PONTIFICIA
UNIVERSIDAD
CATÓLICA
DE CHILE

INTRODUCCIÓN A LA PROGRAMACIÓN

Objetivos

- Ejemplos de listas
  - Ejemplos de código
  - Tres problemas resueltos
  - Aterrizar estrategia de resolución de problemas
- Listas de Listas
  - Motivación
  - Uso básico
  - Aplicaciones – casos particulares

Ejemplos de Listas

Ejemplos de código

Código

```
L = list()
L.append(10)
L.append(100)
L.append(1000)
print(L)
```

Resultado

```
[10, 100, 1000]
```

Código

```
L = list()
L.insert(0,10)
L.insert(0,100)
L.insert(0,1000)
print(L)
```

Resultado

```
[1000, 100, 10]
```

Ejemplos de código

Código

```
L = list("HOLA")
L.pop(2)
L.pop(0)
print(L)
```

Resultado

```
['O', 'A']
```

Código

```
L = list()
L.extend( range(5) )
L.extend( "AEIOU" )
print(L)
```

Resultado

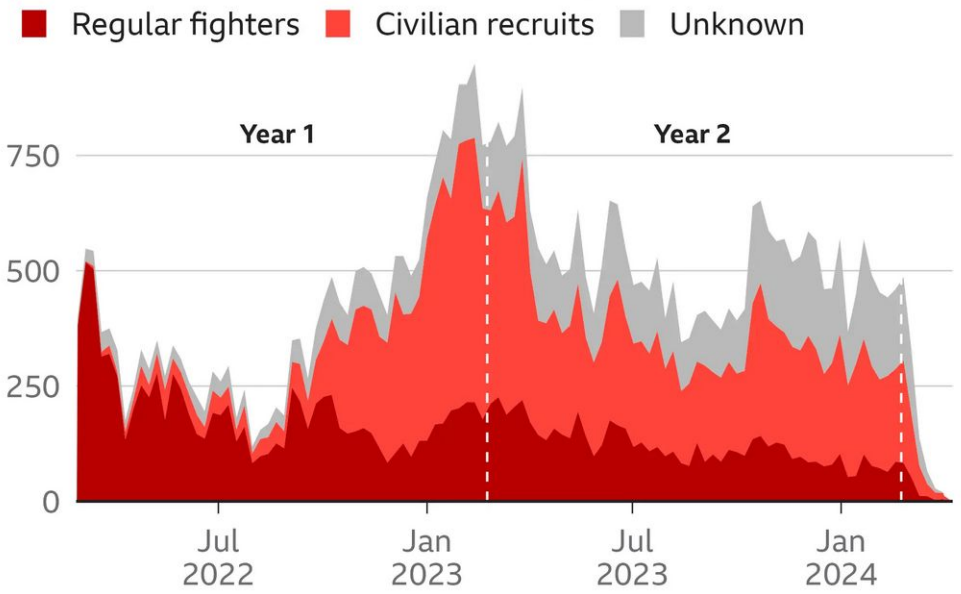
```
[0, 1, 2, 3, 4, 'A', 'E',
'I', 'O', 'U']
```

PONTIFICIA UNIVERSIDAD CATÓLICA DE CHILE

Bajas acumuladas

Over 50,000 Russians have died in Ukraine

Weekly counts of deaths confirmed by the BBC



Regular fighters include Wagner contractors. Civilian recruits include volunteers, draftees and inmates. Data to week ending 7 Apr 2024

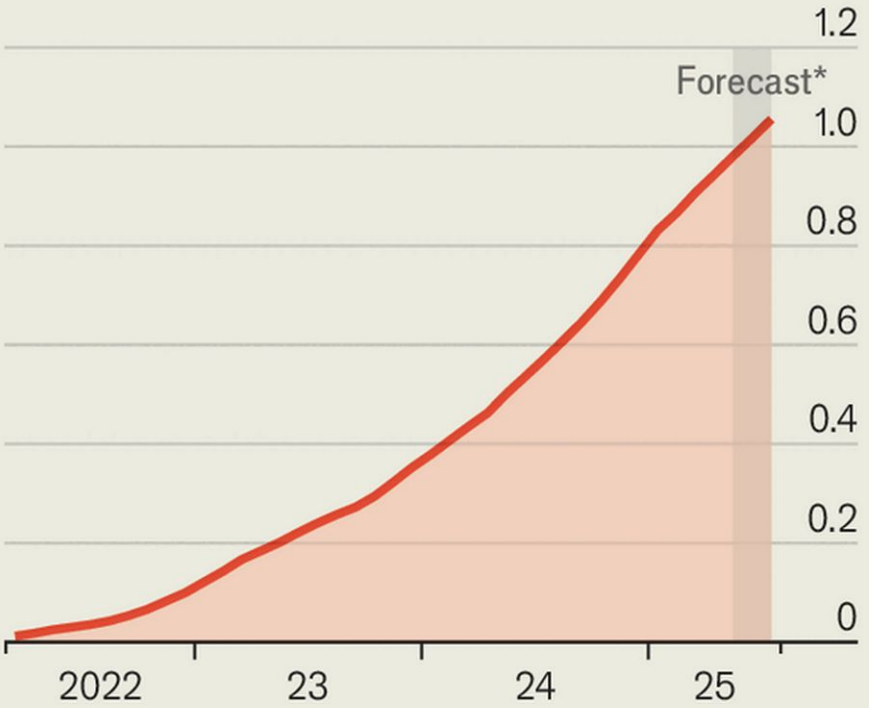
Source: BBC analysis of data from BBC News Russian / Mediazona

BBC

Grim toll

1

Russian soldier casualties in Ukraine war, m



*Assuming April 2025 rate continues

Sources: UK MOD; *The Economist*

Bajas acumuladas

- **Problema**: implemente una función, de nombre **`acum`**, que reciba como argumento una lista de números, que representa una serie de tiempo de bajas diarias, y que retorne otra lista de números, pero con las bajas acumuladas diarias.
- Por ejemplo, `acum( [0, 5, 10, 2, 3] )` debe retornar la lista `[0, 5, 15, 17, 20]`.

Bajas acumuladas

- **Problema**: implemente una función, de nombre **`acum`**, que reciba como argumento una lista de números, que representa una serie de tiempo de bajas diarias, y que retorne otra lista de números, pero con las bajas acumuladas diarias.
- Por ejemplo, `acum( [0, 5, 10, 2, 3] )` debe retornar la lista `[0, 5, 15, 17, 20]`.

