PONTIFICIA
UNIVERSIDAD
CATÓLICA
DE CHILE

INTRODUCCIÓN A LA PROGRAMACIÓN

Objetivos

- Preguntas rápidas de repaso
- Completar el tema de listas de listas
- Problemas tipo solemne

Listas de Listas

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]

Sea:

```
L = list()
for i in range(4):
    L.append( "Z"*i )
print(L)
```

¿Qué imprime este código? (*aprox*)

Sea:

```
L = list()
for i in range(4):
    L.append( "Z"*i )
print(L)
```

¿Qué imprime este código? (*aprox*)

(a) ['Z', 'ZZ']
(b) ['', 'Z', 'ZZ', 'ZZZ']
(c) ['Z', 'ZZ', 'ZZZ', 'ZZZZ']
(d) ['', 'Z', 'ZZ', 'ZZZ', 'ZZZZ']
(e) ['Z', 'ZZ', 'ZZZ']

Sea:

```
L = list()
for i in range(4):
    L.append( "Z"*i )
print(L)
```

¿Qué imprime este código? (*aprox*)

(a) ['Z', 'ZZ']
(b) ['', 'Z', 'ZZ', 'ZZZ']
(c) ['Z', 'ZZ', 'ZZZ', 'ZZZZ']
(d) ['', 'Z', 'ZZ', 'ZZZ', 'ZZZZ']
(e) ['Z', 'ZZ', 'ZZZ']

Sea:

```
L = [ [20], [10,1,0] ]
L.append( L[0][0] )
L.append( L[-1] )
print(L)
```

¿Qué imprime este código? (*aprox*)

Sea:

```
L = [ [20], [10,1,0] ]
L.append( L[0][0] )
L.append( L[-1] )
print(L)
```

¿Qué imprime este código? (*aprox*)

(a) []
(b) [20, 10, 1, 0]
(c) [[20], [10, 1, 0], [20]]
(d) [[20], [10, 1, 0], 20, 20]
(e) [[20], [10, 1, 0], [10, 1, 0]]

Sea:

```
L = [ [20], [10,1,0] ]
L.append( L[0][0] )
L.append( L[-1] )
print(L)
```

¿Qué imprime este código? (*aprox*)

(a) []
(b) [20, 10, 1, 0]
(c) [[20], [10, 1, 0], [20]]
(d) [[20], [10, 1, 0], 20, 20]
(e) [[20], [10, 1, 0], [10, 1, 0]]

Sea:

```
L = [[0,1,0], [100], [50,0]]
w = 0
for fila in L:
    for x in fila:
        w += x
print(w)
```

¿Qué imprime este código? (*aprox*)

Sea:

```
L = [[0,1,0], [100], [50,0]]
w = 0
for fila in L:
    for x in fila:
        w += x
print(w)
```

¿Qué imprime este código? (*aprox*)

(a) 151
(b) 1
(c) 7
(d) 100
(e) 50

Sea:

```
L = [[0,1,0], [100], [50,0]]
w = 0
for fila in L:
    for x in fila:
        w += x
print(w)
```

¿Qué imprime este código? (*aprox*)

