13 – ORDENAMIENTO

Jorge Muñoz

IIC1103 – Introducción a la Programación

ORDENAR LISTA DE INT/STR

¿Puedo ordenar una lista de números?

Sí, con el método *sort* se ordena *ascendentemente*

```
Antes:  [3, 1, 70, -1, 3, 7, 5]
Despues:  [-1, 1, 3, 3, 5, 7, 70]
```

```
#Metodo sort - modifica lista original
p = [3, 1, 70, -1, 3, 7, 5]
print("Antes: ", p)
p.sort()
print("Despues: ", p)
```

```
Antes:  [3, 1, 70, -1, 3, 7, 5]
Despues:  [-1, 1, 3, 3, 5, 7, 70]
```

¿Y una lista de strings?

Sí, sort ordena alfabéticamente

```
Antes:  ['casa', 'auto', 'sillon', 'silla']
Despues:  ['auto', 'casa', 'silla', 'sillon']
```

```
#Metodo sort - modifica lista original
p = ["casa", "auto", "sillon", "silla"]
print("Antes: ", p)
p.sort()
print("Despues: ", p)
```

```
Antes:  ['casa', 'auto', 'sillon', 'silla']
Despues:  ['auto', 'casa', 'silla', 'sillon']
```

… pero recuerda que Python considera mayúsculas y minúsculas diferentes

```
#May y Min son diferentes
p = ["Casa", "caracol", "Barco", "Arbol", "auto"]
print("Antes: ", p)
p.sort()
print("Despues: ", p)
```

```
Antes:  ['Casa', 'caracol', 'Barco', 'Arbol', 'auto']
Despues:  ['Arbol', 'Barco', 'Casa', 'auto', 'caracol']
```

¿¿Eso es todo el tema?? ¿Solo “sort”?
¿Nos puedes enseñar algo más?

Ehh, bueno. Por ejemplo, como ordenar descendentemente

```
Antes:  [3, 1, 70, -1, 3, 7, 5]
Despues:  [70, 7, 5, 3, 3, 1, -1]
```

```
#Ordenar descendentemente con sort y reverse
p = [3, 1, 70, -1, 3, 7, 5]
print("Antes: ", p)
p.sort()
p.reverse()
print("Despues: ", p)
```

también se puede directamente
p.sort(reverse=True)

```
Antes:  [3, 1, 70, -1, 3, 7, 5]
Despues:  [70, 7, 5, 3, 3, 1, -1]
```

¿Y si quiero solo los 3 primeros?

```
Antes:  [13, 10, 5, 70, 8, 7, 3]
Despues:  [3, 5, 7]
```



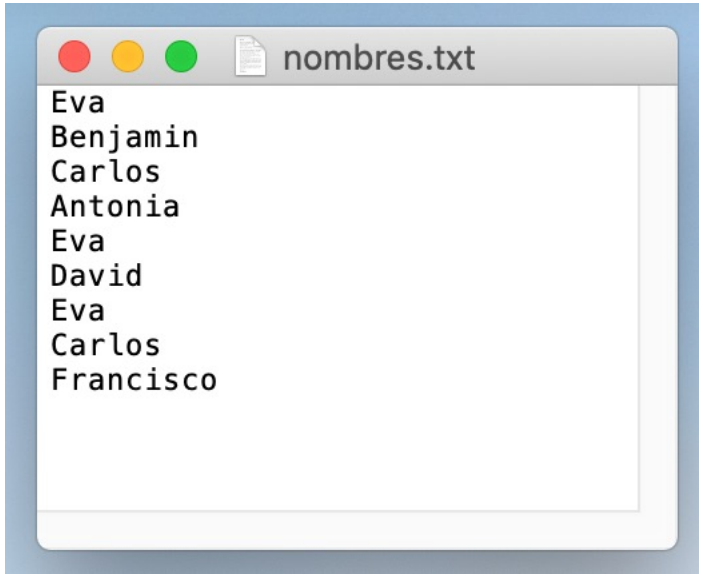
```
#Solo los 3 primeros
p = [13, 10, 5, 70, 8, 7, 3]
print("Antes: ", p)
p.sort()
q = p[0:3] #p es una lista normal y corriente!
print("Despues: ", q)
```

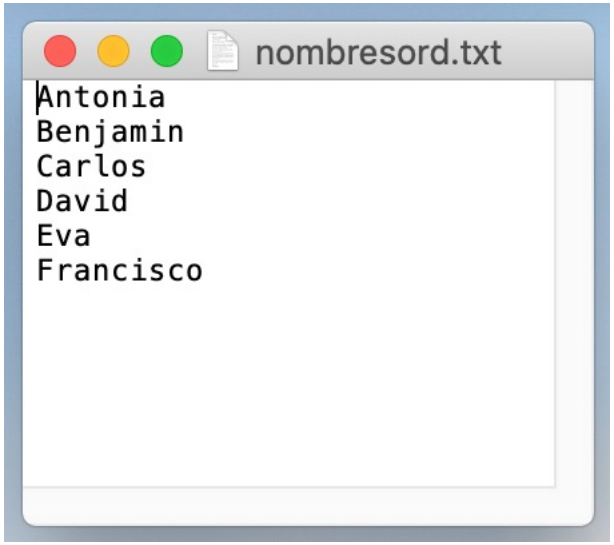
```
Antes:  [13, 10, 5, 70, 8, 7, 3]
Despues:  [3, 5, 7]
```

ORDENADOS Y SIN REPETIDOS

INTERESES

“¿Profe, podría repasar un poco archivos con un ejemplo?”