PONTIFICIA UNIVERSIDAD CATÓLICA DE CHILE

Bajas acumuladas

- **Problema**: implemente una función, de nombre **`acum`**, que reciba como argumento una lista de números, que representa una serie de tiempo de bajas diarias, y que retorne otra lista de números, pero con las bajas acumuladas diarias.
- Por ejemplo, a[illegible] debe retornar la lista `[0, 5,`

```
def acum( bajas_dia ):
    resp = []

    return resp
```

Bajas acumuladas

- **Problema**: implemente una función, de nombre **`acum`**, que reciba como argumento una lista de números, que representa una serie de tiempo de bajas diarias, y que retorne otra lista de números, pero con las bajas acumuladas diarias.
- Por ejemplo, `acum( [0, 5, 10, 2, 3] )` debe retornar la lista `[0, 5, 15, 17, 20]`.
- Bajas acumuladas se construyen para cada número --> bucle for
- Debemos sumar cada número recibido y debemos anexar a lista respuesta

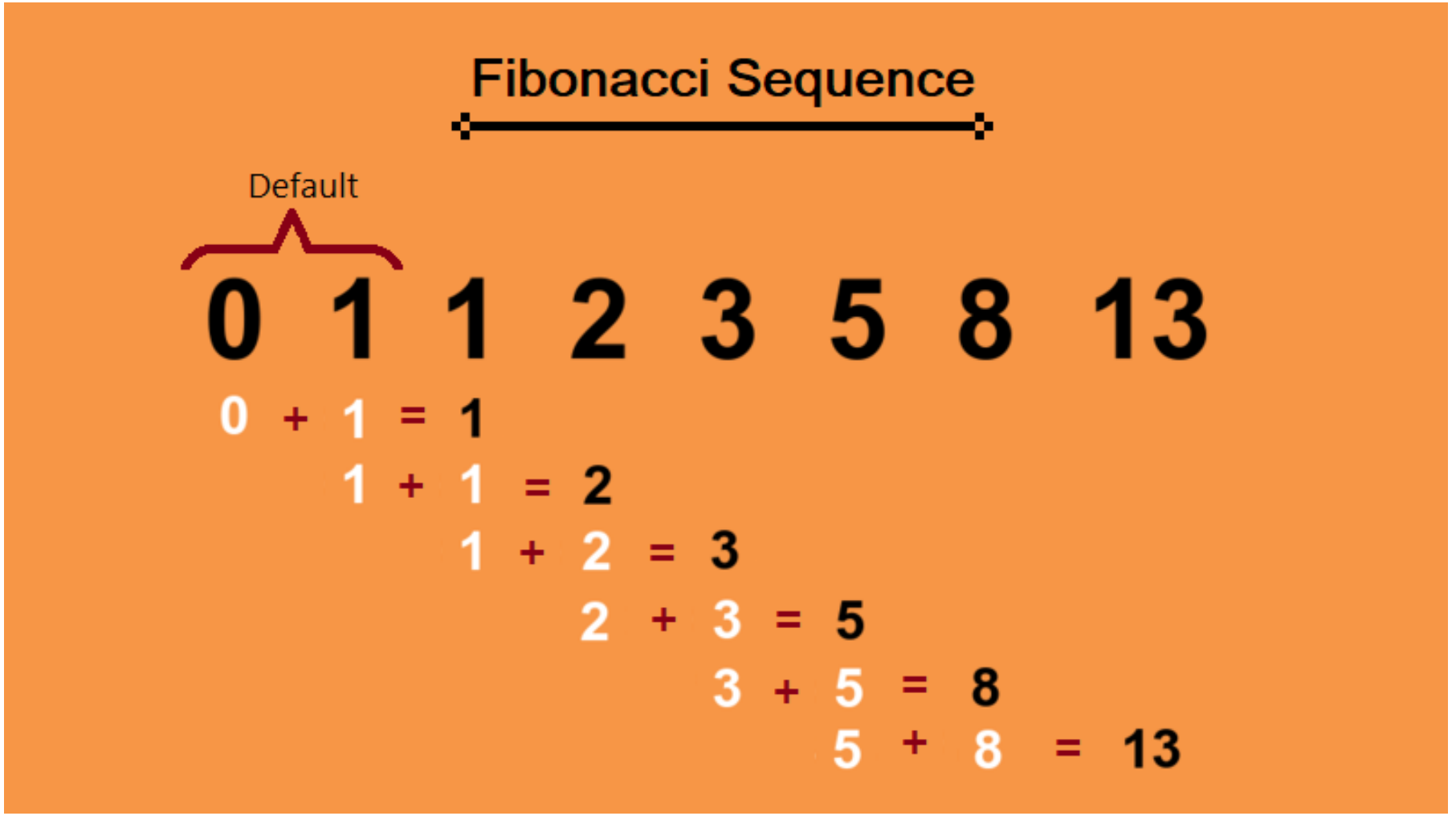
PONTIFICIA UNIVERSIDAD CATÓLICA DE CHILE

Bajas acumuladas

- **Problema**: implemente una función, de nombre **`acum`**, que reciba como argumento una lista de números, que representa una serie de tiempo d ta de números, pero con las
- Por ejemplo ebe retornar la lista `[0, 5`
- Bajas acum nero --> bucle for
- Debemos su os anexar a lista respuesta

```
def acum( bajas_dia ):
    resp = []
    z = 0
    for x in bajas_dia:
        z += x
        resp.append( z )
    return resp
```

Fibonacci


Fibonacci Sequence
Default
0 1 1 2 3 5 8 13
0 + 1 = 1
1 + 1 = 2
1 + 2 = 3
2 + 3 = 5
3 + 5 = 8
5 + 8 = 13

Fibonacci

- La serie de Fibonacci comienza con el 0 (cero), luego el 1 (uno) y los siguientes números se construyen a partir de la suma de los dos números anteriores. Por eso siguen 1, 2, 3, 5, 8, etc.
- Implemente la función **`LFIB`**, la cual recibe un argumento **`N`**, que es un número entero, y que retorne una lista con los **`N`** primeros números de Fibonacci.
- Ejemplo, `LFIB(5)` retorna `[0, 1, 1, 2, 3]`, mientras que `LFIB(1)` retorna `[0]` y `LFIB(0)` retorna `[]`.

PONTIFICIA UNIVERSIDAD CATÓLICA DE CHILE

Fibonacci

- La serie de Fibonacci comienza con el **0** (cero), luego el **1** (uno) y los siguientes números se construyen a partir de la suma de los dos números anteriores. Por eso siguen **1**, **2**, **3**, **5**, **8**, etc.
- Implemente la función **`LFIB`**, la cual recibe un argumento **`N`**, que es un número entero, y que retorne una lista con los **`N`** primeros números de Fibonacci.
- Ejemplo, `LFIB(5)` retorna `[0, 1, 1, 2, 3]`, mientras que `LFIB(1)` retorna `[0]` y `LFIB(0)` retorna `[]`.