(a) 151
(b) 1
(c) 7
(d) 100
(e) 50

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

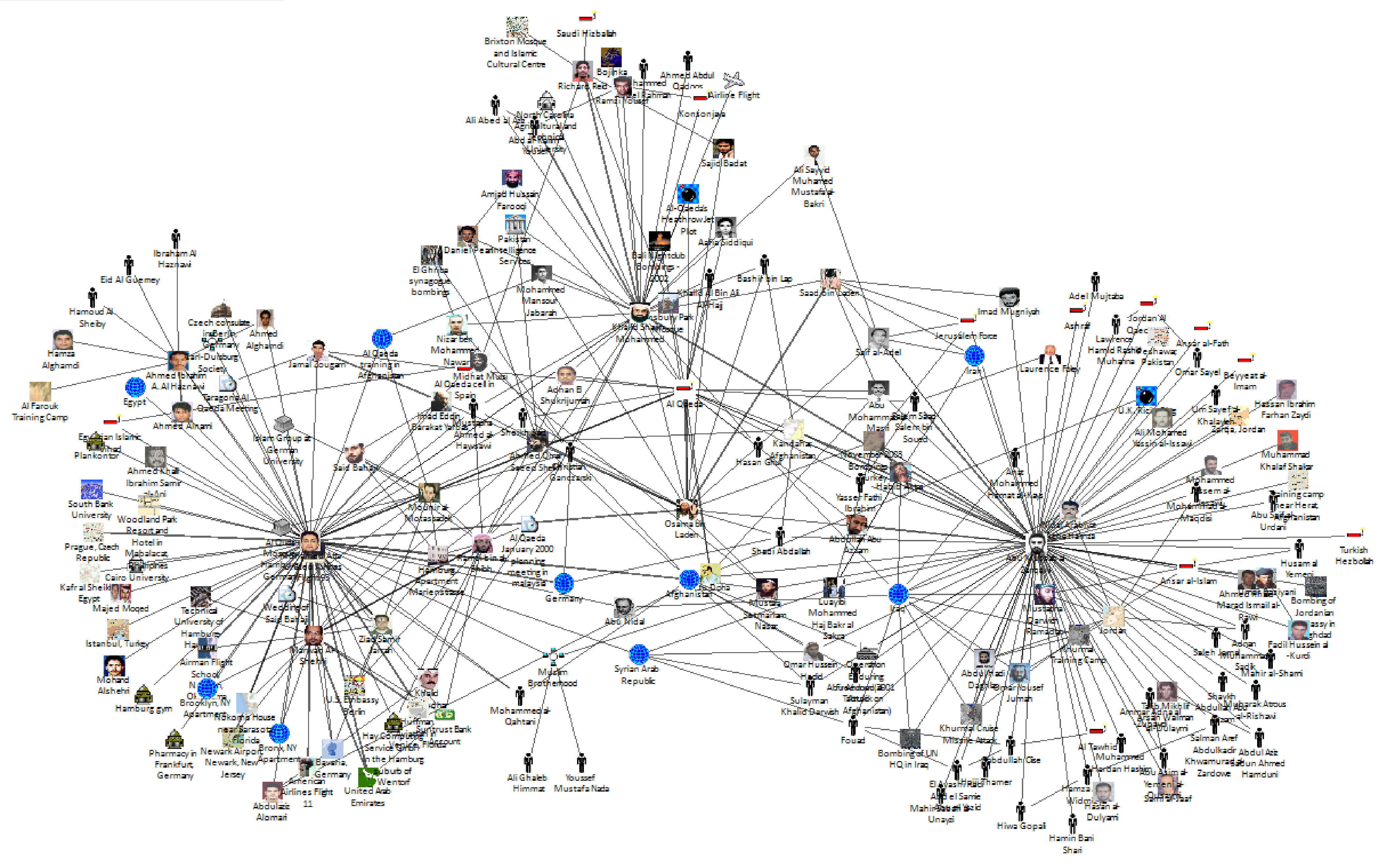
Ejemplos de redes:
- Sociogramas o redes sociales;
- Cadenas tróficas;
- Redes de infrastructura, ej, eléctricas, viales, de agua, etc;
- Filogenia;
- Organigramas; etc.

