11 – ARCHIVOS

Jorge Muñoz

IIC1103 – Introducción a la Programación

ARCHIVOS

Calcular el promedio de la I1

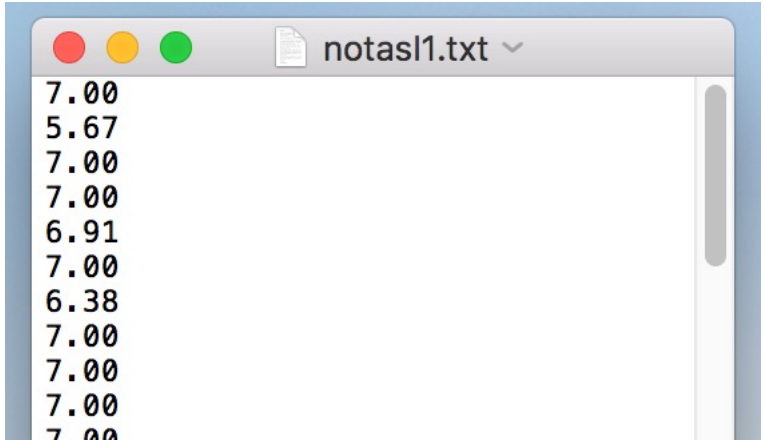

```
nest = int(input("Num de Estudiantes? "))

suma = 0
for i in range(0,nest):
    nota = float(input("Nota? "))
    suma += nota

print("Promedio:",suma/nest)
```

```
ntotal = int(input("Num de Estudiantes? "))

nest = 0
suma = 0
for i in range(0,ntotal):
    nota = float(input("Nota? "))
    if nota != 1.0:
        nest += 1
        suma += nota

print("Promedio:",suma/nest)
```

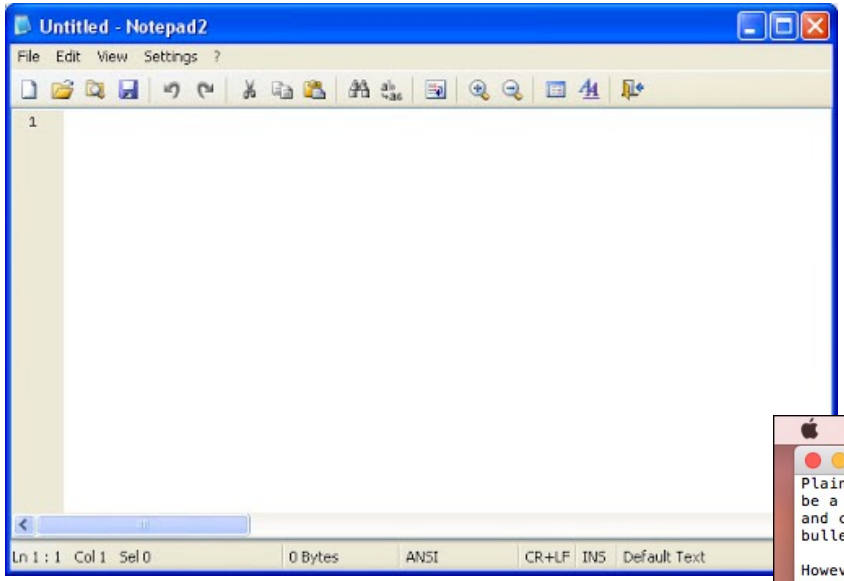
```
Num de Estudiantes? 230
Nota? 6.5
Nota? 5.4
Nota? 4.6
Nota? .... uffff 230 a mano??
```

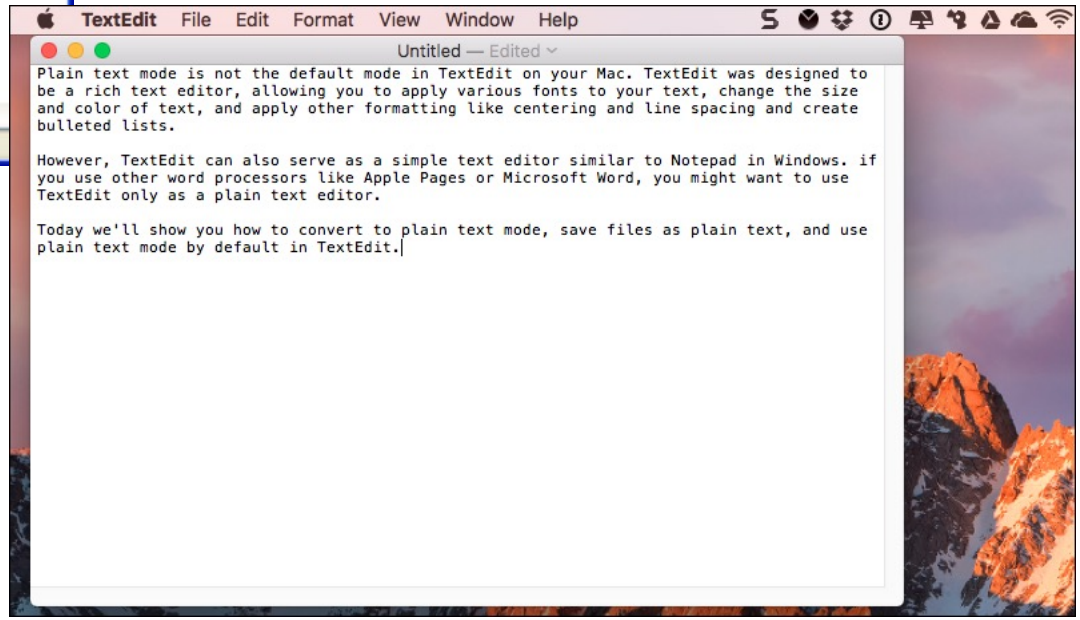
```
Num de Estudiantes? 230
Nota? ESTAS LOCO! Otra vez todos!!??
```

Situación Real



Situación Real





LEER

memesintroalaprogr • Following ...

82 likes

DECEMBER 8, 2019

Add a comment... Post

memesintroalaprogr • Following

583 views

DECEMBER 9, 2019

Add a comment... Post

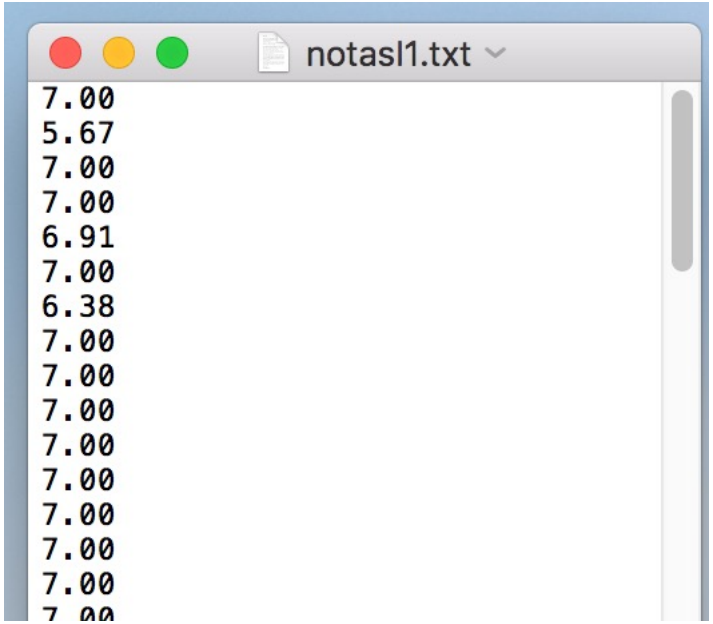
```
# OPEN - READLINE - CLOSE
uwu = open("MiArchivo.txt")
lineas = uwu.readlines()
uwu.close()

# 'lineas' es una list de str
# donde cada elemento es una linea del archivo
lineas ... #Haz lo que quieras
```

PROMEDIO SIN 1

INTERESES

“cosas útiles para un ayudante”

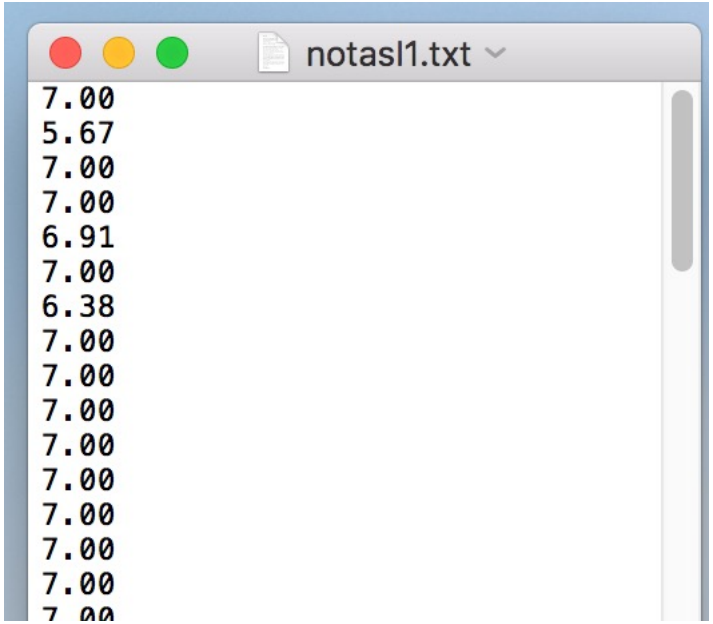
Calcular el promedio de la I1 sin contar los 1 (estudiante que no se presentó)



Promedio: ¿Tu apuesta?