PONTIFICIA UNIVERSIDAD CATÓLICA DE CHILE

Fibonacci

- La serie de Fibonacci comienza con el 0 (cero), luego el 1 (uno) y los siguientes números se construyen a partir de la suma de los dos números anteriores. Por eso siguen 1, 2, 3, 5, 8, etc.
- Implemente la función **`LFIB`**, la cual recibe un argumento **`N`**, que es un número entero, y que retorne una lista con los **`N`** primeros números de Fibonacci.
- Ejemplo, `LFIB(5)` retorna `[0, 1, 1,` `LFIB(1)` retorna `[0]` y `LFIB(0)` retorn

```
def LFIB(N):
    resp = []

    return resp
```

Fibonacci

- Aterricemos el algoritmo
- Casos especiales:
  - Para N = 0, retornar la lista vacía `[]`
  - Para N = 1, retornar la lista `[0]`
  - Para N = 2, retornar la lista `[0, 1]`
- Luego:
  - Desde `[0, 1]` debemos agregar `N-2` elementos
  - Agregar nuevo elemento basándonos en los dos últimos:

```
x = resp[-1] + resp[-2]
resp.append( x )
```

Fibonacci

- Aterricemos el algoritmo
- Casos especiales:
  - Para N = 0, retornar la lista vac
  - Para N = 1, retornar la lista [0
  - Para N = 2, retornar la lista [0
- Luego:
  - Desde [0, 1] debemos agre
  - Agregar nuevo elemento basá

```
x = resp[-1] + resp[-
resp.append( x )
```

```
def LFIB(N):
    if N <= 0:
        return []
    if N == 1:
        return [0]
    if N == 2:
        return [0,1]
    resp = [0,1]
    for i in range( N-2 ):
        x = resp[-1] + resp[-2]
        resp.append( x )
    return resp
```

PONTIFICIA UNIVERSIDAD CATÓLICA DE CHILE

Varianza poblacional

Fórmulas de Varianza y Desviación Estándar			
	Varianza	Desviación Estándar	Media
Población	$\sigma^2 = \frac{\sum_{i=1}^{N}(x_i - \mu)^2}{N}$	$\sigma = \sqrt{\sigma^2} = \sqrt{\frac{\sum_{i=1}^{N}(x_i - \mu)^2}{N}}$	$\mu = \frac{\sum_{i=1}^{N} x_i}{N}$
Muestra	$s^2 = \frac{\sum_{i=1}^{n}(x_i - \bar{x})^2}{n-1}$	$s = \sqrt{s^2} = \sqrt{\frac{\sum_{i=1}^{n}(x_i - \bar{x})^2}{n-1}}$	$\bar{x} = \frac{\sum_{i=1}^{n} x_i}{n}$

Varianza poblacional

- Implemente una función, de nombre `simvarianza`, que reciba una lista de números como argumento/parámetro y que retorne la varianza poblacional de esos números.
- La varianza poblacional se puede calcular así: si A es el promedio simple de esos números y B es el promedio del cuadrado de esos números, entonces la varianza poblacional es $B - A^2$.
- Por ejemplo, para [10, 7, 4, 1, 11, 20, 15, 10, 14, 6], tenemos que A = 9.8 y B = 124.4. La varianza muestral es $124.4 - 9.8^2 = 28.36$.

Varianza poblacional

- Implemente una función, de nombre `simvarianza`, que reciba una lista de números como argumento/parámetro y que retorne la varianza poblacional de esos números.
- La varianza poblacional se puede calcular así: si A es el promedio simple de esos números y B es el promedio del cuadrado de esos números, entonces la varianza poblacional es $B - A^2$.
- Formalidad:
  - Hay que definir una función **simvarianza**
  - La función **recibe un argumento** (una lista de números)
  - La función **retorna un número**

PONTIFICIA UNIVERSIDAD CATÓLICA DE CHILE

Varianza poblacional

- Implemente una función, de nombre `simvarianza`, que reciba una lista de números como argumento/parámetro y que retorne la varianza poblacional de esos núme
- La varianza poblacional se puede c
simple de esos números y B es el p
números, entonces la varianza pob
- Formalidad:
  - Hay que definir una función **simvarianza**
  - La función **recibe un argumento** (una lista de números)
  - La función **retorna un número**

```
def simvarianza( L ):
    resp = 0

    return resp
```

PONTIFICIA UNIVERSIDAD CATÓLICA DE CHILE

Varianza poblacional

- Implemente una función, de nombre **`simvarianza`**, que reciba una lista de números como argumento/parámetro y que retorne la varianza poblacional de esos números.
- La varianza poblacional se puede calcular así: si A es el promedio simple de esos números y B es el promedio del cuadrado de esos números, entonces la varianza poblacional es $B - A^2$.
- Algoritmo:
  - "esos números" --> todos esos números --> bucle **for**
  - Variables intermedias: A y B
  - Fórmula final: $B - A^2$.