nombres.txt
Eva
Benjamin
Carlos
Antonia
Eva
David
Eva
Carlos
Francisco


nombresord.txt
Antonia
Benjamin
Carlos
David
Eva
Francisco

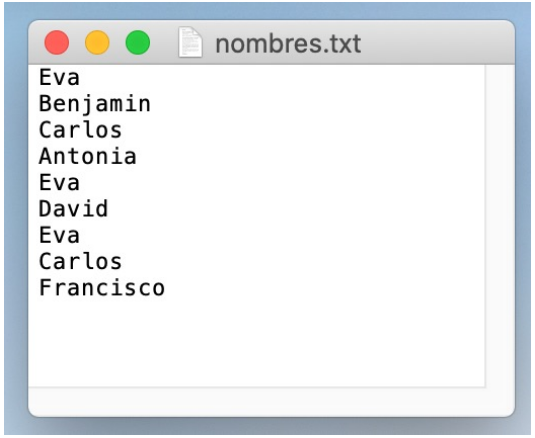
Resolución Problema

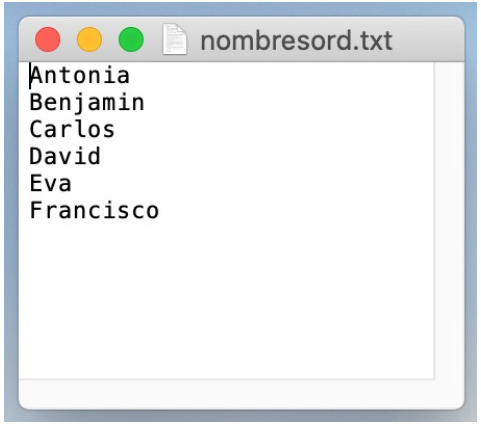
#Leer archivo

#Lista de nombres sin repetidos

#Ordenar

#Escribir archivo



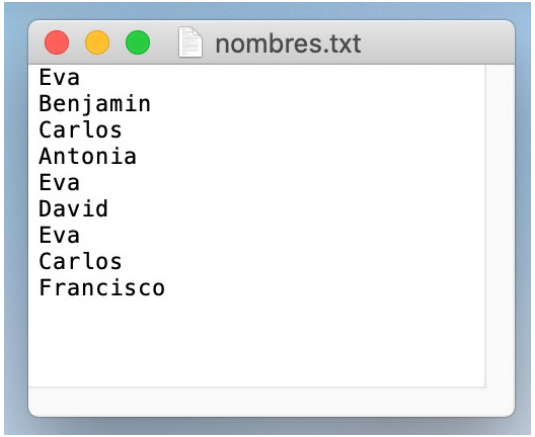


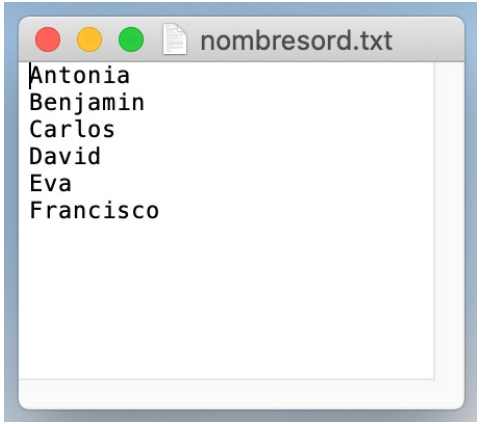
```
#Leer archivo
r = open("nombres.txt")
ls = r.readlines()
r.close()

#Lista de nombres sin repetidos


#Ordenar


#Escribir archivo
```



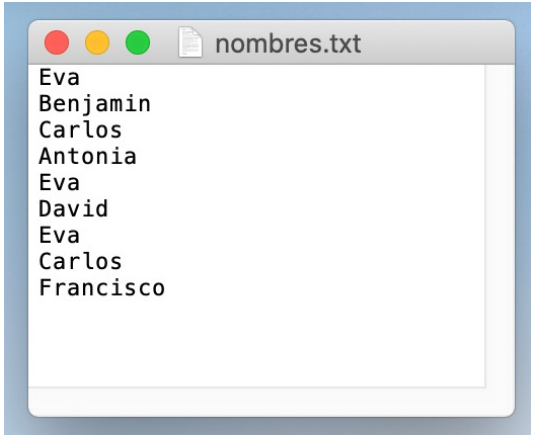


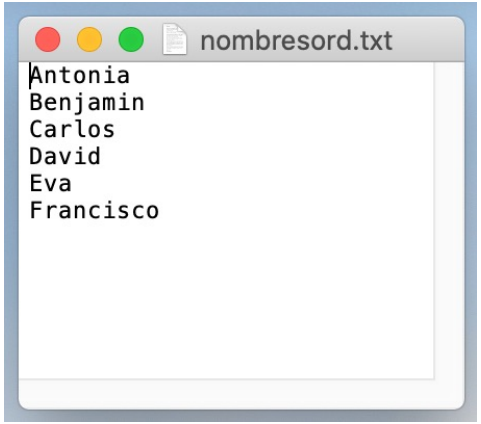
```
#Leer archivo
r = open("nombres.txt")
ls = r.readlines()
r.close()

#Lista de nombres sin repetidos
ns = []
for l in ls:
    n = l.strip()
    if n not in ns:
        ns.append(n)

#Ordenar


#Escribir archivo
```



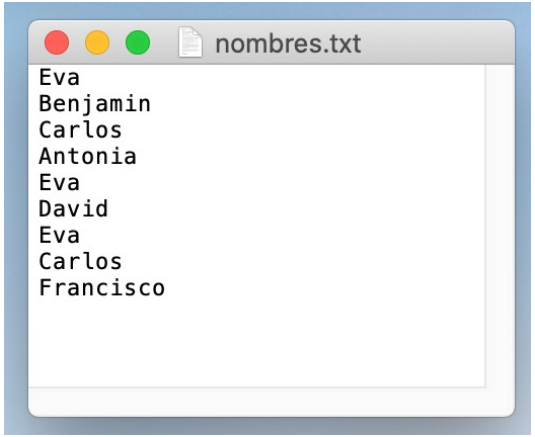


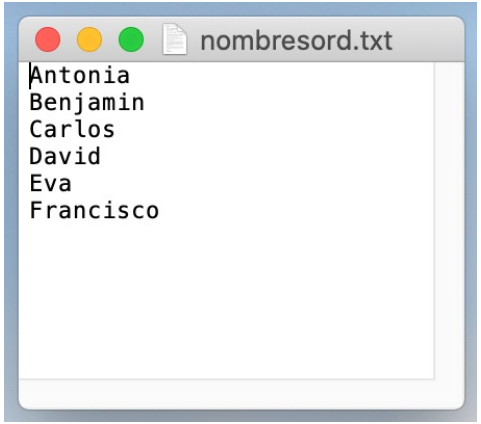
```
#Leer archivo
r = open("nombres.txt")
ls = r.readlines()
r.close()

#Lista de nombres sin repetidos
ns = []
for l in ls:
    n = l.strip()
    if n not in ns:
        ns.append(n)

#Ordenar
ns.sort()

#Escribir archivo
```



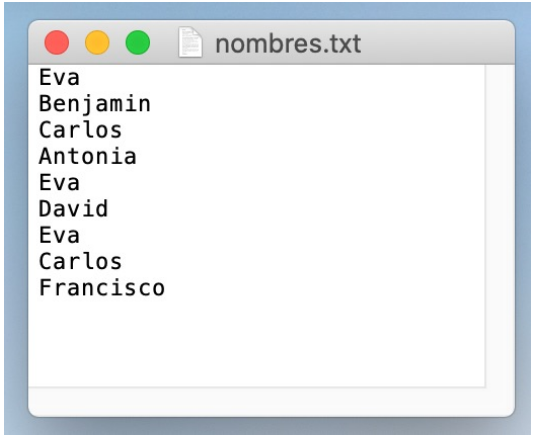


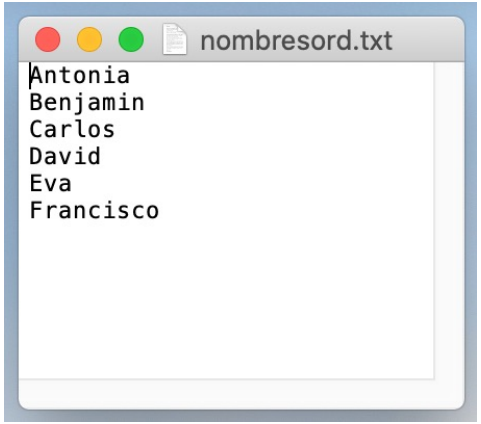
```
#Leer archivo
r = open("nombres.txt")
ls = r.readlines()
r.close()

#Lista de nombres sin repetidos
ns = []
for l in ls:
    n = l.strip()
    if n not in ns:
        ns.append(n)

#Ordenar
ns.sort()

#Escribir archivo
e = open("nombresord.txt","w")
for n in ns:
    print(n,file=e)
e.close()
```