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

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

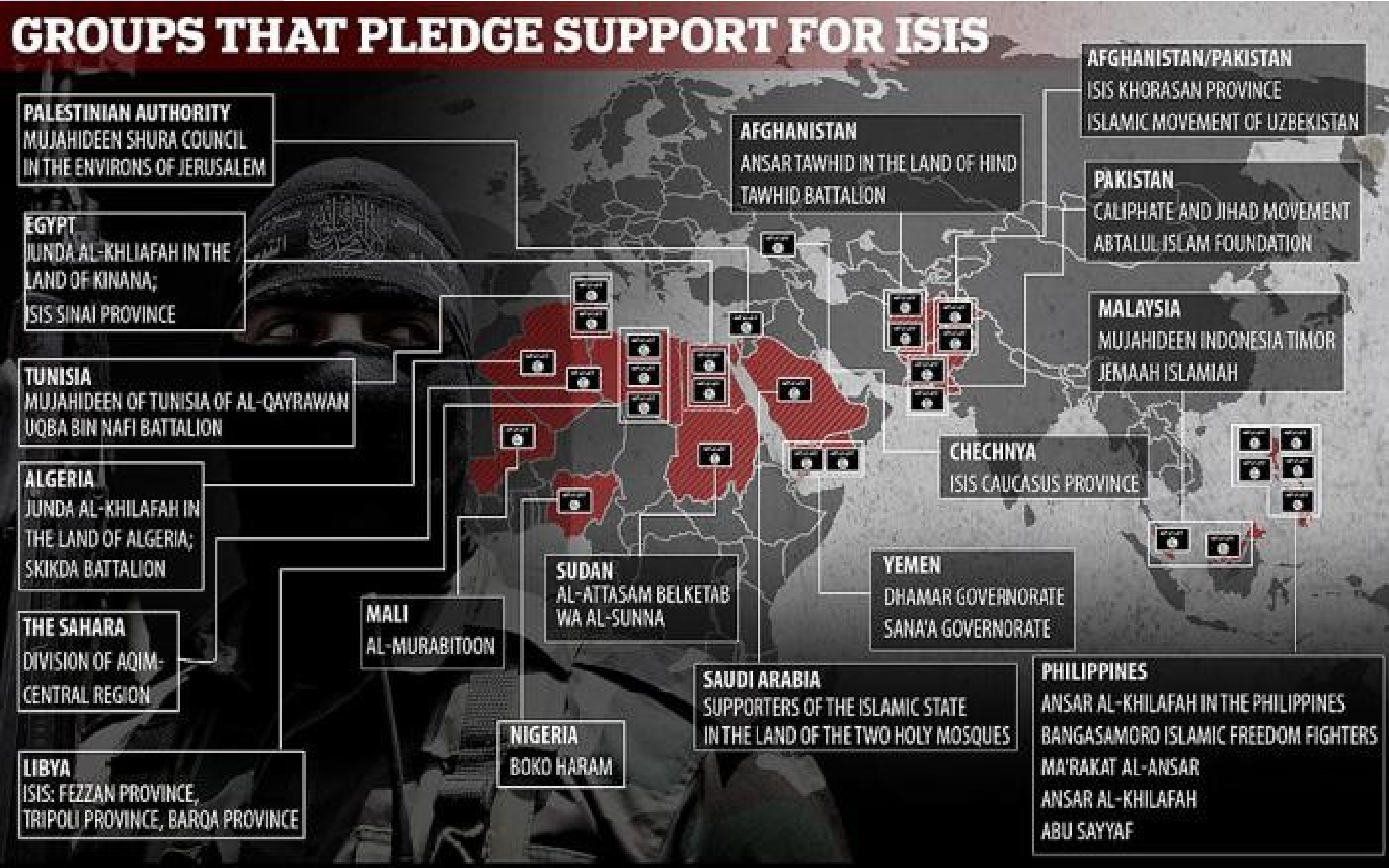
- Otra forma es listando toda la vecindad de cada persona:

```
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
Ahmed Abdul Qadoos
Airline Flight
Ramzi Yousef
Konsonjaya
Ali Abed
Sajid Badat
Ali Sayyid Muhamed Mustafa al-Bakri
Amjad Hussain Farooqi
Al-Qaeda's Heathrow Jet Plot
Aafia Siddiqui
Daniel Pearl
El Ghriba synagogue bombings
Mohammed Mansour Jabarah
Bali Nightclub Bombings - 2002
Bashir bin Lap
Saad bin Laden
Imad Mugniyah
Adel Mujtaba
Ashraf
Jordan Al Qaeda
Ansar al-Fath
Lawrence Foley
Hamid Rashid Muhanna
Peshawar, Pakistan
Omar Sayel
Beyyeat al-Imam
Hassan Ibrahim Farhan Zaydi
Ibraham Al Hazmawi
Eid Al Guemey
Hamoud Al Sheiby
Hamza Alghamdi
Czech consulate in Berlin
Ahmed Alghamdi
Jamal Zougam
Al Qaeda training in Afghanistan
Nizar ben Mohammed Nawar
Midhat Mursi
Al Qaeda cell in Spain
Khalid Sheikh Mohammed
Jerusalem Force
Saif al-Adel
Iran
Laurence Foley
Al Farouk Training Camp
Ahmed Ibrahim A. Al Haznawi
Egypt
Taragona Al Qaeda Meeting
Ahmed Alnami
Adnan G Shukrijumah
Al Qaeda
Abu Mohammed Masri
Salim Saad Salem bin Souid
U.K.
Ali Mohamed Yassin al-Issawi
Khalayleh
Zarqa, Jordan
Muhammad Khalaf Shakar
Islam Group at German University
Said Bahaji
Imad Eddin Barakat Yarbas
Ahmed al-Haznawi
Ahmed Omar Saeed Sheikh
Christian Ganczarski
Hasan Ghul
Kandahar, Afghanistan
November 2003 Bombings in Turkey
Habib Akdas
Yasser Fathi Ibrahim
Mohammed Hamid al-Mustafa
Muhammad Jassem al-Ali
Mohammed al-Maqdisi
Training camp near Herat, Afghanistan
Abu Saif al-Urdani
Ahmed Khalil Ibrahim Samir al-Ani