PONTIFICIA UNIVERSIDAD CATÓLICA DE CHILE

Varianza poblacional

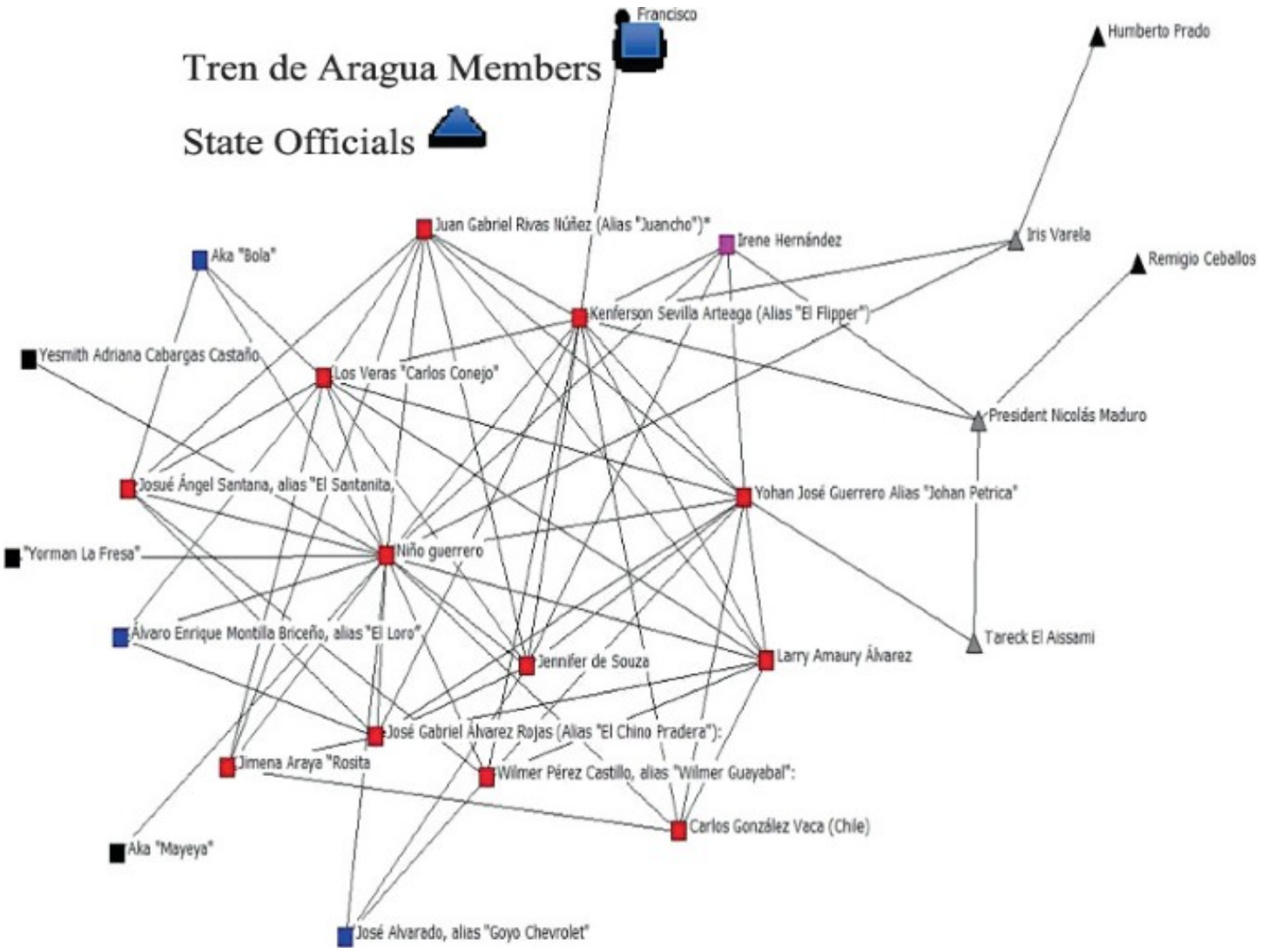
- Implemente una función, de nombre **`simvarianza`**, que reciba una lista de núme
```
def simvarianza( L ):
    A = 0
    B = 0
    for num in L:
        A += num
        B += num**2
    A = A / len(L)
    B = B / len(L)
    resp = B - A**2
    return resp
```
ro y que retorne la varianza poblacio
- La varianza poblac A es el promedio simple de esos nú cuadrado de esos números, entonces – A^2.
- Algoritmo:
  - "esos números" for
  - Variables interm
  - Fórmula final: $B - A^2$.

Listas de Listas

PONTIFICIA UNIVERSIDAD CATÓLICA DE CHILE

Aplicaciones

- Tablas
- Indicadores
- Bases de datos
- Matrices
- Mapas
- Redes



PONTIFICIA UNIVERSIDAD CATÓLICA DE CHILE

Dimensión*	IPM Global (PNUD/OPHI)	CEPAL	Chile	Colombia	Costa Rica	Ecuador	El Salvador	Honduras	México
Educación	X	X	X		X	X	X	X	
Condiciones educativas del hogar				X					
Rezago educativo									X
Salud	X		X	X	X			X	
Acceso a los servicios de salud									X
Salud, servicios básicos y seguridad alimentaria							X		
Salud, agua y alimentación						X			
Acceso a la alimentación									X
Condiciones de la niñez y la juventud				X					
Ingreso									X
Estándar de vida	X	X							
Servicios básicos		X							
Accesos a los servicios básicos de la vivienda									X
Acceso a servicios públicos y condiciones de la vivienda				X					
Vivienda y uso de internet					X				
Condiciones de la vivienda							X		
Vivienda		X						X	
Calidad y espacios de la vivienda									X
Vivienda y entorno			X						
Habitat, vivienda y ambiente sano						X			
Calidad del hábitat							X		
Redes y cohesión social			X						
Empleo				X	X			X	
Empleo y protección social		X							
Trabajo y seguridad social			X			X	X		
Acceso a la seguridad social									X
Protección social					X				

* Se usan los nombres exactos escogidos por los países para definir sus dimensiones


DAY 1
Sami

Lista de listas

```
L = [   [ 1, 5,  2  ],
        [ 0, 0, 7,  1 ],
        [ 3, 2,  1  ]  ]
```

- El código anterior define una lista de listas

¿Cuánto vale?

- `len(L)    ==`
- `len(L[0]) ==`
- `len(L[1]) ==`
- `L[0]      ==`
- `L[1][-1]  ==`

Lista de listas

```
L = [   [ 1, 5,  2  ],
        [ 0, 0, 7,  1 ],
        [ 3, 2,  1  ]  ]
```

- El código anterior define una lista de listas

¿Cuánto vale?

- `len(L)    ==` 3
- `len(L[0]) ==`
- `len(L[1]) ==`
- `L[0]      ==`
- `L[1][-1]  ==`

Lista de listas

```
L = [   [ 1, 5,  2  ],
        [ 0, 0, 7,  1 ],
        [ 3, 2,  1  ]  ]
```

- El código anterior define una lista de listas

¿Cuánto vale?

- `len(L)    == 3`
- `len(L[0]) == 3`
- `len(L[1]) ==`
- `L[0]      ==`
- `L[1][-1]  ==`

Lista de listas

```
L = [   [ 1, 5,  2  ],
        [ 0, 0, 7,  1 ],
        [ 3, 2,  1  ]  ]
```

- El código anterior define una lista de listas

¿Cuánto vale?

- `len(L)    == 3`
- `len(L[0]) == 3`
- `len(L[1]) == 4`
- `L[0]      ==`
- `L[1][-1]  ==`

Lista de listas

```
L = [   [ 1, 5,  2  ],
        [ 0, 0, 7,  1 ],
        [ 3, 2,  1  ]  ]
```

- El código anterior define una lista de listas

¿Cuánto vale?

- `len(L)    == 3`
- `len(L[0]) == 3`
- `len(L[1]) == 4`
- `L[0]      == [1, 5, 2]`
- `L[1][-1]  ==`

Lista de listas

```
L = [   [ 1, 5,  2  ],
        [ 0, 0, 7,  1 ],
        [ 3, 2,  1  ]  ]
```

- El código anterior define una lista de listas

¿Cuánto vale?

