añadir elemento

A.remove(): borrar

A.update() : añadir elementos

min(str) : la letra mas baja del alfabeto

max(str) : la letra mayor del alfabeto

Primos

Is_prime() : booleano

Next_prime() : siguiente primo

Nth_prime(m) : Primo numero m

Prime_range(,) : lista de primos en el rango

Primes(,) : solo para iterar en ese rango

Orbita

Def orbita(ini,f) :

```
L=[]
```

```
While not ini in L :
```

```
    L.append(ini)
```

```
    Ini=f(ini)
```

```
Return L
```

def orbita(ini,N,f):

```
L=[ini]
```

```
for _ in srange(N) :
```

```
    ini=f(ini)
```

```
    L.append(ini)
```

```
return L
```

Cambio


Tupla  Lista : list()

Tupla  Conjunto: set()

Cadena  Lista: list()

Lista  Tupla: tuple()

Lista  Cadena : str()

Diccionario  lista de pares : D.items()

Cadena  entero: int()

Lista de pares  Diccionario:

```
def convert list dict(L):
```

```
    dict = {}
```

```
    for item in L:
```

```
        dict[item[0]]=item[1]
```

```
    return dict
```

Dadas dos listas, L1 y L2, de la misma longitud podemos formar una lista de pares mediante `zip(L1,L2)`, y transformar esta lista en diccionario mediante la función del apartado anterior.

General

`Srange()` : genera una lista

`Xsrange()` : no la genera solo la recorre (para bucles de rangos altos)

`Floor()` : parte entera

`Abs()` : valor absoluto

`Plot()` : grafica

`Sqrt()` : raíz cuadrada

`Max(x,y)` : máximo

`Min(x,y)` : mínimo

`Gcd()` : MCD

`Lcm()` : MCM

`.divisors()` : divisores

`Randint(n,m)` : números al azar entre n y m (no enteros de sage, hay que quitar ' ' A las funciones que se usan)

`[randint(n,m) for j in xrange (k)]`: lista aleatoria

`Var('n')` : definir variable

`permutations([])`

Induccion

Def comprobar(N):

`var('n')` (variable que se usa para la función)

`L=[]` (comprueba si se cumple para los términos anteriores a N mediante 0 ó 1)

`S=1`

`F(n)=` (definimos la formula a probar)

`If N==1` : (primer termino)

`If F(1)==` (igual al primer elemento el cual ya sabemos)

`L.append(1)`

`return S,L`

```
else:  
    L.append(0)  
    Return S,L
```

Else:

S,L=comprobar(N-1) (si no es el primero que compruebe el anterior)

S+= (N con la operación correspondiente)

If S==(F(n=N)) :

Sumar 1 a la lista L y devolver S,L

Else:

Sumar 0 a la lista y devolver S,L

Contar elementos

```
hamlet=Hamlet.lower()
```

```
texto=list(set(hamlet))
```

```
veces=[(j,hamlet.count(j)) for j in texto]
```

```
frecuencia=dict(veces)
```

```
[(vocal,frecuencia[vocal]) for vocal in 'aeiou']
```