I'LL WAIT
FOR YOU HERE

Calcular el promedio de la I1 sin contar los 1 (estudiante que no se presentó)

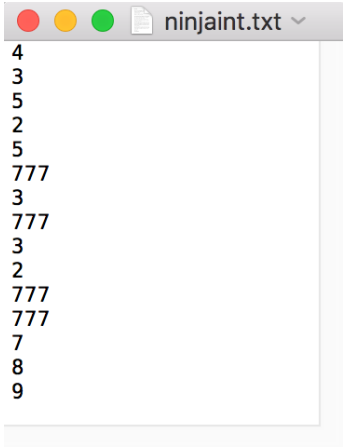



```
Promedio: 6.608741339491920
```

```
# OPEN - READLINE - CLOSE
r = open("notasI1.txt")
ls = r.readlines()
r.close()

# 'ls' es una list de str
# donde cada elemento es una linea del archivo
```

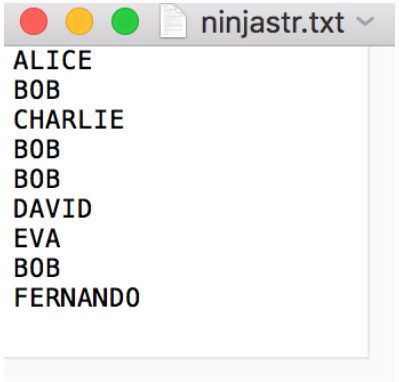
```
# OPEN - READLINE - CLOSE
r = open("notasI1.txt")
ls = r.readlines()
r.close()

# 'ls' es una list de str
# donde cada elemento es una linea del archivo
nest = 0
suma = 0
for l in ls:
    nota = float(l)
    if nota != 1.0:
        nest += 1
        suma += nota

print("Promedio:",suma/nest)
```

EL NINJA SILENCIOSO (STRIP)




```
r = open("ninjaint.txt")
ls = r.readlines()
r.close()

num = 0
for l in ls:
    n = int(l)
    if n == 777:
        num += 1
print("777 ha aparecido",num,"veces")
```

777 ha aparecido 4 veces




```
r = open("ninjaint.txt")
ls = r.readlines()
r.close()

num = 0
for l in ls:
    if l == "BOB":
        num += 1
print("BOB ha aparecido",num,"veces")
```

BOB ha aparecido 0 veces


ninjastr.txt
ALICE\n
BOB\n
CHARLIE\n
BOB\n
BOB\n
DAVID\n
EVA\n
BOB\n
FERNANDO\n

"BOB" == "BOB\n"


strip()
La vieja confiable

```
r = open("ninjastr.txt")
ls = r.readlines()
r.close()

num = 0
for l in ls:
    lx = l.strip() # 'lx' es 'l' pero sin el \n
    if lx == "BOB":
        num += 1
print("BOB ha aparecido",num,"veces")
```

```
BOB ha aparecido 4 veces
```

JURASSIC LOVE

INTERESES

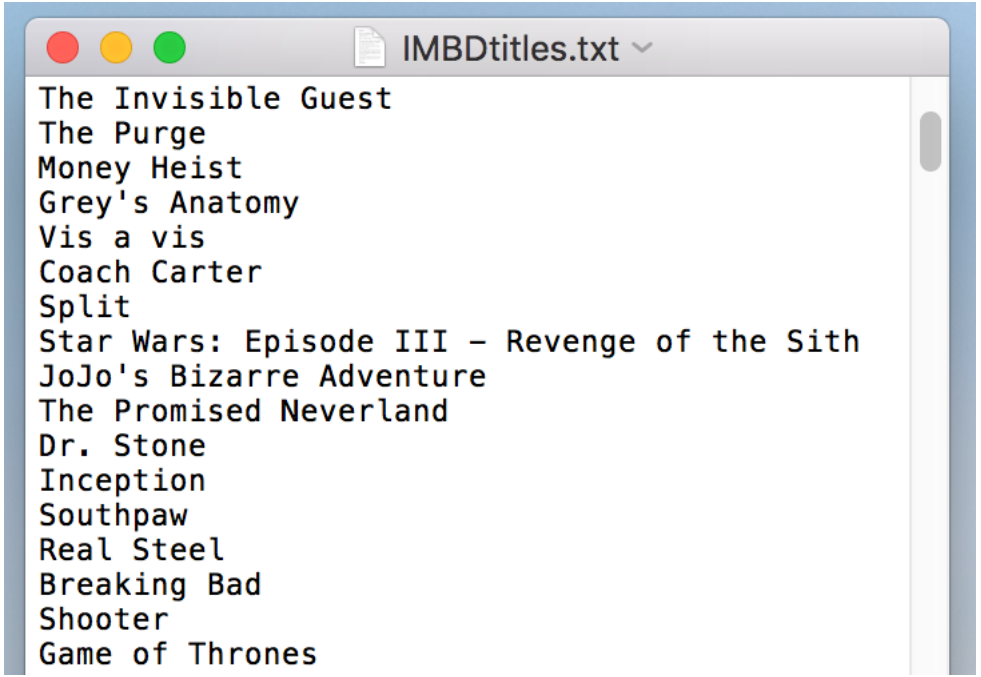
“Va a mandar las lista con todas las recomendaciones?”

“Buena encuestas, ¿Cuál es el fin? podremos ver las recomendaciones de otros?”

“Podría después publicar un excel con todas sus recomendaciones y las de los alumnos para tenerlo guardado en el pc?”

Situación Real

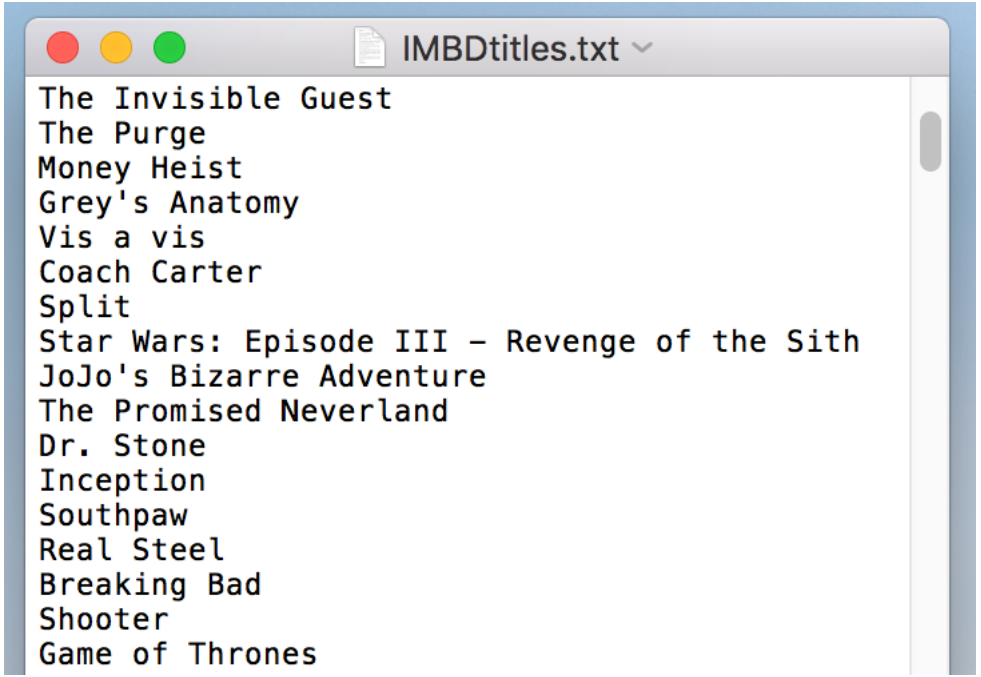
```
Palabra? love
7 veces

Palabra? jurassic
2 veces

Palabra? jorge
0 veces

Palabra? naruto
6 veces

Palabra? Forrest gump
12 veces
```




I'LL WAIT
FOR YOU HERE

Situación Real

```
Palabra? love
7 veces

Palabra? jurassic
2 veces

Palabra? jorge
0 veces

Palabra? naruto
6 veces

Palabra? Forrest gump
12 veces
```



```
r = open("IMDBtitles.txt")
ls = r.readlines()
r.close()

p = input("Palabra? ")
num = 0
for l in ls:
    peli = l.strip()
    if p.lower() in peli.lower():
        num += 1

print(num,"veces")
```

AND THE OSCAR GOES TO …

INTERESES

“Va a mandar las lista con todas las recomendaciones?”

“Buena encuestas, ¿Cuál es el fin? podremos ver las recomendaciones de otros?”

“Podría después publicar un excel con todas sus recomendaciones y las de los alumnos para tenerlo guardado en el pc?”

Avengers: Endgame
Fight Club
The Invisible Guest
Inception
Interstellar
Forrest Gump
The Godfather
Joker
The Prestige
The Lord of the Rings: The Return of the King
Pulp Fiction

Deadpool
Avengers: Infinity War
1917
Parasite
Shrek
Shrek 2
Howl's Moving Castle
The Theory of Everything
Your Name.
Shutter Island
Just Go with It
The Dark Knight

35 Interstellar

20 Avengers: Endgame

17 Parasite

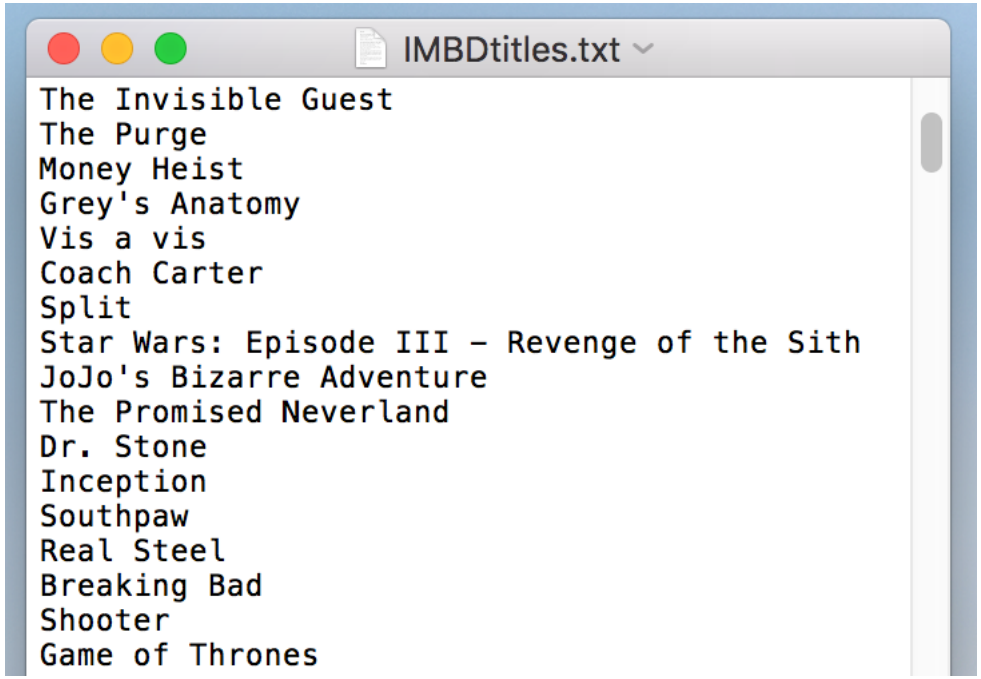
Chernobyl
Suits
Better Call Saul
Mindhunter
Black Mirror
Hunter x Hunter
Attack on Titan
How I Met Your Mother
Avatar: The Last Airbender
BoJack Horseman
Dark
Prison Break
Death Note
Sex Education
Parasite
The Simpsons