ORDENAR LISTA DE TIPO PROPIO

[5 , 8 , 2]

[“casa” , “auto” , “barco”]

[ ,  , ]

```
class Persona:
    def __init__(self,n,a,e):
        self.nom = n
        self.ape = a
        self.edad = e

    def __str__(self):
        t = self.nom+" "+self.ape+" ("+str(self.edad)+")"
        return t
```

Anna Caceres (22)

```
p1 = Persona("Carlos","Barranco",15)
p2 = Persona("Anna","Caceres",22)
p3 = Persona("Benjamin","Alvarez",38)
q = [p1,p2,p3]
```



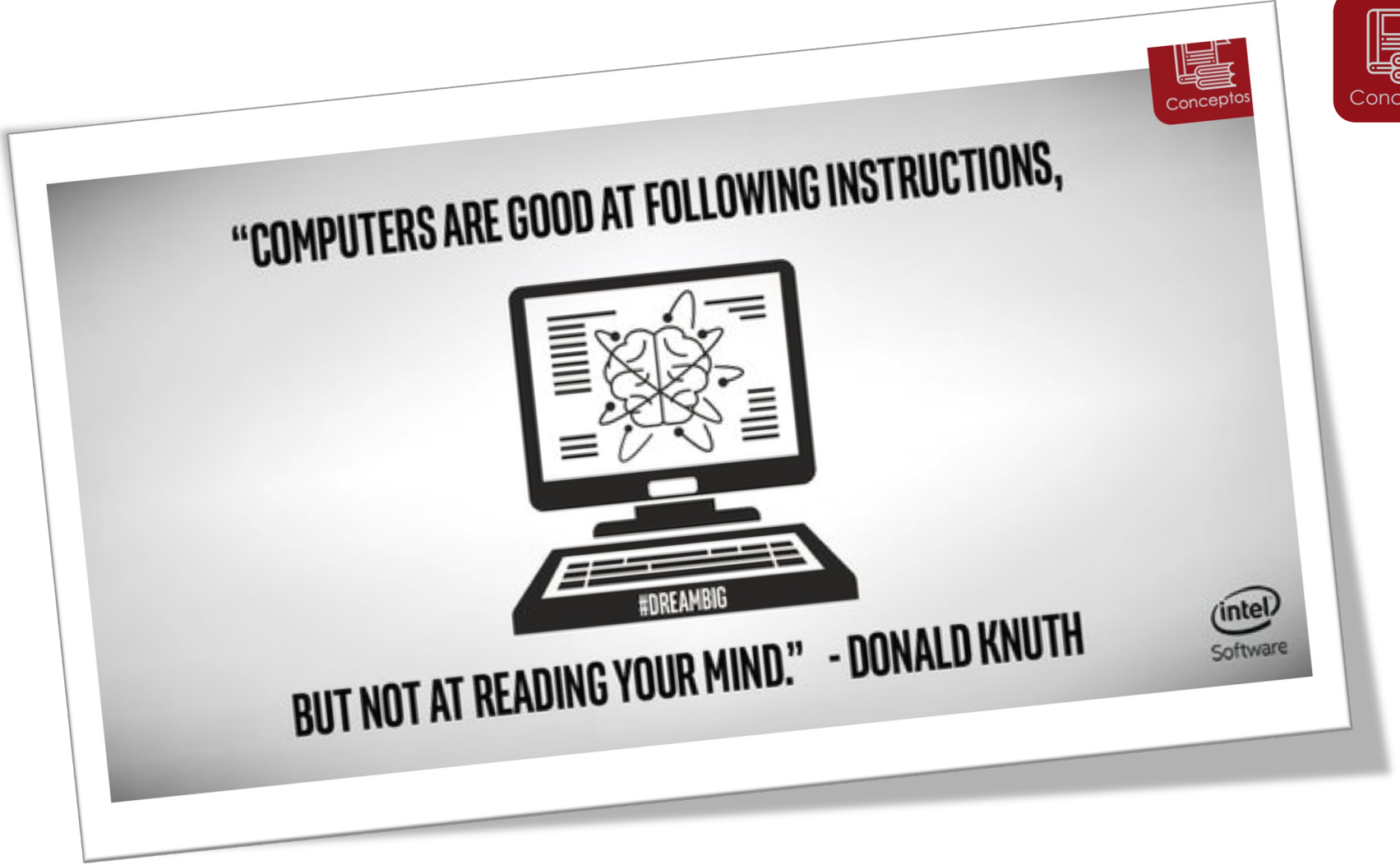
```
q.sort()

print("0->",q[0])
print("1->",q[1])
print("2->",q[2])
```

```
0-> Anna Caceres (22)
1-> Benjamin Alvarez (38)
2-> Carlos Barranco (15)
```

```
0-> Benjamin Alvarez (38)
1-> Carlos Barranco (15)
2-> Anna Caceres (22)
```

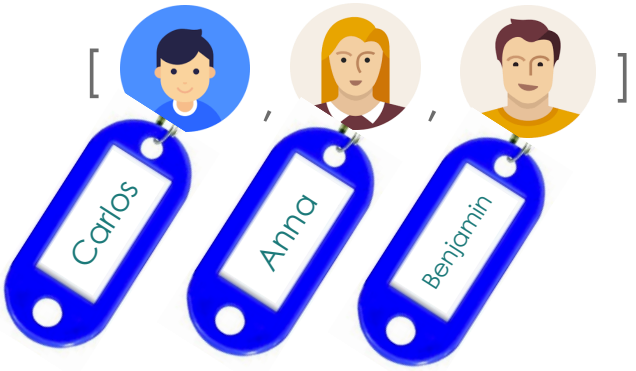
```
0-> Carlos Barranco (15)
1-> Anna Caceres (22)
2-> Benjamin Alvarez (38)
```


Conceptos
"COMPUTERS ARE GOOD AT FOLLOWING INSTRUCTIONS,
#DREAMBIG
BUT NOT AT READING YOUR MIND." - DONALD KNUTH
intel
Software

```
o/codigos/13-Ordenamiento/sortobjetos.py", line 16, in <module>
    q.sort()
TypeError: '<' not supported between instances of 'Persona' and 'Persona'
```