- `len(L)    == 3`
- `len(L[0]) == 3`
- `len(L[1]) == 4`
- `L[0]      == [1, 5, 2]`
- `L[1][-1]  == 1`

Cambiando 7 por 9

```
L = [   [ 1, 5,  2  ],
        [ 0, 0, 7,  1 ],
        [ 3, 2,  1  ]  ]

L
```

- Consideremos la lista apuntada por la variable *L* (a la izquierda)
- ¿Cómo cambiamos ese 7 por un 9?

PONTIFICIA UNIVERSIDAD CATÓLICA DE CHILE

Cambiando 7 por 9

```
L = [   [ 1, 5,  2  ],
        [ 0, 0, 7,  1 ],
        [ 3, 2,  1  ]  ]


L
```

```
L = [   [ 1, 5,  2  ],
        [ 0, 0, 7,  1 ],
        [ 3, 2,  1  ]  ]


L[1]
```

PONTIFICIA UNIVERSIDAD CATÓLICA DE CHILE

Cambiando 7 por 9

```
L = [   [ 1, 5,  2  ],
        [ 0, 0, 7,  1 ],
        [ 3, 2,  1  ]  ]


L
```

```
L = [   [ 1, 5,  2  ],
        [ 0, 0, 7,  1 ],
        [ 3, 2,  1  ]  ]


L[1]
```

```
L = [   [ 1, 5,  2  ],
        [ 0, 0, 7,  1 ],
        [ 3, 2,  1  ]  ]


L[1][2]
```

PONTIFICIA UNIVERSIDAD CATÓLICA DE CHILE

Cambiando 7 por 9

```
L = [   [ 1, 5,  2  ],
        [ 0, 0, 7,  1 ],
        [ 3, 2,  1  ]  ]


L
```

```
L = [   [ 1, 5,  2  ],
        [ 0, 0, 7,  1 ],
        [ 3, 2,  1  ]  ]


L[1]
```

```
L = [   [ 1, 5,  2  ],
        [ 0, 0, 7,  1 ],
        [ 3, 2,  1  ]  ]


L[1][2]
```

```
L = [   [ 1, 5,  2  ],
        [ 0, 0, 9,  1 ],
        [ 3, 2,  1  ]  ]


L[1][2] = 9
```

Sea:

```
L = [ [1,2,3], [4,5,6], [7,8,9] ]
L[1] = L[2]
print(L)
```

¿Qué imprime este código? (*aprox*)

Sea:

```
L = [ [1,2,3], [4,5,6], [7,8,9] ]
L[1] = L[2]
print(L)
```

¿Qué imprime este código? (*aprox*)

(a) [[1,2,3], [1,2,3], [1,2,3]]
(b) [[4,5,6], [4,5,6], [7,8,9]]
(c) [[1,2,3], [4,5,6], [7,8,9]]
(d) [[1,2,3], [4,5,6], [4,5,6]]
(e) [[1,2,3], [7,8,9], [7,8,9]]

Sea:

```
L = [ [1,2,3], [4,5,6], [7,8,9] ]
L[1] = L[2]
print(L)
```

¿Qué imprime este código? (*aprox*)

(a) [[1,2,3], [1,2,3], [1,2,3]]
(b) [[4,5,6], [4,5,6], [7,8,9]]
(c) [[1,2,3], [4,5,6], [7,8,9]]
(d) [[1,2,3], [4,5,6], [4,5,6]]
(e) [[1,2,3], [7,8,9], [7,8,9]]

Sea:

```
L = [ [1,2,3], [4,5,6], [7,8,9] ]
L.append( L.pop(0) )
print(L)
```

¿Qué imprime este código? (*aprox*)

Sea:

```
L = [ [1,2,3], [4,5,6], [7,8,9] ]
L.append( L.pop(0) )
print(L)
```

¿Qué imprime este código? (*aprox*)

(a) [[4,5,6], [1,2,3], [1,2,3]]
(b) [[1,2,3], [7,8,9], [7,8,9]]
(c) [[4,5,6], [1,2,3], [7,8,9]]
(d) [[4,5,6], [7,8,9], [1,2,3]]
(e) [[4,5,6], [7,8,9]]

Sea:

```
L = [ [1,2,3], [4,5,6], [7,8,9] ]
L.append( L.pop(0) )
print(L)
```

¿Qué imprime este código? (*aprox*)

(a) [[4,5,6], [1,2,3], [1,2,3]]
(b) [[1,2,3], [7,8,9], [7,8,9]]
(c) [[4,5,6], [1,2,3], [7,8,9]]
(d) [[4,5,6], [7,8,9], [1,2,3]]
(e) [[4,5,6], [7,8,9]]

Sea:

```
L = [ list(), list() ]
for k in range(10):
  if k%2 == 0:
    L[0].append(k)
  else:
    L[1].append(k)
print(L)
```

¿Qué imprime este código? (*aprox*)

Sea:

```
L = [ list(), list() ]
for k in range(10):
  if k%2 == 0:
    L[0].append(k)
  else:
    L[1].append(k)
print(L)
```

¿Qué imprime este código? (*aprox*)

(a) [[0,2,4,6,8], [1,3,5,7,9]]
(b) [[1,2,3,4,5], [6,7,8,9,10]]
(c) [[1,3,5,7,9], [0,2,4,6,8]]
(d) [[1,3,5,7,9], [1,3,5,7,9]]
(e) [5, 5]

Sea:

```
L = [ list(), list() ]
for k in range(10):
  if k%2 == 0:
    L[0].append(k)
  else:
    L[1].append(k)
print(L)
```

¿Qué imprime este código? (*aprox*)

(a) [[0,2,4,6,8], [1,3,5,7,9]]
(b) [[1,2,3,4,5], [6,7,8,9,10]]
(c) [[1,3,5,7,9], [0,2,4,6,8]]
(d) [[1,3,5,7,9], [1,3,5,7,9]]
(e) [5, 5]

PONTIFICIA UNIVERSIDAD CATÓLICA DE CHILE

Representaciones típicas

- Objetivos:
  - Conocer cómo las listas de listas pueden representar algunas objetos o ideas
  - Entender cómo implementar soluciones
  - Dar una dimensión ampliamente aplicable a listas de listas

PONTIFICIA UNIVERSIDAD CATÓLICA DE CHILE

Representado planillas de cálculo

- Un caso típico de uso de *listas de listas* es en el procesamiento de datos en tablas o planillas
- Los datos de la planilla de la derecha vienen en la forma

  L = [...,[sexo,edad,peso,altura],...]