Friends
Breaking Bad
Fullmetal Alchemist: Brotherhood
Elite
Money Heist
Grey's Anatomy
Vis a vis
Game of Thrones
Vikings
Brooklyn Nine-Nine
Peaky Blinders
The Office
Sherlock
Stranger Things
Rick and Morty
Cosmos: A Spacetime Odyssey

33 Breaking Bad

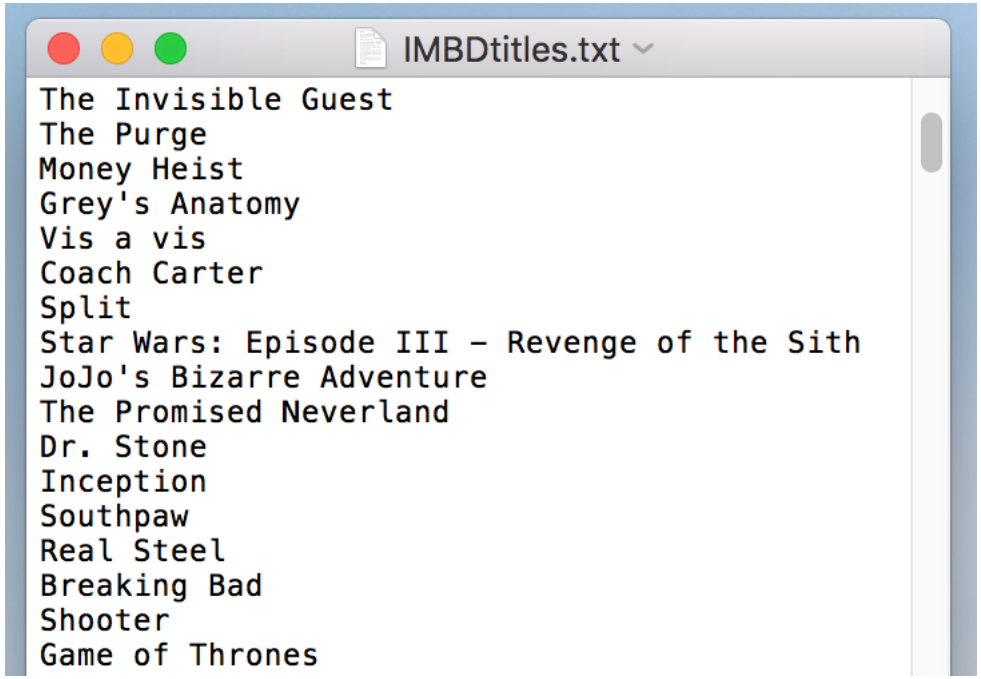
18 Game of Thrones
18 Peaky Blinders

26 Money Heist



Ganador: Interstellar con 35 votos


I'LL WAIT
FOR YOU HERE



Ganador: Interstellar con 35 votos

```
r = open("IMDBtitles.txt")
ls = r.readlines()
r.close()
```

Resolución Problema

```
r = open("IMDBtitles.txt")
ls = r.readlines()
r.close()

# Lista de pelis, donde cada peli es
# una list [nom,num]
pelis = []

#Crear esta lista de pelis y nums
for l in ls:
    nom = l.strip()

    #Mirar si esta
    #Y si esta incrementar
    esta = False
    for peli in pelis:
        if peli[0] == nom:
            peli[1] += 1
            esta = True

    #Si no esta la creo y la agrego
    if not esta:
        x = [nom,1]
        pelis.append(x)
```

```
r = open("IMDBtitles.txt")
ls = r.readlines()
r.close()

# Lista de pelis, donde cada peli es
# una list [nom,num]
pelis = []

#Crear esta lista de pelis y nums
for l in ls:
    nom = l.strip()

    #Mirar si esta
    #Y si esta incrementar
    esta = False
    for peli in pelis:
        if peli[0] == nom:
            peli[1] += 1
            esta = True

    #Si no esta la creo y la agrego
    if not esta:
        x = [nom,1]
        pelis.append(x)


#Buscar la mas votada
maxnum = -1
maxnom = ""
for peli in pelis:
    if peli[1] > maxnum:
        maxnum = peli[1]
        maxnom = peli[0]

print("Ganador:",maxnom,"con",maxnum,"votos")
```

AND THE OSCAR GOES TO … 2020-2

Avengers: Endgame
Fight Club
The Invisible Guest
Inception
Interstellar
Forrest Gump
The Godfather
Joker
The Prestige
The Lord of the Rings: The Return of the King
Pulp Fiction
Deadpool
Avengers: Infinity War
1917
Parasite
Shrek
Shrek 2
Howl's Moving Castle
The Theory of Everything
Your Name.
Shutter Island
Just Go with It
The Dark Knight

Interstellar 14

Avengers: Endgame 10

Inception 8

Chernobyl
Suits
Better Call Saul
Mindhunter
Black Mirror
Hunter x Hunter
Attack on Titan
How I Met Your Mother
Avatar: The Last Airbender
BoJack Horseman
Dark
Prison Break
Death Note
Sex Education
Parasite
The Simpsons

Friends
Breaking Bad
Fullmetal Alchemist: Brotherhood
Elite
Money Heist
Grey's Anatomy
Vis a vis
Game of Thrones
Vikings
Brooklyn Nine-Nine
Peaky Blinders
The Office
Sherlock
Stranger Things
Rick and Morty
Cosmos: A Spacetime Odyssey

Breaking Bad 21

Friends 19

Grey's Anatomy 16

AND THE OSCAR GOES TO … 2021-1

Your Name.
Coraline
Avengers: Infinity War
Shutter Island
Shrek 2
Shrek
Avengers: Endgame
Forrest Gump
The Wolf of Wall Street
The Intouchables

The Dark Knight
Fight Club
Whiplash
Pirates of the Caribbean: The Curse of the Black Pearl
How to Lose a Guy in 10 Days
Howl's Moving Castle
Hacksaw Ridge
Inception
The Godfather
Interstellar

Interstellar 9

Shrek 2 7

Inception 6

Modern Family
The Office
House
Stranger Things
Gossip Girl
How I Met Your Mother
Attack on Titan
Elite
Avatar: The Last Airbender
Prison Break

Friends
Game of Thrones
Breaking Bad
Peaky Blinders
Money Heist
BoJack Horseman
Dark
Grey's Anatomy
Vikings
Malcolm in the Middle

Breaking Bad 16

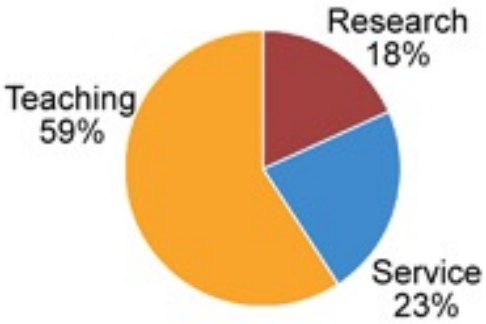
Grey's Anatomy 14

Game of Thrones 11
Peaky Blinders 11

CARACTERES ESPECIALES (Fuera del Temario)

HOW PROFESSORS SPEND THEIR TIME

How they actually spend their time:



Source: Higher Education Research Institute Survey (1999)

How departments expect them to spend their time:

How Professors would *like* to spend their time:

Control-flow analysis of procedural skills competencies in medical training through process mining

Rene de la Fuente ,[1] Ricardo Fuentes ,[1] Jorge Munoz-Gama,[2] Arnoldo Riquelme,[3] Fernando R. Altermatt,[1] Juan Pedemonte,[1] Marcia Corvetto,[1] Marcos Sepúlveda[2]

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

Recomposing conformance: Closing the circle on decomposed alignment-based conformance checking in process mining

Wai Lam Jonathan Lee[a], H.M.W. Verbeek[b], Jorge Munoz-Gama[a,*], Wil M.P. van der Aalst[c], Marcos Sepúlveda[a]

[a] Department of Computer Science, School of Engineering, Pontificia Universidad Católica de Chile, Vicuña Mackenna 4860, Macul 7820436, Chile
[b] Department of Mathematics and Computer Science, Eindhoven University of Technology, The Netherlands
[c] Lehrstuhl für Informatik 9 / Process and Data Science, RWTH Aachen University, D-52056 Aachen, Germany

Contents lists available at ScienceDirect

Computers in Human Behavior

journal homepage: www.elsevier.com/locate/comphumbeh

Full length article

Mining theory-based patterns from Big data: Identifying self-regulated learning strategies in Massive Open Online Courses

Jorge Maldonado-Mahauad [a, b, *], Mar Pérez-Sanagustín [a], René F. Kizilcec [c], Nicolás Morales [a], Jorge Munoz-Gama [a]

[a] Pontificia Universidad Católica de Chile, Department of Computer Science, Avda. Vicuña Mackenna 4860, Macul, Santiago, Chile
[b] Universidad de Cuenca, Department of Computer Science, Av. 12 Abril, Cuenca, Ecuador
[c] Stanford University, Graduate School of Education, 485 Lausen Mall, Stanford, CA 94305, USA