South Bank University
Woodland Park Resort and Hotel in Mabalacat, Philippines
Prague, Czech Republic
Cairo University
Kafr al Sheikh, Egypt
Majed Moqed
Mounir al Motassadeq
Osama bin Laden
Al Qaeda January 2000 planning meeting in Malaysia
Ramzi bin al-Shibh
Hamburg Apartment Marienstrasse
Germany
Afghanistan
Shadi Abdallah
Abdullah Abu Azzam
Turkish Hezbollah
Husam al Yemeni
Ansar al-Islam
Bombing of Jordanian
Technical University of Hamburg
Wedding of Said Bahaji
Ziad Samir Jarrah
Abu Nidal
Mustafa Setmariam Nasar
Luayb Mohammed Haj Bakr al Sakka
Iraq
Mustafa Darwish Ramadan
Jordan
Istanbul, Turkey
Airman Flight School
Marwan Al-Shehhi
Muslim Brotherhood
Syrian Arab Republic
Omar Hussein Hadid
Abdul Hadi Daghlas
Umar Yousef Jumah
Khurmal Training Camp
Fadil Hussein al-Kurdi
Mahir al-Shami
Mohand Alshehri
Hamburg gym
Brooklyn, NY Apartment
Nokomis House near Sarasota, Florida
U.S. Embassy Berlin
Khalid
Suntrust Bank Account
Mohammed al-Qahtani
Sulayman Khalid Darwish
Fouad
Bombing of UN HQ in Iraq
Khurmal Cruise Missile Attack
Abdullah Ose
Shaykh Abdullah
Mubarak Atrous al-Rishawi
Salman Aref Abdulkadir Khwamurad al-Zardowe
Abdul Aziz Saadun Ahmed Hamduni
Pharmacy in Frankfurt, Germany
Newark Airport, Newark, New Jersey
Bronx, NY Apartment
Bavaria, Germany
American Airlines Flight 11
United Arab Emirates
Abdulaziz Alomari
Ali Ghaleb Himmat
Youssef Mustafa Nada
Al Tawhid
Muhammed Hardan Hashim
Hiwa Gopal
Hamin Bani Shari
Hasan al-Dulyami

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]]
```

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

Conclusiones

- Hemos terminado la materia del curso
- Sólo sigue... *¿repasar?*
- Practicar, practicar, practicar