  (ver siguiente lámina)
- Los datos representados vienen de una muestra *real* de estadísticas biométricas sencillas

PLANILLA

Sexo	Edad	Peso	Altura
Mujer	24	68	156
Hombre	35	75	170
Mujer	26	62	175
Hombre	62	61	169
Mujer	50	93	180
Hombre	31	67	171
Mujer	44	79	182
Hombre	30	71	159
Mujer	41	69	160
Mujer	51	68	158
Mujer	26	72	169

PONTIFICIA UNIVERSIDAD CATÓLICA DE CHILE

Representado planillas de cálculo

- Como *lista de listas*:

```
planilla = [
   ['Sexo', 'Edad', 'Peso', 'Altura'],
   ['Mujer', 24, 68, 156],
   ['Hombre', 35, 75, 170],
   ['Mujer', 26, 62, 175],
   ['Hombre', 62, 61, 169],
   ['Mujer', 50, 93, 180],
   ['Hombre', 31, 67, 171],
   ['Mujer', 44, 79, 182],
   ['Hombre', 30, 71, 159],
   ['Mujer', 41, 69, 160],
   ['Mujer', 51, 68, 158],
   ['Mujer', 26, 72, 169]  ]
```

PLANILLA

Sexo	Edad	Peso	Altura
Mujer	24	68	156
Hombre	35	75	170
Mujer	26	62	175
Hombre	62	61	169
Mujer	50	93	180
Hombre	31	67	171
Mujer	44	79	182
Hombre	30	71	159
Mujer	41	69	160
Mujer	51	68	158
Mujer	26	72	169

PONTIFICIA UNIVERSIDAD CATÓLICA DE CHILE

Representado planillas de cálculo

- Aquí va la *lista de listas* completa:

```
planilla = [['Sexo', 'Edad', 'Peso', 'Altura'], ['Mujer', 24, 68, 156],
 ['Hombre', 35, 75, 170], ['Mujer', 26, 62, 175],  ['Hombre', 62, 61, 169],
 ['Mujer', 50, 93, 180],  ['Hombre', 31, 67, 171], ['Mujer', 44, 79, 182],
 ['Hombre', 30, 71, 159], ['Mujer', 41, 69, 160],  ['Mujer', 51, 68, 158],
 ['Mujer', 26, 72, 169],  ['Hombre', 23, 73, 178], ['Mujer', 28, 56, 168],
 ['Mujer', 25, 65, 159],  ['Mujer', 30, 82, 166],  ['Mujer', 44, 71, 164],
 ['Hombre', 33, 66, 162], ['Mujer', 35, 79, 188],  ['Mujer', 46, 76, 187],
 ['Hombre', 32, 89, 173], ['Mujer', 43, 81, 160],  ['Hombre', 47, 77, 181],
 ['Mujer', 37, 73, 178],  ['Hombre', 45, 71, 164], ['Mujer', 49, 72, 181],
 ['Hombre', 30, 64, 184], ['Mujer', 32, 70, 162],  ['Mujer', 40, 23, 168],
 ['Hombre', 41, 81, 187], ['Mujer', 38, 82, 177]]
```

PONTIFICIA UNIVERSIDAD CATÓLICA DE CHILE

Determinar usando Python

- Con los datos anteriores, responda las siguientes preguntas →

- ¿Cuántas mujeres hay en la planilla?
- ¿Cuál es la edad promedio en los datos?
- ¿Cuántos hombres pesan más de 80 kg?
- ¿Cuál es la edad de la persona con menor índice de masa corporal (IMC) en los datos? IMC = peso/altura^2

PONTIFICIA UNIVERSIDAD CATÓLICA DE CHILE

Representando redes y relaciones

Ejemplos de redes:

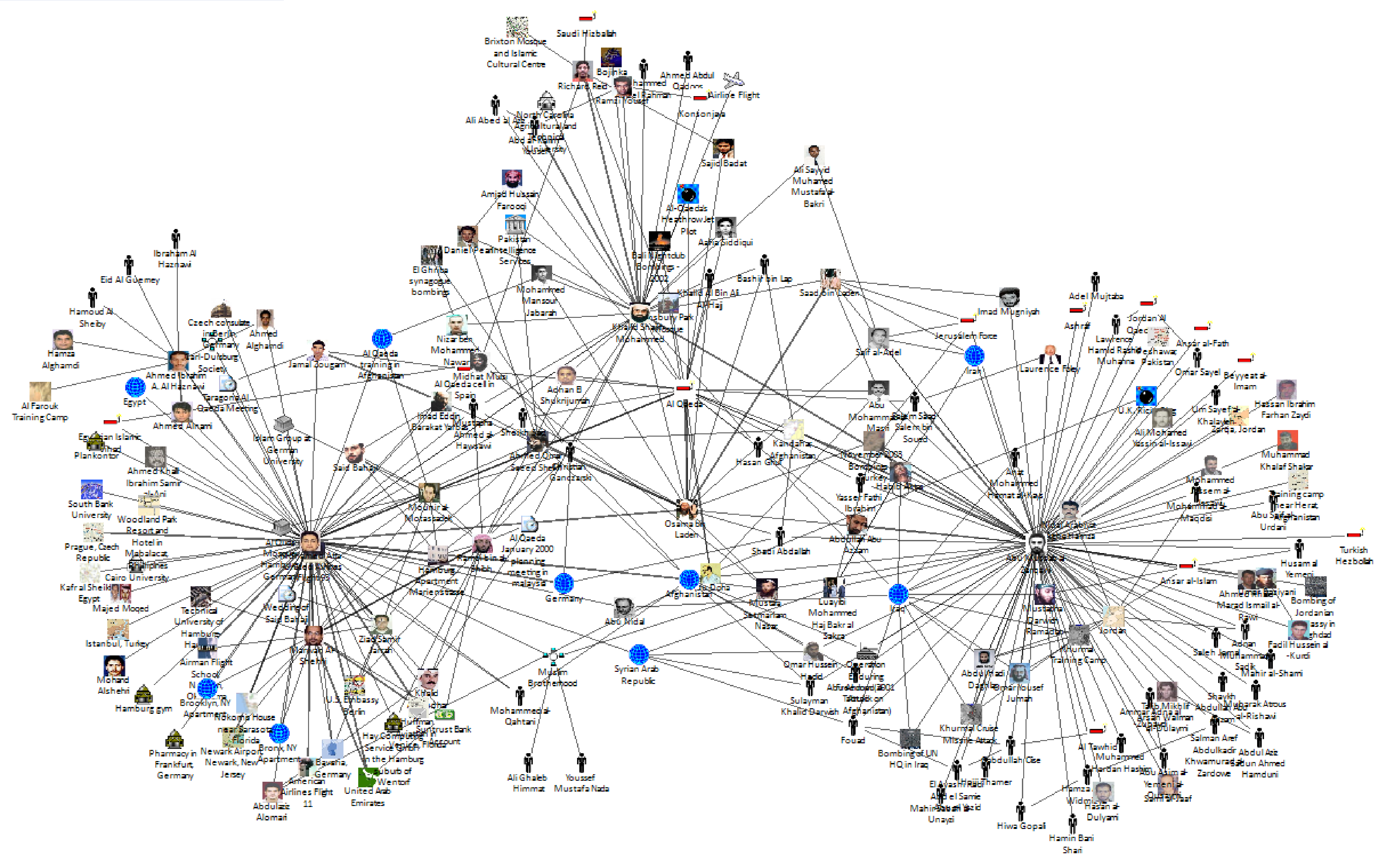
- Sociogramas o redes sociales;
- Cadenas tróficas;
- Redes de infrastructura, ej, eléctricas, viales, de agua, etc;
- Filogenia;
- Organigramas; etc.