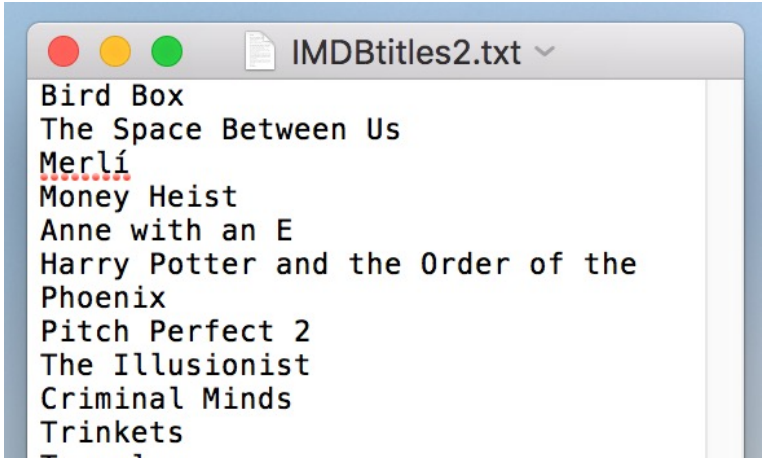
A Fresh Look at Precision in Process Conformance

Jorge Muñoz-Gama and Josep Carmona

Universitat Politècnica de Catalunya, Spain
jmunoz@lsi.upc.edu, jcarmona@lsi.upc.edu

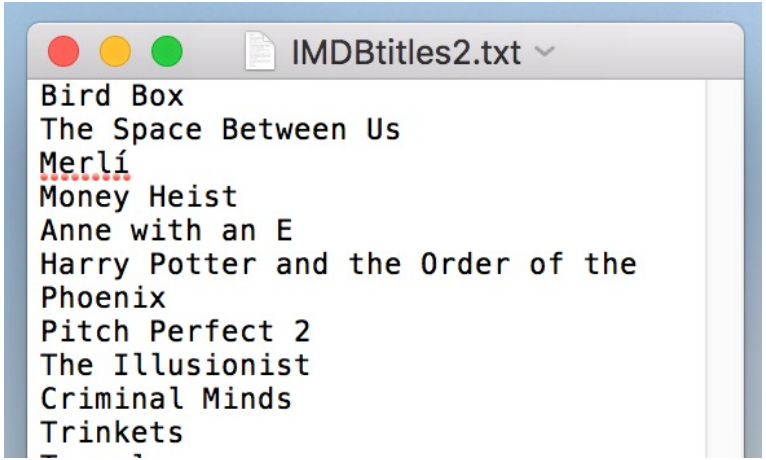
7. J. Muoz-Gama and J. Carmona. A Fresh Look at Precision in Process Conformances. In *Proceedings of the 8th International Conference on Business Process Management (BPM 2010)*, 2010.

Mu, J., & Carmona, J. (2010). A fresh look at Precision in Process Conformance. *Business Process Management, 211-226.*




```
r = open("IMDBtitles2.txt")
ls = r.readlines()
r.close()

print("Hay",len(ls),"peliculas y series")
```

```
    ls = r.readlines()
  File "/Library/Frameworks/Python.framework/Versions/3.4/lib/python3.4/encodings
/ascii.py", line 26, in decode
    return codecs.ascii_decode(input, self.errors)[0]
UnicodeDecodeError: 'ascii' codec can't decode byte 0xc2 in position 1236: ordina
l not in range(128)
```




```
r = open("IMDBtitles2.txt", encoding="utf-8")
ls = r.readlines()
r.close()

print("Hay",len(ls),"peliculas y series")
```

```
Hay 1216 peliculas y series
```

NO ENTRA EN EL TEMARIO

- Encoding
  - https://en.wikipedia.org/wiki/Character_encoding
  - https://en.wikipedia.org/wiki/ASCII
  - https://en.wikipedia.org/wiki/UTF-8

LET

INTERESES

“universidad”

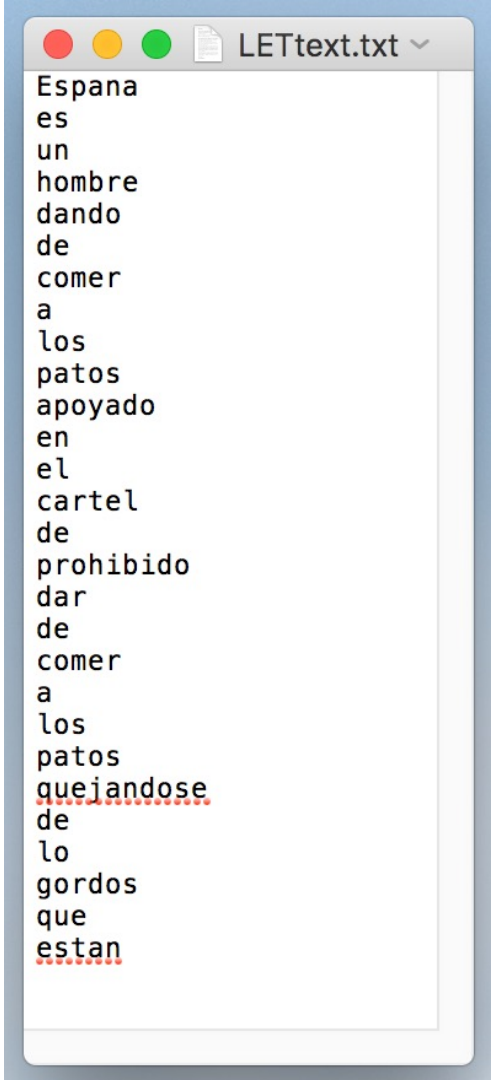
“vida cuotidiana”

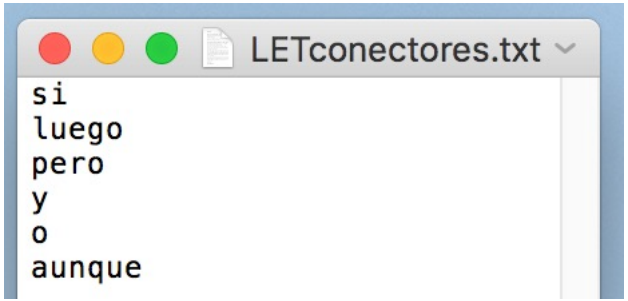
“útiles”

Estimado Profe:

Esperando que se encuentre muy bien, le escribo porque le dije que le enviaría el primer código que creé con fines externos al curso. Contextualizando, para el curso de habilidades comunicativas (LET) se necesita hacer un informe, una de las condiciones es que es un párrafo no se deben repetir palabras que no sean conectores, y pues a mi siempre se me pasaban y me costaba ver cuales se repetían, ante esto decidí programar y me ayudó muchísimo, este es el código. Le quiero dar las gracias por ser tan buen profe, y haber cambiado mi visión de las cosas.

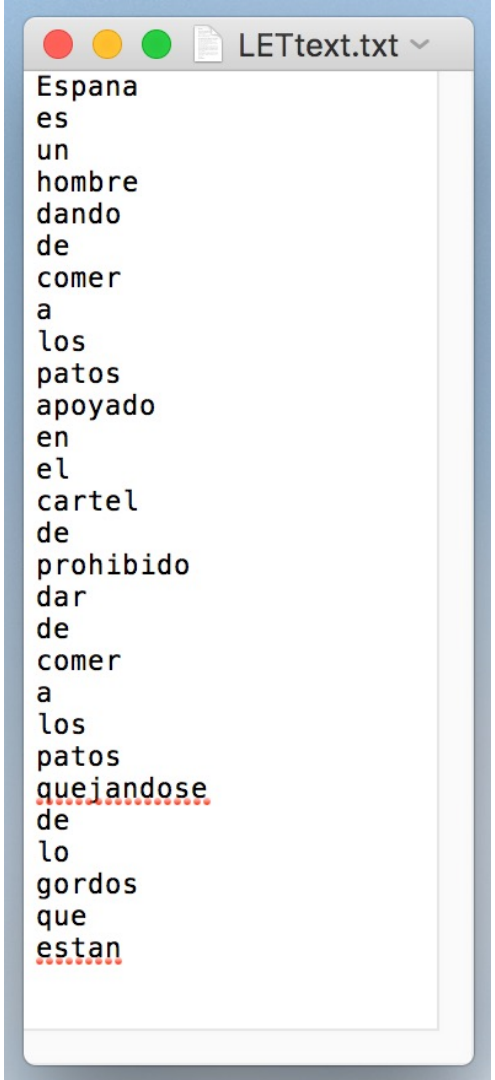
Muchos saludos

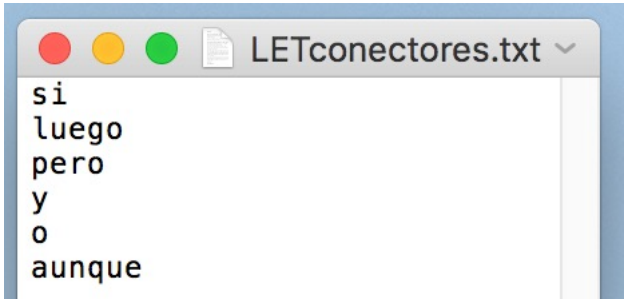

LETtext.txt
Espana
es
un
hombre
dando
de
comer
a
los
patos
apoyado
en
el
cartel
de
prohibido
dar
de
comer
a
los
patos
quejandose
de
lo
gordos
que
estan


LETconectores.txt
si
luego
pero
y
o
aunque

comer
patos


I'LL WAIT
FOR YOU HERE


LETtext.txt
Espana
es
un
hombre
dando
de
comer
a
los
patos
apoyado
en
el
cartel
de
prohibido
dar
de
comer
a
los
patos
quejandose
de
lo
gordos
que
estan


LETconectores.txt
si
luego
pero
y
o
aunque

comer
patos

```
tr = open("LETtext.txt")
tls = tr.readlines()
tr.close()

cr = open("LETconectores.txt")
cls = cr.readlines()
cr.close()
```