[5 , 8 , 2]

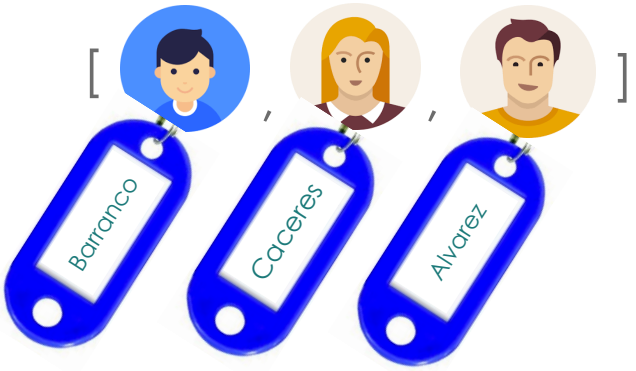
[“casa” , “auto” , “barco”]




```
0-> Anna Caceres (22)
1-> Benjamin Alvarez (38)
2-> Carlos Barranco (15)
```

[5 , 8 , 2]

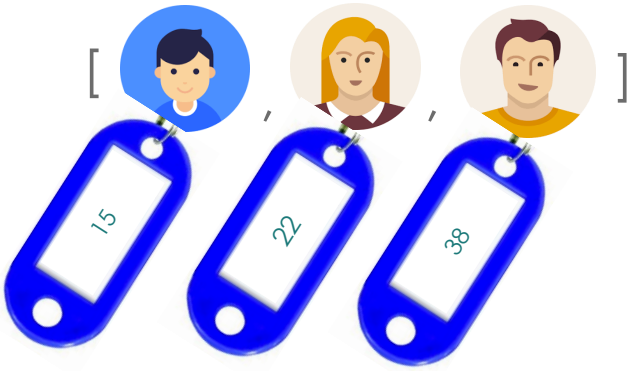
[“casa” , “auto” , “barco”]




```
0-> Benjamin Alvarez (38)
1-> Carlos Barranco (15)
2-> Anna Caceres (22)
```

[5 , 8 , 2]

[“casa” , “auto” , “barco”]




```
0-> Carlos Barranco (15)
1-> Anna Caceres (22)
2-> Benjamin Alvarez (38)
```

```
class Persona:
    def __init__(self,n,a,e):
        self.nom = n
        self.ape = a
        self.edad = e

    def __str__(self):
        t = self.nom+" "+self.ape+" ("+str(self.edad)+")"
        return t


p1 = Persona("Carlos","Barranco",15)
p2 = Persona("Anna","Caceres",22)
p3 = Persona("Benjamin","Alvarez",38)
q = [p1,p2,p3]

#1 - Definir la funcion para key
```

Recibe 1 elemento del tipo Persona y retorna un int o str

```
#2- Ordenar usando la key
```

Indicar la función anterior sin los paréntesis

```
print("0->",q[0])
print("1->",q[1])
print("2->",q[2])
```

```
0-> Anna Caceres (22)
1-> Benjamin Alvarez (38)
2-> Carlos Barranco (15)
```

```
class Persona:
    def __init__(self,n,a,e):
        self.nom = n
        self.ape = a
        self.edad = e

    def __str__(self):
        t = self.nom+" "+self.ape+" ("+str(self.edad)+")"
        return t


p1 = Persona("Carlos","Barranco",15)
p2 = Persona("Anna","Caceres",22)
p3 = Persona("Benjamin","Alvarez",38)
q = [p1,p2,p3]

#1 - Definir la funcion para key
def porNombre(elem):
    return elem.nom


#2- Ordenar usando la key
```

Indicar la función anterior sin los paréntesis

```
print("0->",q[0])
print("1->",q[1])
print("2->",q[2])
```

```
0-> Anna Caceres (22)
1-> Benjamin Alvarez (38)
2-> Carlos Barranco (15)
```

```
class Persona:
    def __init__(self,n,a,e):
        self.nom = n
        self.ape = a
        self.edad = e

    def __str__(self):
        t = self.nom+" "+self.ape+" ("+str(self.edad)+")"
        return t


p1 = Persona("Carlos","Barranco",15)
p2 = Persona("Anna","Caceres",22)
p3 = Persona("Benjamin","Alvarez",38)
q = [p1,p2,p3]

#1 - Definir la funcion para key
def porNombre(elem):
    return elem.nom


#2- Ordenar usando la key
q.sort(key=porNombre)

print("0->",q[0])
print("1->",q[1])
print("2->",q[2])
```

```
0-> Anna Caceres (22)
1-> Benjamin Alvarez (38)
2-> Carlos Barranco (15)
```

¿Y por apellido?

```
#1 - Definir la funcion para key
def porApellido(elem):
    return elem.ape

#2- Ordenar usando la key
q.sort(key=porApellido)
```

```
0-> Benjamin Alvarez (38)
1-> Carlos Barranco (15)
2-> Anna Caceres (22)
```

¿Y por edad?

```
#1 - Definir la funcion para key
def porEdad(elem):
    return elem.edad

#2- Ordenar usando la key
q.sort(key=porEdad)
```

```
0-> Carlos Barranco (15)
1-> Anna Caceres (22)
2-> Benjamin Alvarez (38)
```

¿Y por … número de letras de su nombre?

```
#1 - Definir la funcion para key
def porNumLetras(elem):
    num = len(elem.nom)
    return num

#2- Ordenar usando la key
q.sort(key=porNumLetras)
```

```
0-> Anna Caceres (22)
1-> Carlos Barranco (15)
2-> Benjamin Alvarez (38)
```

¿Y por … número de vocales diferentes que tiene su nombre?

```
#1 - Definir la funcion para key
def porVocalDif(elem):
    vocales = []
    for c in elem.nom:
        c = c.lower()
        if (c in "aeiou") and (c not in vocales):
            vocales.append(c)
    return len(vocales)

#2- Ordenar usando la key
q.sort(key=porVocalDif)
```

```
0-> Anna Caceres (22)
1-> Carlos Barranco (15)
2-> Benjamin Alvarez (38)
```

U MULTI RANK

INTERESES

“vida universitaria”

Pregrado
- Intercambio UC
  - ¿Qué es el Intercambio?
  - Procesos y guía de postulación
  - Requisitos para intercambio
  - Testimonios de intercambio
  - Becas
- Doble grado
- Becas
- Postulación en línea
- Estado de Postulación
- Universidades con convenios
- Difusión DRI
- Lista de Cupos