- Hay varias opciones para representar relaciones
- Una forma es mediante díadas o pares así:

```
L = [ ['Daniela','Javier'],
      ['Daniela','Isidora'],
      ['Isidora','Mario'],
      ['Mario','Luigi'] ]
```

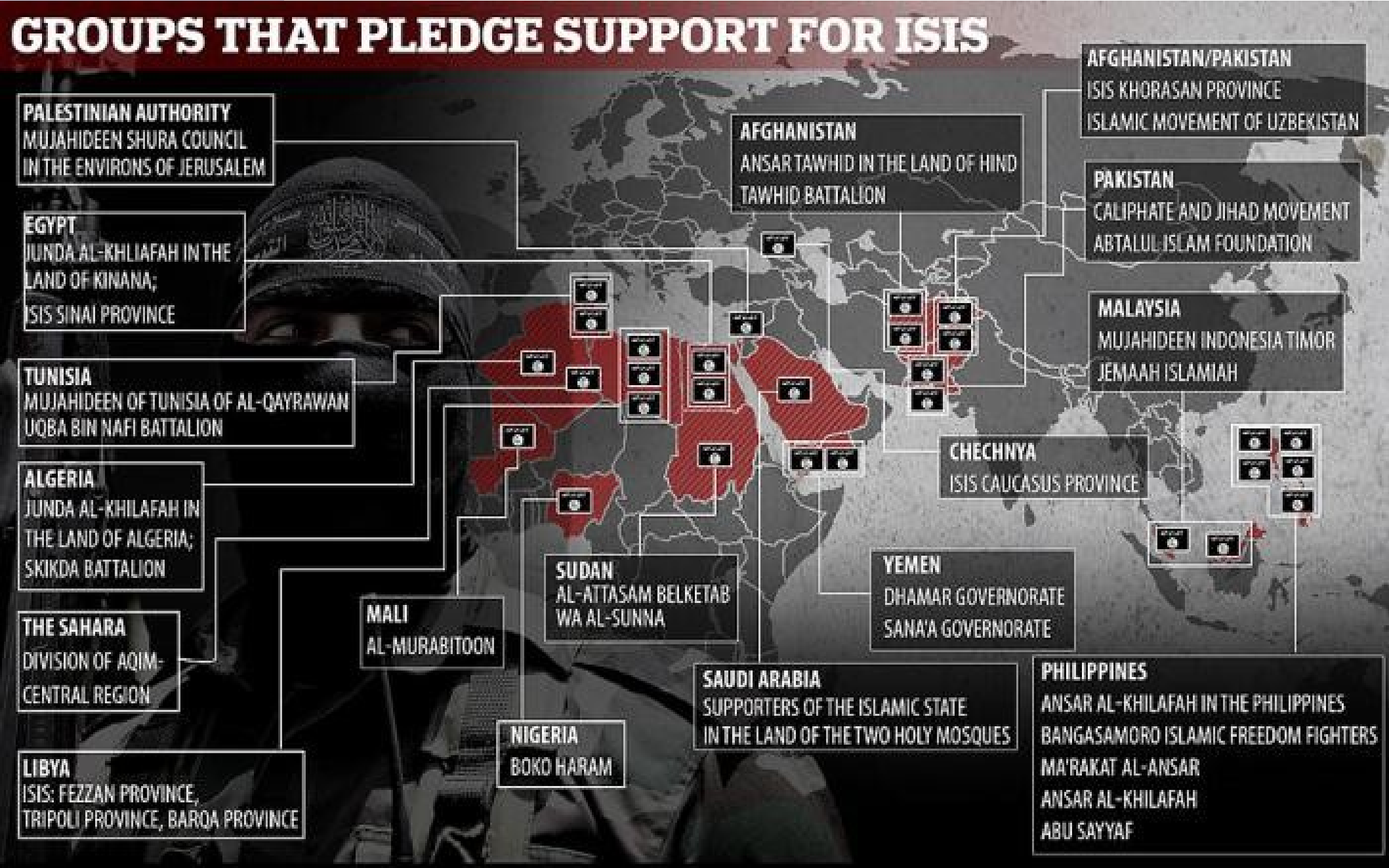
- Otra forma es listando toda la vecindad de cada persona:

```
L = [ ['Daniela',['Javier','Isidora']],
      ['Javier',['Isidora']]
      ['Isidora',['Daniela','Mario']],
      ['Mario',['Isidora','Luigi']],
      ['Luigi',['Mario']] ]
```



Saudi Hizballah
Brixton Mosque and Islamic Cultural Centre
Richard Reid
Bojinka
Airline Flight
Konsonjaya
Sajid Badat
Ali Sayyid Muhamed Mustafa al-Bakri
Al-Qaeda's Heathrow Jet Plot
Amjad Hussain Farooqi
Aafia Siddiqui
Daniel Pearl
El Ghriba synagogue bombing
Mohammed Mansour Jabarah
Bashir bin Lap
Saad bin Laden
Imad Mugniyah
Adel Mujtaba
Jerusalem Force
Saif al-Adel
Iran
Laurence Foley
Ashraf
Ansar al-Fath
Omar Sayel
Hassan Ibrahim Farhan Zaydi
Muhammad Khalaf Shakar
Turkish Hezbollah
Ansar al-Islam
Mahir al-Shami
Al Tawhid
Khurmal Training Camp
Fouad
Bombing of UN HQ in Iraq
Hiwa Gopal
Hamin Bari Shari
Iraq
Jordan
Syrian Arab Republic
Muslim Brotherhood
Mohammed al-Qahtani
Ali Ghaleb Himmat
Youssef Mustafa Nada
Germany
Afghanistan
Abu Nidal
Osama bin Laden
Al Qaeda
Shadi Abdallah
Hasan Gul
Yasser Fathi Ibrahim
Abdullah Abu Azzam
Adnan G. Shukrijumah
Christian Ganczarski
Said Bahaji
Mounir al Motassadeq
Al Qaeda training in Afghanistan
Al Qaeda cell in Spain
Jamal Zougam
Ahmed Alghamdi
Ibraham Al Hazmawi
Eid Al Guemey
Hamoud Al Sheiby
Hamza Alghamdi
Al Farouk Training Camp
Egypt
Ahmed Alnami
Islam Group at German University
South Bank University
Prague, Czech Republic
Cairo University
Majed Moqed
Istanbul, Turkey
Airman Flight School
Mohand Alshehri
Hamburg gym
Pharmacy in Frankfurt, Germany
Newark Airport, Newark, New Jersey
Abdulaziz Alomari
United Arab Emirates
Bavaria, Germany
U.S. Embassy Berlin
Ziad Samir Jarrah