...

```
tr = open("LETtext.txt")
tls = tr.readlines()
tr.close()

cr = open("LETconectores.txt")
cls = cr.readlines()
cr.close()

#Crear list de str con conectores
#sin el ninja (\n)
cons = []
for cl in cls:
    cons.append(cl.strip())
```

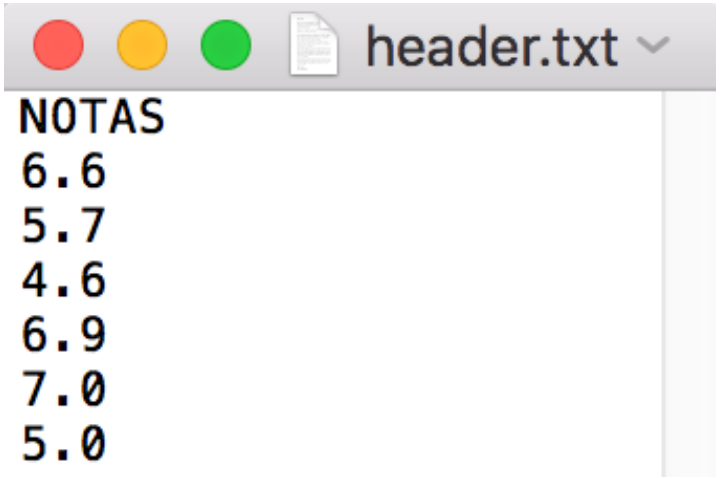
...

```
tr = open("LETtext.txt")
tls = tr.readlines()
tr.close()

cr = open("LETconectores.txt")
cls = cr.readlines()
cr.close()

#Crear list de str con conectores
#sin el ninja (\n)
cons = []
for cl in cls:
    cons.append(cl.strip())


unavez = []
repetidas = []
for tl in tls:
    p = tl.strip()
    if p not in cons:

        #Primera vez
        if p not in unavez:
            unavez.append(p)
        #Segunda vez
        elif p not in repetidas:
            repetidas.append(p)

for p in repetidas:
    print(p)
```

```
#Lista de palabras (inicialm
ps = []

#Pedir letra
let = input("Letra: ")
let = let.upper()

continuar = True
while continuar:

    #Pedir palabra
    p = input("Palabra: ")

    # En mayusculas, para est
    p = p.upper()

    if not (let in p):
        print("La palabra NO
        continuar = False

    elif p in ps:
        print("Has repetido",
        continuar = False

    else:
        print("Bien!")
        ps.append(p)

print("Juego acabado")
```

```
let = input("Letra:
let = let.upper()

continuar = True
while continuar:
    let
    let
```

HEADER (CABECERA)



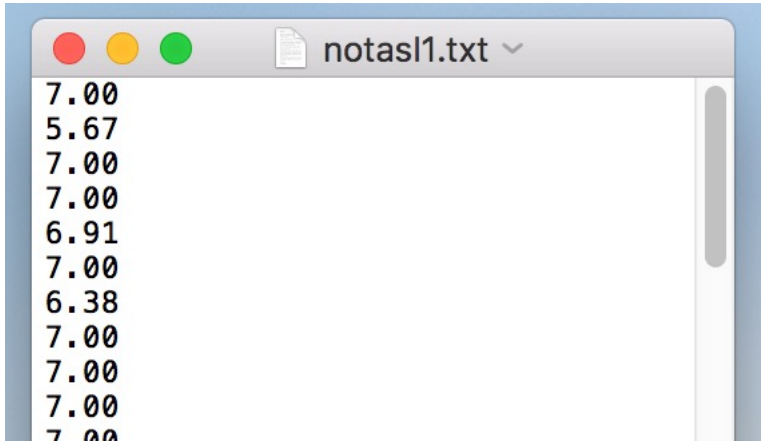

```
suma = 0
for l in ls:
    nota = float(l.strip())
    suma += nota

print("PROMEDIO",suma/len(ls))
```

```
e/publico/codigos/11-Archivos/header.py", line 25, in <module>
    nota = float(l.strip())
ValueError: could not convert string to float: 'NOTAS'
```

```
r = open("header.txt")
ls = r.readlines()
r.close()

# Es una lista normal!
# El header es el elem 0 de la lista
# Toda la list des del elemento 1 hasta el final
ls2 = ls[1:]

suma = 0
for l in ls2:
    nota = float(l.strip())
    suma += nota

print("PROMEDIO",suma/len(ls2))
```

```
PROMEDIO 5.68
```

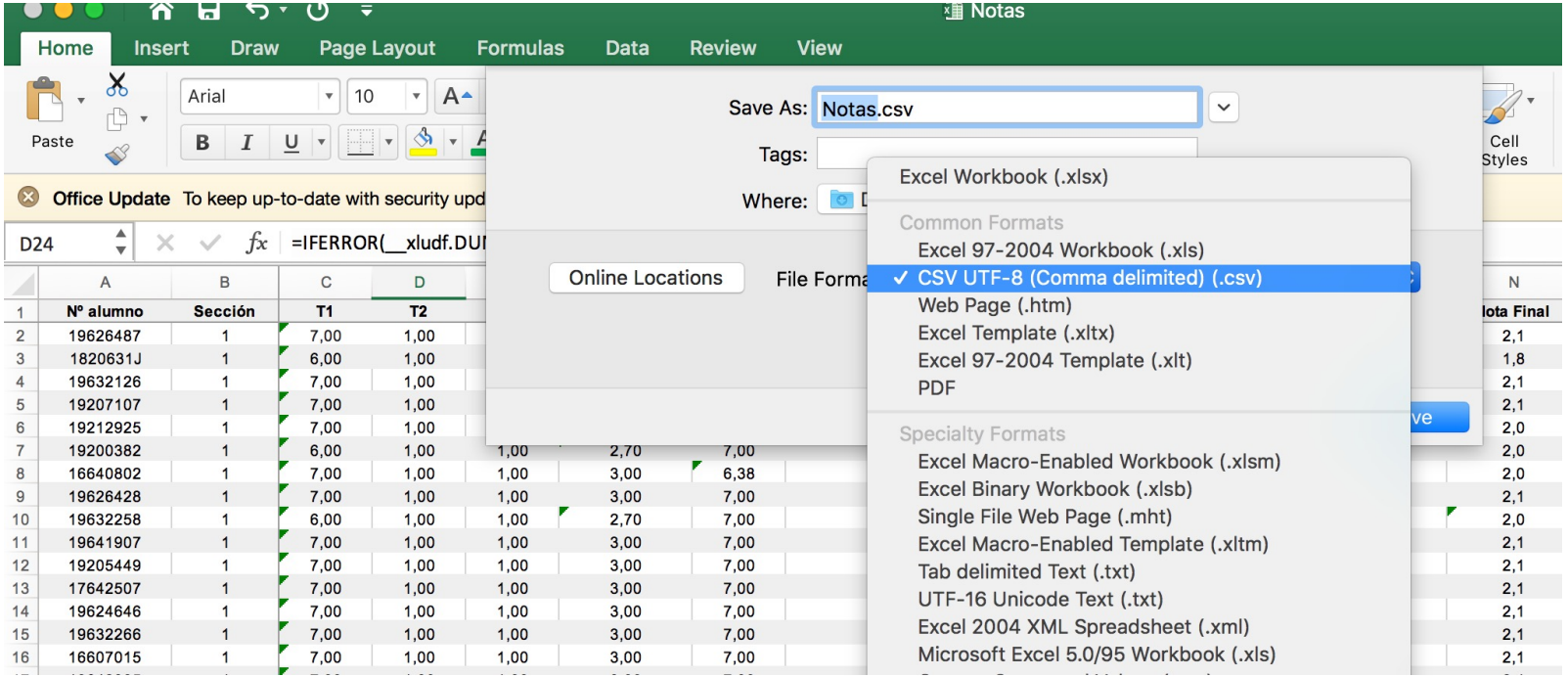
CSV


notasl1.txt
7.00
5.67
7.00
7.00
6.91
7.00
6.38
7.00
7.00
7.00

	A	B	C	D	E	F	G	H	I	J	K	L	M	N
1	**Nº alumno**	**Sección**	**T1**	**T2**	**T3**	**Tareas**	**I1**	**I2**	**Examen**	**Evaluaciones**	**Participación**		**Pre-Final**	**Nota Final**
2	19626487	1	7,00	1,00	1,00	3,00	7,00			1,80	1,00		2,05	2,1
3	1820631J	1	6,00	1,00	1,00	2,70	5,67			1,40	1,00		1,75	1,8
4	19632126	1	7,00	1,00	1,00	3,00	7,00			1,80	1,00		2,05	2,1
5	19207107	1	7,00	1,00	1,00	3,00	7,00			1,80	1,00		2,05	2,1
6	19212925	1	7,00	1,00	1,00	3,00	6,91			1,70	1,00		2,04	2,0
7	19200382	1	6,00	1,00	1,00	2,70	7,00			1,80	1,00		1,95	2,0
8	16640802	1	7,00	1,00	1,00	3,00	6,38			1,60	1,00		1,96	2,0
9	19626428	1	7,00	1,00	1,00	3,00	7,00			1,80	1,00		2,05	2,1
10	19632258	1	6,00	1,00	1,00	2,70	7,00			1,80	1,00		1,95	2,0
11	19641907	1	7,00	1,00	1,00	3,00	7,00			1,80	1,00		2,05	2,1
12	19205449	1	7,00	1,00	1,00	3,00	7,00			1,80	1,00		2,05	2,1
13	17642507	1	7,00	1,00	1,00	3,00	7,00			1,80	1,00		2,05	2,1
14	19624646	1	7,00	1,00	1,00	3,00	7,00			1,80	1,00		2,05	2,1
15	19632266	1	7,00	1,00	1,00	3,00	7,00			1,80	1,00		2,05	2,1
16	16607015	1	7,00	1,00	1,00	3,00	7,00			1,80	1,00		2,05	2,1
17	19642385	1	7,00	1,00	1,00	3,00	7,00			1,80	1,00		2,05	2,1
18	19626290	1	7,00	1,00	1,00	3,00	7,00			1,80	1,00		2,05	2,1
19	1820645J	1	7,00	1,00	1,00	3,00	6,33			1,60	1,00		1,95	2,0
20	19625197	1	7,00	1,00	1,00	3,00	6,46			1,60	1,00		1,97	2,0
21	1863897J	1	7,00	1,00	1,00	3,00	7,00			1,80	1,00		2,05	2,1