Magíster

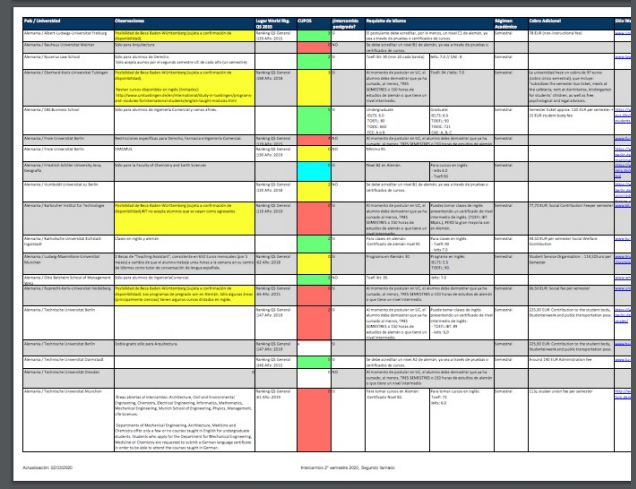
Doctorado

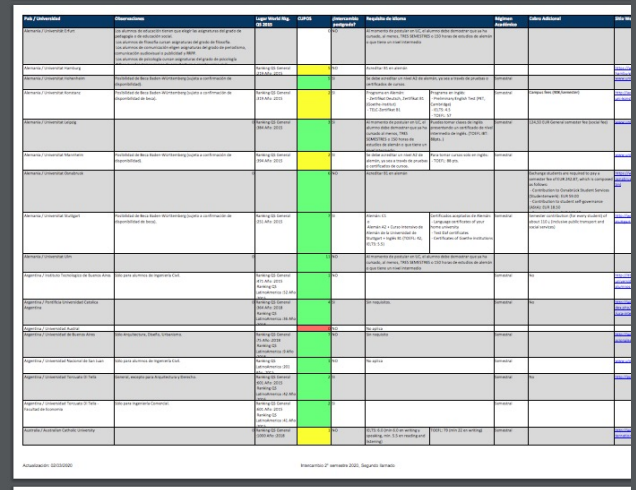
Inicio / Alumnos UC / Pregrado / Intercambio UC / Testimonios de intercambio

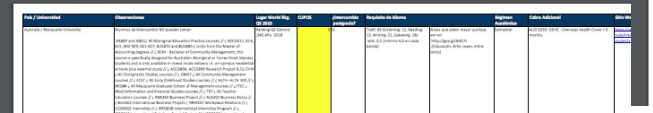
Testimonios de intercambio

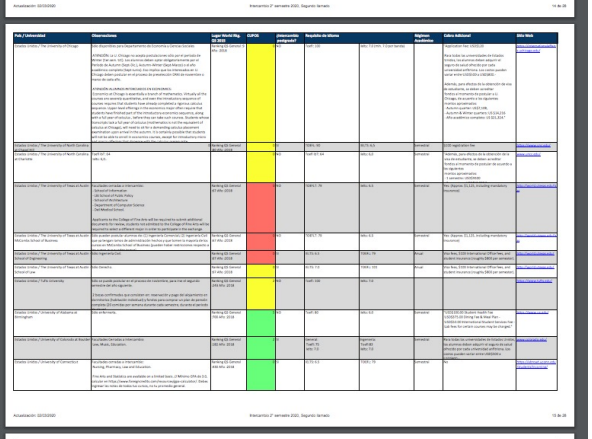
¿Todavía te quedan algunas dudas?
Escucha lo que tus compañeros quieren contarte sobre sus experiencias con Intercambio UC.

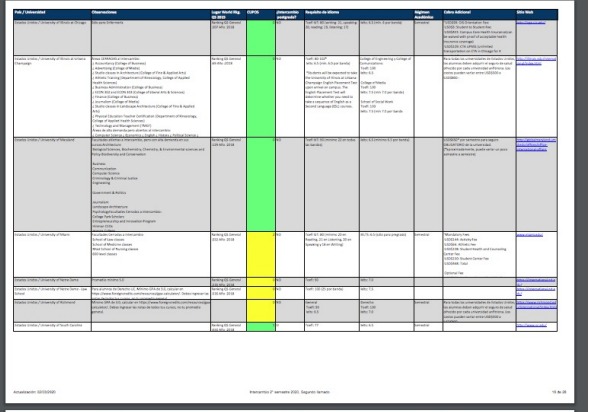


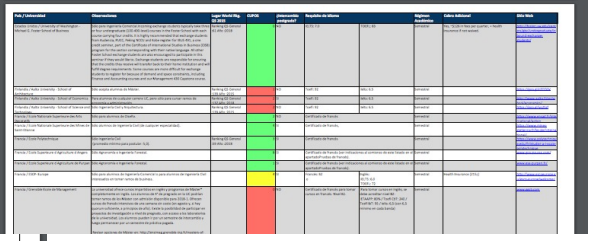


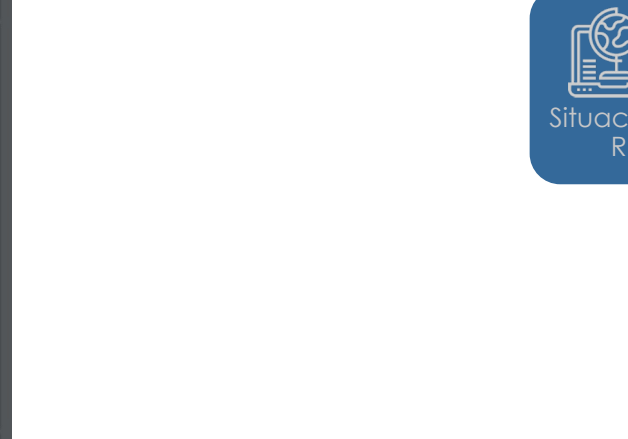


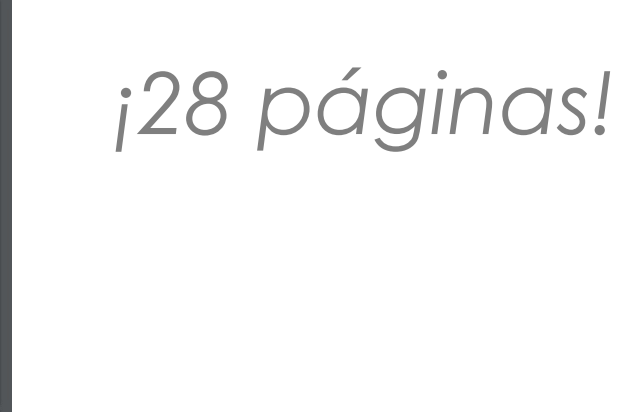












Situación Real

¡28 páginas!