GROUPS THAT PLEDGE SUPPORT FOR ISIS



Detecting Key Players in Terrorist Networks

Ala Berzinji
University of Sulaimani
Sulaimani, Iraq
Email: ala.berzinji@gmail.com

Lisa Kaati
FOI
Stockholm, Sweden
Email: lisa.kaati@foi.se

Ahmed Rezine
Linköpings University
Linköping, Sweden
email: ahmed.rezine@liu.se

***Abstract*—The interest in analyzing loosely connected and decentralized terrorist networks of global reach has grown during the past decade. Social Network Analysis (SNA) is one approach towards understanding terrorist networks since it can be used to analyze the structure of a network and to detect important persons and links.**

In this work we study decentralized terrorist networks with different types of nodes. The nodes can be either organizations, places or persons. We use a combination of different centrality measures to detect key players in such networks.

I. Introduction

Criminal and terrorist social networks can often be divided into hierarchies with actor roles varying from ordinary operatives to the finance manager and the leader. The most important roles include the leader who acts as the guide and mentor for active and that acts as gateway in the network. The finance manager is detected using a combination of different well-known centrality measures.

Terrorist social networks are also known as dark networks. Recent studies such as [2] have explored the use of SNA methods to analyze dark networks mostly focusing on assigning roles to actors in the network. In [3] the use of centrality measures to identify key actors in criminal networks is explored and in [4] centrality measures are used to identify the group leader of the September 11th hijackers. However, none of these operates on networks with different categories of nodes.

In [5] SNA is used to determine the leader and gatekeeper role for individual nodes in addition to hierarchical clustering methods to identify subgroups within criminal networks. In [6]

PONTIFICIA UNIVERSIDAD CATÓLICA DE CHILE

Determinar usando Python

Sea *L* como sigue:

```
L = [ ['Daniela','Javier'],
      ['Daniela','Isidora'],
      ['Isidora','Mario'],
      ['Mario','Luigi'],
      ['Mario','Koopa'],
      ['Bario','Mario'],
      ['Daniela','Arturo'],
      ['Ana','Arturo'] ],
```

Implemente →

- Defina la función **`actores(L)`** que retorne la lista de los nombres de los actores de las relaciones en ***L***, sin que se repitan o falten nombres
- Defina la función **`vecinos(nom,L)`** que retorne la lista de los contactos asociados al actor de nombre ***nom***
- Defina la función **`vecinos2(nom,L)`** que retorne la lista de nombres de los contactos de los contactos (sin repetir nombres) asociados al actor de nombre ***nom***

PONTIFICIA UNIVERSIDAD CATÓLICA DE CHILE

Representando matrices

$$\begin{bmatrix} 2 & 4 & 6 \\ 6 & 4 & 2 \end{bmatrix}$$

- Podemos representar una matriz como una lista de *filas*
- La matriz a la izquierda se puede representar como:

```
L = [ [2, 4, 6], [6, 4, 2] ]
```

- El elemento de la fila *i* y columna *j* es entonces

```
L[i][j]
```

- Si la matriz es *dispersa*, o sea, la gran mayoría de sus elementos son cero, entonces podemos indicar sólo aquellos valores que son nulos (listas de [i, j, val])
- Para cosas más avanzandas, hay módulos especiales como numpy y scipy

Ejercicios de matrices

Considere matrices del tipo:

```
M = [[ m11, m12, ..., m1n ],
     [ m21, m22, ..., m2n ],
       ...          ...
     [ mn1, mn2, ..., mnn ]]
```

Implemente las siguientes funciones:→

- **`suma(A,B)`** que retorna una *nueva* matriz representando A+B
- **`mult(A,B)`** que retorna una *nueva* matriz representando A·B
- **`frobenius(A)`** que retorna la raíz cuadrada de la suma de los cuadrados de los términos de la matriz
- **`maxim(A)`** que retorna el máximo de los valores absolutos de los términos de A

Suma de matrices

```
def suma(A,B):
  resp = []
  for i in range(len(A)):
    fila = []
    for j in range(len(A[i])):
      fila.append( A[i][j] + B[i][j] )
    resp.append(fila)
  return resp

A = [ [1,0,0], [0,2,0], [0,0,3] ]
B = [ [1,1,1], [0,1,0], [2,0,2] ]
print( "A:  ", A)
print( "B:  ", B)
print( "A+B:", suma(A,B) )
```

- Solución a la izquierda (testcase para verificar incluido)
- Output:

```
A:   [[1, 0, 0], [0, 2, 0], [0, 0, 3]]
B:   [[1, 1, 1], [0, 1, 0], [2, 0, 2]]
A+B: [[2, 1, 1], [0, 3, 0], [2, 0, 5]]
```

$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 3 \end{pmatrix} + \begin{pmatrix} 1 & 1 & 1 \\ 0 & 1 & 0 \\ 2 & 0 & 2 \end{pmatrix} = \begin{pmatrix} 2 & 1 & 1 \\ 0 & 3 & 0 \\ 2 & 0 & 5 \end{pmatrix}$$

Multiplicación de matrices

```
def mult(A,B):
  resp = []
  for i in range(len(A)):
    fila = []
    for j in range(len(A[i])):
      coef = 0
      for k in range(len(A[i])):
        coef += A[i][k] * B[k][j]
      fila.append(coef)
    resp.append(fila)
  return resp

A = [ [1,0,0], [0,2,0], [0,0,3] ]
B = [ [1,1,1], [0,1,0], [2,0,2] ]
print( "A:  ", A)
print( "B:  ", B)
print( "A*B:", mult(A,B) )
```

- Solución a la izquierda (testcase para verificar incluido)
- Output:

```
A:   [[1, 0, 0], [0, 2, 0], [0, 0, 3]]
B:   [[1, 1, 1], [0, 1, 0], [2, 0, 2]]
A*B: [[1, 1, 1], [0, 2, 0], [6, 0, 6]]
```

$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 3 \end{pmatrix} \times \begin{pmatrix} 1 & 1 & 1 \\ 0 & 1 & 0 \\ 2 & 0 & 2 \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 \\ 0 & 2 & 0 \\ 6 & 0 & 6 \end{pmatrix}$$