Notas.csv

```
Nº alumno;Sección;T1;T2;T3;Tareas;I1;I2;Examen;Evaluaciones;Participación;;Pre-Final;Nota Final
19626487;1;7,00;1,00;1,00;3,00;7,00;;;1,80;1,00;;2,05;2,1
1820631J;1;6,00;1,00;1,00;2,70;5,67;;;1,40;1,00;;1,75;1,8
19632126;1;7,00;1,00;1,00;3,00;7,00;;;1,80;1,00;;2,05;2,1
19207107;1;7,00;1,00;1,00;3,00;7,00;;;1,80;1,00;;2,05;2,1
19212925;1;7,00;1,00;1,00;3,00;6,91;;;1,70;1,00;;2,04;2,0
19200382;1;6,00;1,00;1,00;2,70;7,00;;;1,80;1,00;;1,95;2,0
16640802;1;7,00;1,00;1,00;3,00;6,38;;;1,60;1,00;;1,96;2,0
19626428;1;7,00;1,00;1,00;3,00;7,00;;;1,80;1,00;;2,05;2,1
19632258;1;6,00;1,00;1,00;2,70;7,00;;;1,80;1,00;;1,95;2,0
19641907;1;7,00;1,00;1,00;3,00;7,00;;;1,80;1,00;;2,05;2,1
19205449;1;7,00;1,00;1,00;3,00;7,00;;;1,80;1,00;;2,05;2,1
17642507;1;7,00;1,00;1,00;3,00;7,00;;;1,80;1,00;;2,05;2,1
19624646;1;7,00;1,00;1,00;3,00;7,00;;;1,80;1,00;;2,05;2,1
19632266;1;7,00;1,00;1,00;3,00;7,00;;;1,80;1,00;;2,05;2,1
```



	A	B	C	D	N
1	Nº alumno	Sección	T1	T2	Nota Final
2	19626487	1	7,00	1,00	2,1
3	1820631J	1	6,00	1,00	1,8
4	19632126	1	7,00	1,00	2,1
5	19207107	1	7,00	1,00	2,1
6	19212925	1	7,00	1,00	2,0
7	19200382	1	6,00	1,00	2,0
8	16640802	1	7,00	1,00	2,0
9	19626428	1	7,00	1,00	2,1
10	19632258	1	6,00	1,00	2,0
11	19641907	1	7,00	1,00	2,1
12	19205449	1	7,00	1,00	2,1
13	17642507	1	7,00	1,00	2,1
14	19624646	1	7,00	1,00	2,1
15	19632266	1	7,00	1,00	2,1
16	16607015	1	7,00	1,00	2,1

Notas.csv

```
Nº alumno;Sección;T1;T2;T3;Tareas;I1;I2;Examen;Evaluaciones;Participación;;Pre-Final;Nota Final
19626487;1;7,00;1,00;1,00;3,00;7,00;;;1,80;1,00;;2,05;2,1
1820631J;1;6,00;1,00;1,00;2,70;5,67;;;1,40;1,00;;1,75;1,8
19632126;1;7,00;1,00;1,00;3,00;7,00;;;1,80;1,00;;2,05;2,1
19207107;1;7,00;1,00;1,00;3,00;7,00;;;1,80;1,00;;2,05;2,1
19212925;1;7,00;1,00;1,00;3,00;6,91;;;1,70;1,00;;2,04;2,0
19200382;1;6,00;1,00;1,00;2,70;7,00;;;1,80;1,00;;1,95;2,0
16640802;1;7,00;1,00;1,00;3,00;6,38;;;1,60;1,00;;1,96;2,0
19626428;1;7,00;1,00;1,00;3,00;7,00;;;1,80;1,00;;2,05;2,1
19632258;1;6,00;1,00;1,00;2,70;7,00;;;1,80;1,00;;1,95;2,0
19641907;1;7,00;1,00;1,00;3,00;7,00;;;1,80;1,00;;2,05;2,1
19205449;1;7,00;1,00;1,00;3,00;7,00;;;1,80;1,00;;2,05;2,1
17642507;1;7,00;1,00;1,00;3,00;7,00;;;1,80;1,00;;2,05;2,1
19624646;1;7,00;1,00;1,00;3,00;7,00;;;1,80;1,00;;2,05;2,1
19632266;1;7,00;1,00;1,00;3,00;7,00;;;1,80;1,00;;2,05;2,1
```

Notas

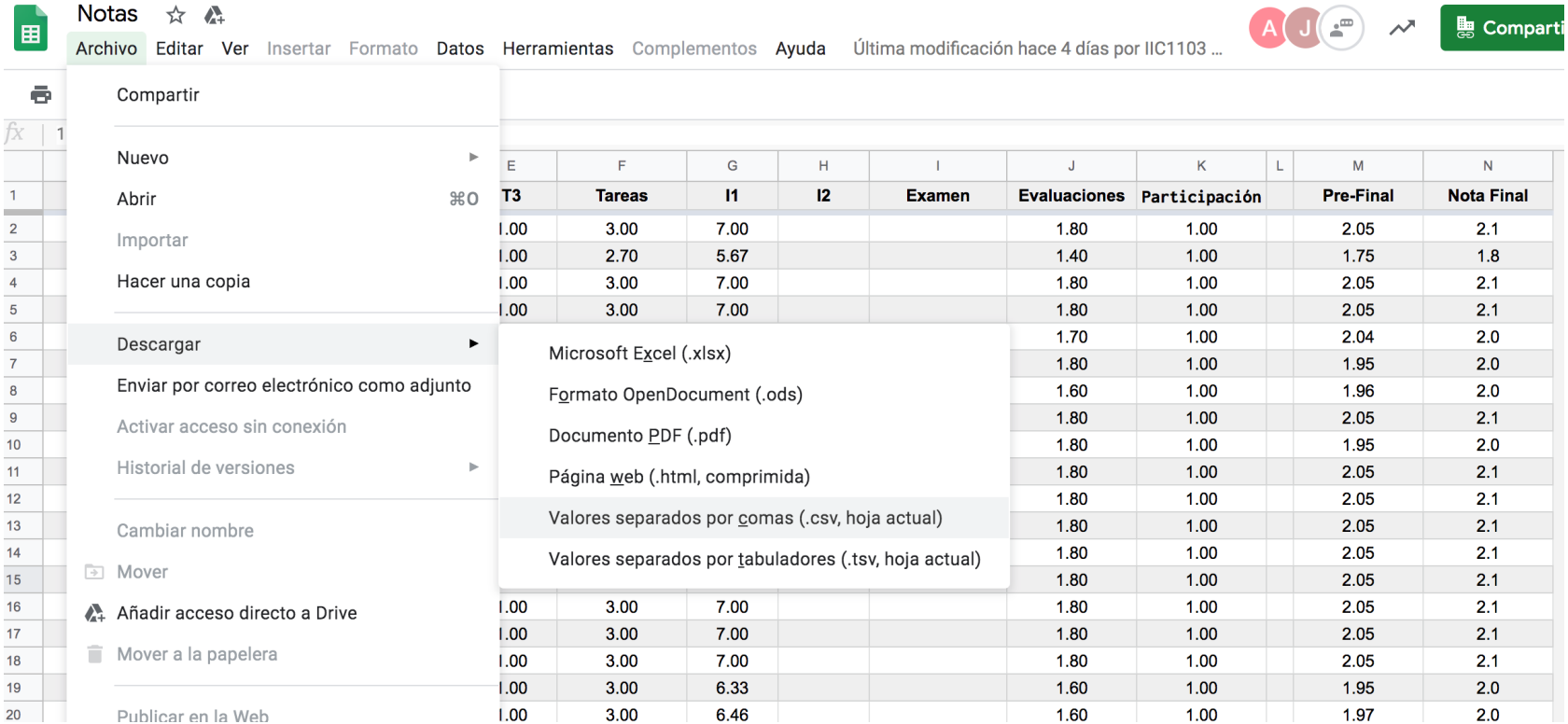
Archivo Editar Ver Insertar Formato Datos Herramientas Complementos Ayuda Última modificación hace 4 días por IIC1103 ...