Situación Real

LA TERCERA

publicidad

NACIONAL Educación Superior

UC avanza en Ranking QS y se acerca a las 100 mejores del mundo

Daniela Muñoz

6 JUN 2018 06:56 PM



QS TOPUNIVERSITIES Rankings

University search: Study Level Subject

1	Massachusetts Institute of Technology (MIT)
2	Stanford University
3	Harvard University
4	University of Oxford
5	California Institute of Technology (Caltech)
6	ETH Zurich - Swiss Federal Institute of Technology
7	University of Cambridge

Acreditación (ANECA) son reacios a estos *rankings* porque consideran que la calidad de las universidades no se puede medir en su totalidad. Consideran que existen tantas clasificaciones como perfiles de estudiantes y que la mejor universidad es la que más se a justa a las necesidades y preferencias del futuro

10 claves para identificar una universidad de calidad

¿Una pequeña facultad donde sea más fácil interactuar con el profesorado o un gran campus? El centro perfecto: el que mejor se adapta a las necesidades del alumno



What do you want to study?

Situación Real

Select level of degree

- (●) I'm looking for a first university course
- ( ) I already have a degree and I'm looking for an advanced course

Select a subject area

- Engineering
- Mathematics & Sciences
- Social Sciences
- Health

Select what matters most to you

- [ ] The teaching is good, and students learn well
- [ ] The students like it
- [ ] The university is strong in research
- [ ] A university with a focus on being international

Not Sure?

Help me to choose

```
class Uni:
    def __init__(self,n,d,i,t):
        self.nom = n
        self.doc = d
        self.inv = i
        self.trans = t

    def __str__(self):
        return self.nom


sts = Uni("Sants University"    ,8,7,6)
oso = Uni("OsloTech"            ,6,9,6)
zom = Uni("University of Zoom"  ,3,4,3)
vit = Uni("VIT"                 ,8,9,8)
unis = [sts,oso,zom,vit]
```

Ordenar de mejor a peor al gusto de Jorge (70% de peso a la Docencia, 30% a la Transferencia, y 0% a la Investigación)

```
for i in range(0,len(unis)):
    print(i,"->",unis[i])
```

```
0 -> VIT
1 -> Sants University
2 -> OsloTech
3 -> University of Zoom
```

Resolución Problema

```
class Uni:
    def __init__(self,n,d,i,t):
        self.nom = n
        self.doc = d
        self.inv = i
        self.trans = t

    def __str__(self):
        return self.nom


sts = Uni("Sants University"    ,8,7,6)
oso = Uni("OsloTech"            ,6,9,6)
zom = Uni("University of Zoom"  ,3,4,3)
vit = Uni("VIT"                 ,8,9,8)
unis = [sts,oso,zom,vit]

#1 - Definir la funcion para key
def alGustoDeJorge(elem):
    puntos = (elem.doc*0.7) + (elem.trans*0.3)
    return puntos

#2- Ordenar descendentemente usando la key
unis.sort(key=alGustoDeJorge)
unis.reverse()

for i in range(0,len(unis)):
    print(i,"->",unis[i])
```

```
0 -> VIT
1 -> Sants University
2 -> OsloTech
3 -> University of Zoom
```

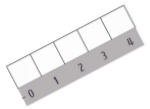
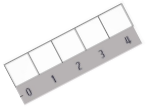
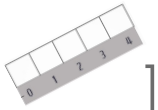
- U Multirank
  - https://www.umultirank.org/
  - https://comein.uoc.edu/divulgacio/comein/es/numero57/articles/Article-Victor-Cavaller.html
  - https://en.wikipedia.org/wiki/College_and_university_rankings#U-Multirank
  - https://elpais.com/economia/2014/09/16/actualidad/1410883809_851331.html
- Ranking QS
  - https://www.topuniversities.com/university-rankings/world-university-rankings/2020
- Intercambio UC
  - https://relacionesinternacionales.uc.cl/alumnos-uc/pregrado

ORDENAR LISTA DE LISTAS

[5 , 8 , 2]

[“casa” , “auto” , “barco”]

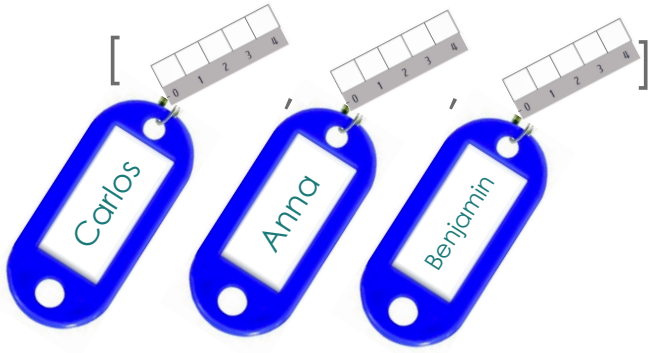
[ ,  , ]

[ ,  , ]

[5 , 8 , 2]

[“casa” , “auto” , “barco”]

[ ,  , ]

[

]

¿Por interrogación?

```
#Nombre, Interrog, Tareas
a = ["Alice"  ,6.5,5.5]
b = ["Bob"    ,4.3,5.9]
c = ["Charlie",1.3,6.1]
ns = [a,b,c]

#1- Definir la funcion para key
def porInt(e):
    return e[1]

#2- Ordenar usando la key
ns.sort(key=porInt)

print("0->",ns[0])
print("1->",ns[1])
print("2->",ns[2])
```

```
0-> ['Charlie', 1.3, 6.1]
1-> ['Bob', 4.3, 5.9]
2-> ['Alice', 6.5, 5.5]
```

¿Por tarea?

```
#Nombre, Interrog, Tareas
a = ["Alice"  ,6.5,5.5]
b = ["Bob"    ,4.3,5.9]
c = ["Charlie",1.3,6.1]
ns = [a,b,c]

#1- Definir la funcion para key
def porTareas(e):
    return e[2]

#2- Ordenar usando la key
ns.sort(key=porTareas)

print("0->",ns[0])
print("1->",ns[1])
print("2->",ns[2])
```

```
0-> ['Alice', 6.5, 5.5]
1-> ['Bob', 4.3, 5.9]
2-> ['Charlie', 1.3, 6.1]
```

¿Por promedio?


```
#Nombre, Interrog, Tareas
a = ["Alice"  ,6.5,5.5]
b = ["Bob"    ,4.3,5.9]
c = ["Charlie",1.3,6.1]
ns = [a,b,c]

#1- Definir la funcion para key
def porPromedio(e):
    promedio = e[1]+e[2]/2
    return promedio

#2- Ordenar usando la key
ns.sort(key=porPromedio)

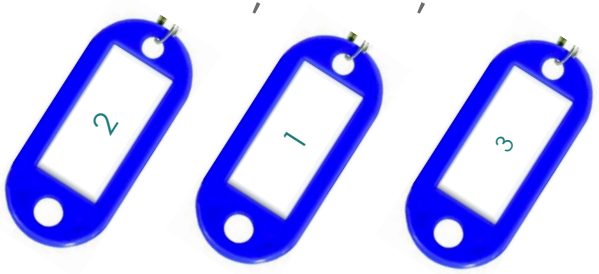
print("0->",ns[0])
print("1->",ns[1])
print("2->",ns[2])
```

```
0-> ['Charlie', 1.3, 6.1]
1-> ['Bob', 4.3, 5.9]
2-> ['Alice', 6.5, 5.5]
```

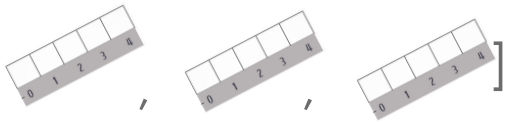
ORDENAR LISTAS DE STR/INT A MI MANERA

[5 , 8 , 2]

[“casa” , “auto” , “barco”]



[]

[]

¿Y podríamos usar el key para ordenar de forma diferente una lista de str o ints?