Compartir

100% Solo lectura

fx 1

	A	B	C	D	E	F	G	H	I	J	K	L	M	N
1	Nº alumno	Sección	T1	T2	T3	Tareas	I1	I2	Examen	Evaluaciones	Participación		Pre-Final	Nota Final
2	19626487	1	7.00	1.00	1.00	3.00	7.00			1.80	1.00		2.05	2.1
3	1820631J	1	6.00	1.00	1.00	2.70	5.67			1.40	1.00		1.75	1.8
4	19632126	1	7.00	1.00	1.00	3.00	7.00			1.80	1.00		2.05	2.1
5	19207107	1	7.00	1.00	1.00	3.00	7.00			1.80	1.00		2.05	2.1
6	19212925	1	7.00	1.00	1.00	3.00	6.91			1.70	1.00		2.04	2.0
7	19200382	1	6.00	1.00	1.00	2.70	7.00			1.80	1.00		1.95	2.0
8	16640802	1	7.00	1.00	1.00	3.00	6.38			1.60	1.00		1.96	2.0
9	19626428	1	7.00	1.00	1.00	3.00	7.00			1.80	1.00		2.05	2.1
10	19632258	1	6.00	1.00	1.00	2.70	7.00			1.80	1.00		1.95	2.0
11	19641907	1	7.00	1.00	1.00	3.00	7.00			1.80	1.00		2.05	2.1
12	19205449	1	7.00	1.00	1.00	3.00	7.00			1.80	1.00		2.05	2.1
13	17642507	1	7.00	1.00	1.00	3.00	7.00			1.80	1.00		2.05	2.1
14	19624646	1	7.00	1.00	1.00	3.00	7.00			1.80	1.00		2.05	2.1
15	19632266	1	7.00	1.00	1.00	3.00	7.00			1.80	1.00		2.05	2.1
16	16607015	1	7.00	1.00	1.00	3.00	7.00			1.80	1.00		2.05	2.1
17	19642385	1	7.00	1.00	1.00	3.00	7.00			1.80	1.00		2.05	2.1



	E	F	G	H	I	J	K	L	M	N
1	T3	Tareas	I1	I2	Examen	Evaluaciones	Participación		Pre-Final	Nota Final
2	1.00	3.00	7.00			1.80	1.00		2.05	2.1
3	1.00	2.70	5.67			1.40	1.00		1.75	1.8
4	1.00	3.00	7.00			1.80	1.00		2.05	2.1
5	1.00	3.00	7.00			1.80	1.00		2.05	2.1
6						1.70	1.00		2.04	2.0
7						1.80	1.00		1.95	2.0
8						1.60	1.00		1.96	2.0
9						1.80	1.00		2.05	2.1
10						1.80	1.00		1.95	2.0
11						1.80	1.00		2.05	2.1
12						1.80	1.00		2.05	2.1
13						1.80	1.00		2.05	2.1
14						1.80	1.00		2.05	2.1
15						1.80	1.00		2.05	2.1
16	1.00	3.00	7.00			1.80	1.00		2.05	2.1
17	1.00	3.00	7.00			1.80	1.00		2.05	2.1
18	1.00	3.00	7.00			1.80	1.00		2.05	2.1
19	1.00	3.00	6.33			1.60	1.00		1.95	2.0
20	1.00	3.00	6.46			1.60	1.00		1.97	2.0

Notas - Resumen.csv

```
Nº alumno,Sección,T1,T2,T3,Tareas,I1,I2,Examen,Evaluaciones,Participación,,Pre-Final,Nota Final
19626487,1,7.00,1.00,1.00,3.00,7.00,,,1.80,1.00,,2.05,2.1
1820631J,1,6.00,1.00,1.00,2.70,5.67,,,1.40,1.00,,1.75,1.8
19632126,1,7.00,1.00,1.00,3.00,7.00,,,1.80,1.00,,2.05,2.1
19207107,1,7.00,1.00,1.00,3.00,7.00,,,1.80,1.00,,2.05,2.1
19212925,1,7.00,1.00,1.00,3.00,6.91,,,1.70,1.00,,2.04,2.0
19200382,1,6.00,1.00,1.00,2.70,7.00,,,1.80,1.00,,1.95,2.0
16640802,1,7.00,1.00,1.00,3.00,6.38,,,1.60,1.00,,1.96,2.0
19626428,1,7.00,1.00,1.00,3.00,7.00,,,1.80,1.00,,2.05,2.1
19632258,1,6.00,1.00,1.00,2.70,7.00,,,1.80,1.00,,1.95,2.0
19641907,1,7.00,1.00,1.00,3.00,7.00,,,1.80,1.00,,2.05,2.1
19205449,1,7.00,1.00,1.00,3.00,7.00,,,1.80,1.00,,2.05,2.1
17642507,1,7.00,1.00,1.00,3.00,7.00,,,1.80,1.00,,2.05,2.1
19624646,1,7.00,1.00,1.00,3.00,7.00,,,1.80,1.00,,2.05,2.1
```

Bibliografía & Investigación

Base de Datos COVID-19

INICIO | Submesa de Datos | Visualizador GOB.CL | ¿Cómo es usada la BBDD COVID-19? | Datos por terceros

Para apoyar la investigación científica, clínica y epidemiológica del COVID19 en Chile, ponemos a disposición de la comunidad este recurso que reúne la información oficial del Ministerio de Salud en un formato estándar para su análisis.

Esta Base de Datos es resultado de la submesa de datos, que nace al alero de la Mesa Social COVID19, con el objetivo de facilitar el trabajo de todos y todas quienes busquen aportar con soluciones a esta emergencia sanitaria a través del análisis de información.

(*) Link a *GitHub* del Ministerio de Ciencia, Tecnología, Conocimiento e Innovación, donde se encuentran todos los datos en la lista a continuación (La automatización de este proceso y disposición de datos es desarrollada por el equipo del Data Observatory y ha sido complementada por la comunidad).

(**) Link a la licencia que otorga el Ministerio de Ciencia, Tecnología, Conocimiento e Innovación sobre las bases de datos incluidas en GitHub.

Datos Nacionales

Totales nacionales incremental diario: archivo con valores separados por coma (csv) con casos totales, nuevos, activos, recuperados y fallecidos totales con el valor diario reportado por el MINSAL. Contiene los campos 'Casos Nuevos', 'Casos totales', 'Casos nevos', 'Casos Activos por Fecha de Diagnostico', 'Casos activos por Fecha de inicio de síntomas', 'Casos Recuperados por fecha de diagnóstico', 'Recuperados por Fecha de Inicio de Sintomas', 'Fallecidos', '[fecha]', donde la última columna tiene los valores reportados diariamente por MINSAL. Nota: Casos activos en este reporte (a diferencia del reporte en 16) corresponde al resultado de restar fallecidos y personas recuperadas al total de casos diagnosticados. Las personas recuperadas por fecha de diagnostico son casos que tras ser confirmados, han estado en cuarentena pasando 14 días sin síntomas, por fecha de inicio de síntomas son casos que han pasado 14 días desde el inicio de su cuadro clínico. Ver más.

http://www.minciencia.gob.cl/covid19

Branch: master ▾ Datos-COVID19 / output / producto2 / 2020-06-08-CasosConfirmados.csv

Find file | Copy path

actions-user [create-pull-request] automated change c5ac626 4 days ago

1 contributor

347 lines (347 sloc) | 14.3 KB

Raw | Blame | History

Search this file...

	Region	Codigo region	Comuna	Codigo comuna	Poblacion	Casos Confirmados
2	Arica y Parinacota	15	Arica	15101	247552.0	904.0
3	Arica y Parinacota	15	Camarones	15102	1233.0	0.0
4	Arica y Parinacota	15	General Lagos	15202	810.0	0.0
5	Arica y Parinacota	15	Putre	15201	2515.0	1.0
6	Tarapacá	01	Alto Hospicio	01107	129999.0	1238.0
7	Tarapacá	01	Camina	01402	1375.0	10.0
8	Tarapacá	01	Colchane	01403	1583.0	5.0
9	Tarapacá	01	Huara	01404	3000.0	37.0
10	Tarapacá	01	Iquique	01101	223463.0	1904.0
11	Tarapacá	01	Pica	01405	5958.0	59.0
12	Tarapacá	01	Pozo Almonte	01401	17395.0	121.0
13	Antofagasta	02	Antofagasta	02101	425725.0	1876.0
14	Antofagasta	02	Calama	02201	190336.0	756.0
15	Antofagasta	02	Maria Elena	02302	6814.0	168.0
16	Antofagasta	02	Mejillones	02102	14776.0	279.0

http://www.minciencia.gob.cl/covid19

```
Region,Codigo region,Comuna,Codigo comuna,Poblacion,Casos Confirmados
Arica y Parinacota,15,Arica,15101,247552.0,904.0
Arica y Parinacota,15,Camarones,15102,1233.0,0.0
Arica y Parinacota,15,General Lagos,15202,810.0,0.0
Arica y Parinacota,15,Putre,15201,2515.0,1.0
Tarapacá,01,Alto Hospicio,01107,129999.0,1238.0
Tarapacá,01,Camina,01402,1375.0,10.0
Tarapacá,01,Colchane,01403,1583.0,5.0
Tarapacá,01,Huara,01404,3000.0,37.0
Tarapacá,01,Iquique,01101,223463.0,1904.0
Tarapacá,01,Pica,01405,5958.0,59.0
Tarapacá,01,Pozo Almonte,01401,17395.0,121.0
Antofagasta,02,Antofagasta,02101,425725.0,1876.0
Antofagasta,02,Calama,02201,190336.0,756.0
Antofagasta,02,Maria Elena,02302,6814.0,168.0
Antofagasta,02,Mejillones,02102,14776.0,279.0
Antofagasta,02,Ollague,02202,287.0,0.0
Antofagasta,02,San Pedro de Atacama,02203,10434.0,11.0
Antofagasta,02,Sierra Gorda,02103,1746.0,21.0
Antofagasta,02,Taltal,02104,13657.0,37.0
Antofagasta,02,Tocopilla,02301,28079.0,38.0
Atacama,03,Alto del Carmen,03302,5729.0,2.0
Atacama,03,Caldera,03102,19426.0,10.0
Atacama,03,Chanaral,03201,13164.0,1.0
Atacama,03,Copiapo,03101,171766.0,207.0
Atacama,03,Diego de Almagro,03202,14358.0,15.0
Atacama,03,Freirina,03303,7681.0,1.0
Atacama,03,Huasco,03304,11264.0,5.0
Atacama,03,Tierra Amarilla,03103,14312.0,26.0
Atacama,03,Vallenar,03301,57009.0,80.0
Coquimbo,04,Andacollo,04103,11791.0,19.0
```