¡Obvio!

(List de strings por número de 'a')

```
Antes:  ['raton', 'botella', 'taza', 'camara', 'ventana', 'libro']
Despues:  ['libro', 'raton', 'botella', 'taza', 'ventana', 'camara']
```

```
p = ["raton","botella","taza","camara","ventana","libro"]
print("Antes: ", p)

def porAs(e):
    numA = 0
    for c in e:
        if c == "a":
            numA += 1
    return numA

p.sort(key=porAs)

print("Despues: ", p)
```

```
Antes:  ['raton', 'botella', 'taza', 'camara', 'ventana', 'libro']
Despues:  ['libro', 'raton', 'botella', 'taza', 'ventana', 'camara']
```

Incluso un sort que no tenga en cuenta may/min sin usar comandos raros …

```
Antes:  ['casa', 'caracol', 'Barco', 'Arbol', 'Auto', 'arte', 'Cortina']
Despues:  ['Arbol', 'arte', 'Auto', 'Barco', 'caracol', 'casa', 'Cortina']
```

```
Antes:  ['casa', 'caracol', 'Barco', 'Arbol', 'Auto', 'arte', 'Cortina']
Despues:  ['Arbol', 'arte', 'Auto', 'Barco', 'caracol', 'casa', 'Cortina']
```

```python
p = ["casa","caracol","Barco","Arbol","Auto","arte","Cortina"]
print("Antes: ", p)

def porNoMay(e):
    return e.lower()

p.sort(key=porNoMay)

print("Despues: ", p)
```

ORDENAR POR VARIOS CRITERIOS

```
#1 - Definir la funcion para key
def porApeYLuegoNombre(elem):
    return elem.ape,elem.nom
```

```
0-> Benjamin Barranco (38)
1-> Carlos Barranco (15)
2-> Anna Caceres (22)
```

Si la función de key retorna varias cosas separadas por comas, ordena por la primera, y en caso de empate por la segunda, …

ORDENAR CARTAS

INTERESES

“Tipo Prueba”

Situación Real

Pregunta 3

Para bajar el estrés luego de los exámenes, una buena idea es invitar a tus compañeros de curso a jugar naipes. Piensan en juntarse luego de esta prueba, y decides empezar a ejercitar tus habilidades con las cartas al mismo tiempo que practicas programación.

Parte a) (30 puntos)

Escribe una función `ordenar_cartas` que reciba una lista de cartas y las ordene según pinta y número. No es necesario que la función retorne una nueva lista; basta con ordenar la lista de entrada.

La lista de cartas está en el formato $[[n_1, p_1], [n_2, p_2], ...]$

donde n_i es el número de la carta i y p_i es su pinta.

La pinta puede ser C (corazones), P (picas), D (diamantes) o T (tréboles). Considera que el orden de las pintas, de izquierda a derecha es corazones, picas, diamantes y tréboles. No hay cartas repetidas.

Un ejemplo de mano ordenada de cartas se muestra en la Figura 2. Los números de las cartas van del 1 al 13. (El As se representa por 1 y las cartas literales (J,Q,R) por 11, 12 y 13.)

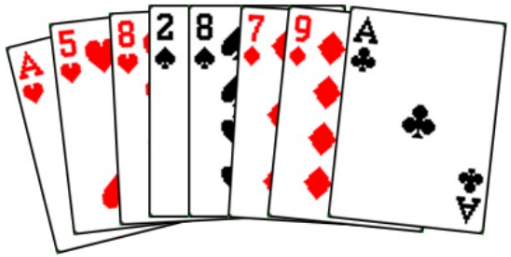


Figura 2: Conjunto de cartas ordenado según número y pinta, de izquierda a derecha

```
Des: [[8, 'P'], [9, 'D'], [1, 'T'], [8, 'C'], [2, 'P'], [5, 'C'], [1, 'C'], [7, 'D']]
Ord: [[1, 'C'], [5, 'C'], [8, 'C'], [2, 'P'], [8, 'P'], [7, 'D'], [9, 'D'], [1, 'T']]
```




I'LL WAIT
FOR YOU HERE

Situación Real

Pregunta 3

Para bajar el estrés luego de los exámenes, una buena idea es invitar a tus compañeros de curso a jugar naipes. Piensan en juntarse luego de esta prueba, y decides empezar a ejercitar tus habilidades con las cartas al mismo tiempo que practicas programación.

Parte a) (30 puntos)

Escribe una función `ordenar_cartas` que reciba una lista de cartas y las ordene según pinta y número. No es necesario que la función retorne una nueva lista; basta con ordenar la lista de entrada.

La lista de cartas está en el formato $[[n_1, p_1], [n_2, p_2], ...]$

donde n_i es el número de la carta i y p_i es su pinta.

La pinta puede ser C (corazones), P (picas), D (diamantes) o T (tréboles). Considera que el orden de las pintas, de izquierda a derecha es corazones, picas, diamantes y tréboles. No hay cartas repetidas.

Un ejemplo de mano ordenada de cartas se muestra en la Figura 2. Los números de las cartas van del 1 al 13. (El As se representa por 1 y las cartas literales (J,Q,R) por 11, 12 y 13.)

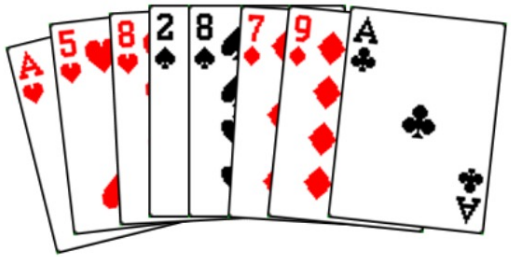


Figura 2: Conjunto de cartas ordenado según número y pinta, de izquierda a derecha

```
Des: [[8, 'P'], [9, 'D'], [1, 'T'], [8, 'C'], [2, 'P'], [5, 'C'], [1, 'C'], [7, 'D']]
Ord: [[1, 'C'], [5, 'C'], [8, 'C'], [2, 'P'], [8, 'P'], [7, 'D'], [9, 'D'], [1, 'T']]
```

```
def porPintaYNum(e):
    n = e[0]
    p = e[1]

    #Corazones el 0, picas el 1, ...
    if p == "C":
        np = 0
    elif p == "P":
        np = 1
    elif p == "D":
        np = 2
    elif p == "T":
        np = 3

    return np,n

def ordenar_cartas(b):
    b.sort(key=porPintaYNum)
```

¡¡¿¿Tu siempre dices que esto NO es un curso de comandos sino de pensamiento computacional, y ahora nos enseñas lo de los dos criterios??!!

¡Tienes toda la razón!

Situación Real

Pregunta 3

Para bajar el estrés luego de los exámenes, una buena idea es invitar a tus compañeros de curso a jugar naipes. Piensan en juntarse luego de esta prueba, y decides empezar a ejercitar tus habilidades con las cartas al mismo tiempo que practicas programación.

Parte a) (30 puntos)

Escribe una función `ordenar_cartas` que reciba una lista de cartas y las ordene según pinta y número. No es necesario que la función retorne una nueva lista; basta con ordenar la lista de entrada.

La lista de cartas está en el formato $[[n_1, p_1], [n_2, p_2], ...]$

donde n_i es el número de la carta i y p_i es su pinta.

La pinta puede ser C (corazones), P (picas), D (diamantes) o T (tréboles). Considera que el orden de las pintas, de izquierda a derecha es corazones, picas, diamantes y tréboles. No hay cartas repetidas.

Un ejemplo de mano ordenada de cartas se muestra en la Figura 2. Los números de las cartas van del 1 al 13. (El As se representa por 1 y las cartas literales (J,Q,R) por 11, 12 y 13.)

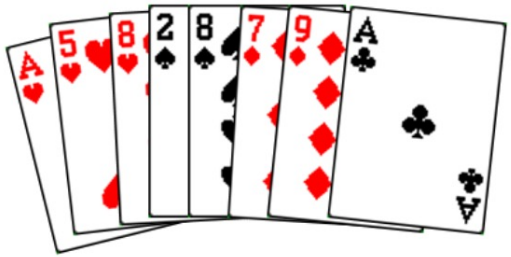


Figura 2: Conjunto de cartas ordenado según número y pinta, de izquierda a derecha

```
Des: [[8, 'P'], [9, 'D'], [1, 'T'], [8, 'C'], [2, 'P'], [5, 'C'], [1, 'C'], [7, 'D']]
Ord: [[1, 'C'], [5, 'C'], [8, 'C'], [2, 'P'], [8, 'P'], [7, 'D'], [9, 'D'], [1, 'T']]
```

```
def porPintaYNumMan(e):
    n = e[0]
    p = e[1]

    #Los corazones seran los 100algo,
    #Las picas los 200algo, ...
    if p == "C":
        v = 100+n
    elif p == "P":
        v = 200+n
    elif p == "D":
        v = 300+n
    elif p == "T":
        v = 400+n

    return v

def ordenar_cartas(b):
    b.sort(key=porPintaYNumMan)
```

¿… y puedo usar el key para que sea ascendente en un criterio pero descendente en otro?

¡Lo que quieras!

```
Des: [[8, 2], [9, 3], [1, 4], [8, 1], [2, 2], [5, 1], [1, 1], [7, 3]]
Ord: [[1, 4], [1, 1], [2, 2], [5, 1], [7, 3], [8, 2], [8, 1], [9, 3]]
```

```
def porAscDes(e):
    n1 = e[0]
    n2 = e[1]

    # Para n2 quiero que 12 antes de 4
    # Como mayor es el n2 mas pequeno lo quiero
    # Pues lo convierto en negativo
    # (multiplicandolo por -1)
    # -12 es mas pequeno que -4
    nn2 = n2 * -1

    return n1, nn2

def ordenar_cartas(b):
    b.sort(key=porAscDes)
```

```
Des: [[8, 2], [9, 3], [1, 4], [8, 1], [2, 2], [5, 1], [1, 1], [7, 3]]
Ord: [[1, 4], [1, 1], [2, 2], [5, 1], [7, 3], [8, 2], [8, 1], [9, 3]]
```

ALGORITMOS DE ORDENAMIENTO

INTERESES

“¿qué hay detrás de los comandos?”

¿Qué hay detras del sort?



- https://www.toptal.com/developers/sorting-algorithms

Ordenamiento por Selección

- Temario solo entra ‘Ordenamiento por Selección’
- Idea:
  - Miro el minimo de 0 a N y lo pongo en 0
  - Luego miro el mínimo de 1 a N y lo pongo en 1
  - Luego el mínimo de 2 a N y lo pongo en 2
  - Y así.

- https://es.wikipedia.org/wiki/Ordenamiento_por_selecci%C3%B3n

	8
	5
	2
	6
	9
	3
	1
	4
	0
	7

```
def ordenacion_seleccion(p):
    ini = 0
    while ini < len(p):

        #Buscar la pos del minimo desde ini hasta el final
        i = ini
        pos_min = -1
        while i < len(p):
            if p[i] < p[pos_min]:
                pos_min = i
            i += 1

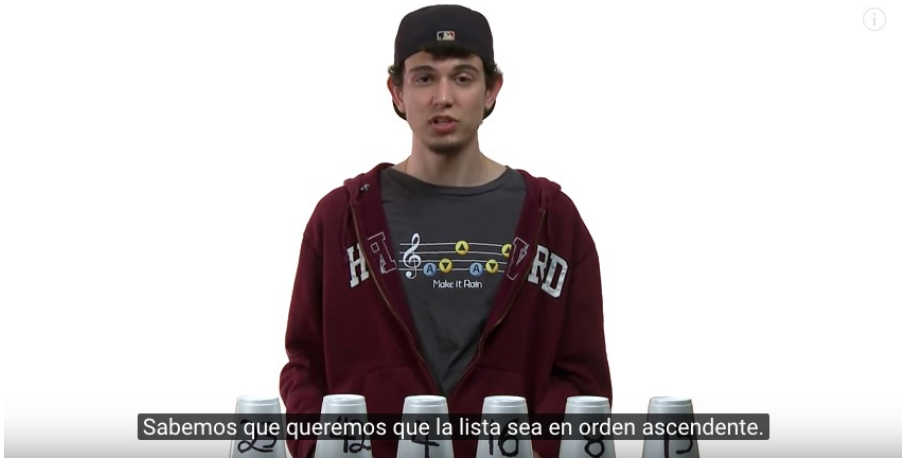
        #Intercambiar el de ini y pos_min
        aux = p[ini]
        p[ini] = p[pos_min]
        p[pos_min] = aux

        #Incrementar el ini
        ini += 1



p = [6, 14, 3, 8, 50, 4, 30, 44, 12]
print("Original", p)
ordenacion_seleccion(p)
print("Ordenado", p)
```

```
Original [6, 14, 3, 8, 50, 4, 30, 44, 12]
Ordenado [3, 4, 6, 8, 12, 14, 30, 44, 50]
```



https://www.youtube.com/watch?v=f8hXR_Hvybo

https://www.youtube.com/watch?v=Ns4TPTC8whw