SPLIT

```
Region,Codigo region,Comuna,Codigo comuna,Poblacion,Casos Confirmados
Arica y Parinacota,15,Arica,15101,247552.0,904.0
Arica y Parinacota,15,Camarones,15102,1233.0,0.0
Arica y Parinacota,15,General Lagos,15202,810.0,0.0
Arica y Parinacota,15,Putre,15201,2515.0,1.0
Tarapacá,01,Alto Hospicio,01107,129999.0,1238.0
Tarapacá,01,Camina,01402,1375.0,10.0
```

```
s = "Tarapaca,01,Camina,01402,1375.0,10.0"
```

```
...
```

```
print(cs)
```

```
['Tarapaca', '01', 'Camina', '01402', '1375.0', '10.0']
```

```
Region,Codigo region,Comuna,Codigo comuna,Poblacion,Casos Confirmados
Arica y Parinacota,15,Arica,15101,247552.0,904.0
Arica y Parinacota,15,Camarones,15102,1233.0,0.0
Arica y Parinacota,15,General Lagos,15202,810.0,0.0
Arica y Parinacota,15,Putre,15201,2515.0,1.0
Tarapacá,01,Alto Hospicio,01107,129999.0,1238.0
Tarapacá,01,Camina,01402,1375.0,10.0
```

```python
s = "Tarapaca,01,Camina,01402,1375.0,10.0"

#Parte un str en list de srt usando la ','
cs = s.split(",")
#SIEMPRE es list de STR

print(cs)
```

```
['Tarapaca', '01', 'Camina', '01402', '1375.0', '10.0']
```

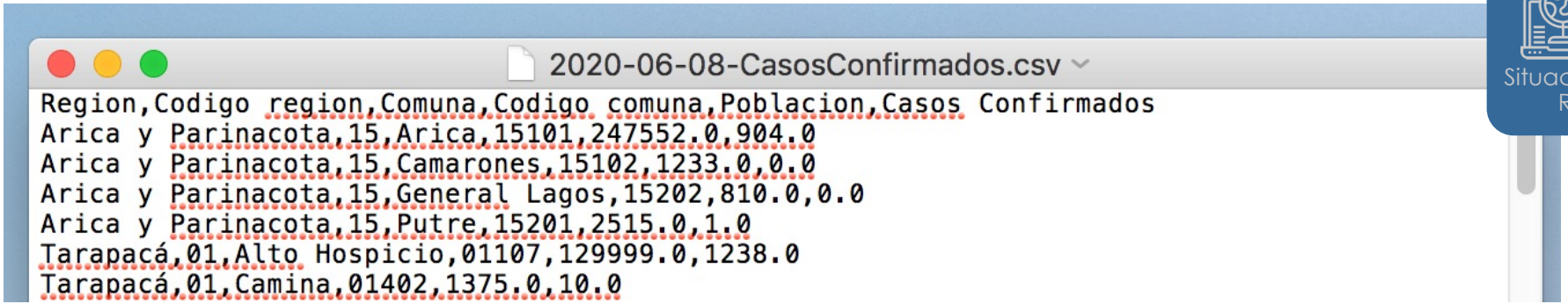
DATOS COVID

INTERESES

“COVID”

“Expansión de las enfermedades”

“Actualidad”



Situación Real

Branch: master | Datos-COVID19 / output / producto2 / 2020-06-08-CasosConfirmados.csv | Find file | Copy path

actions-user [create-pull-request] automated change c5ac626 4 days ago

1 contributor

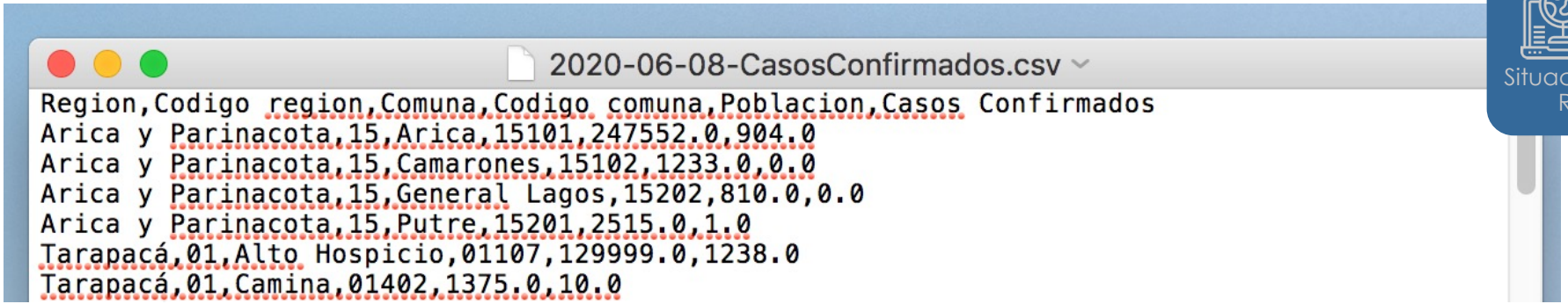
347 lines (347 sloc) | 14.3 KB | Raw | Blame | History

Search this file...

	Region	Codigo region	Comuna	Codigo comuna	Poblacion	Casos Confirmados
2	Arica y Parinacota	15	Arica	15101	247552.0	904.0
3	Arica y Parinacota	15	Camarones	15102	1233.0	0.0
4	Arica y Parinacota	15	General Lagos	15202	810.0	0.0
5	Arica y Parinacota	15	Putre	15201	2515.0	1.0
6	Tarapacá	01	Alto Hospicio	01107	129999.0	1238.0

```
Codigo Region? 15
Region 15 - 905.0
```


I'LL WAIT
FOR YOU HERE



Situación Real

Branch: master | Datos-COVID19 / output / producto2 / 2020-06-08-CasosConfirmados.csv | Find file | Copy path

actions-user [create-pull-request] automated change c5ac626 4 days ago

1 contributor

347 lines (347 sloc) | 14.3 KB | Raw | Blame | History

Search this file...

	Region	Codigo region	Comuna	Codigo comuna	Poblacion	Casos Confirmados
2	Arica y Parinacota	15	Arica	15101	247552.0	904.0
3	Arica y Parinacota	15	Camarones	15102	1233.0	0.0
4	Arica y Parinacota	15	General Lagos	15202	810.0	0.0
5	Arica y Parinacota	15	Putre	15201	2515.0	1.0
6	Tarapacá	01	Alto Hospicio	01107	129999.0	1238.0

```
Codigo Region? 15
Region 15 - 905.0
```

```
r = open("2020-06-08-CasosConfirmados.csv", encoding="utf-8")
ls = r.readlines()
r.close()

region = input("Codigo Region? ")

suma = 0
for l in ls:
    cs = l.strip().split(",")
    if cs[1] == region:
        casos = float(cs[5])
        suma += casos

print("Region",region,"-",suma),
```

```
...
Codigo Region? 15
Region 15 - 905.0
```

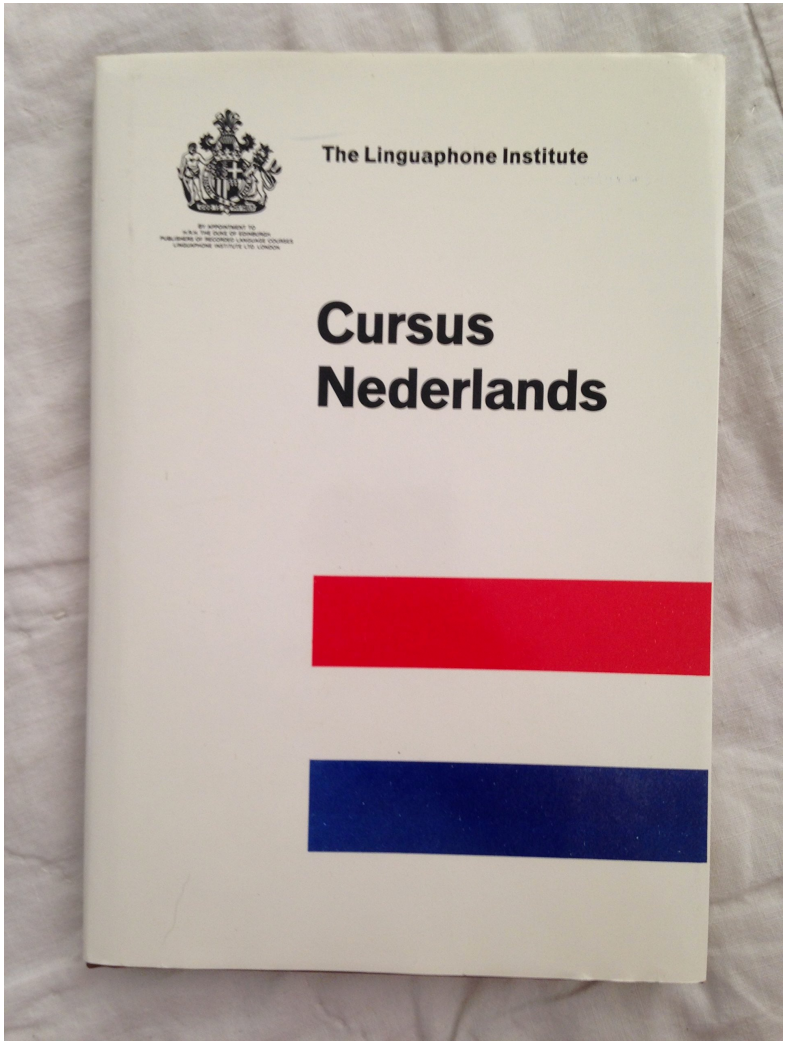
APRENDIENDO IDIOMAS

INTERESES

“inglés”

“me gustaría aprender catalán”

“aprender un idioma nuevo”



beroemde	famous
deas	dollar
stroom	stream
angst	fear
zicht	sight
dun	thin
driehoek	triangle
planeet	planet
haast	hurry
chief	chief
kolonie	colony
klok	clock
de mijne	mine
stropdas	tie
voeren	enter
grote	major
vers	fresh
zoeken	search

	A	B	C	D
1	CAT	ESP	ENG	FRA
2	cadira	silla	chair	chaise
3	llit	cama	bed	lit
4	porta	puerta	door	porte
5	fruita	fruta	fruit	fruit
6	poma	manzana	apple	pomme
7	casa	casa	house	maison
8	orella	oreja	ear	oreille
9	sol	sol	sun	soleil

idiomas.csv

```
CAT;ESP;ENG;FRA
cadira;silla;chair;chaise
llit;cama;bed;lit
porta;puerta;door;porte
fruita;fruta;fruit;fruit
poma;manzana;apple;pomme
casa;casa;house;maison
orella;oreja;ear;oreille
sol;sol;sun;soleil
```

```
0 <- CAT
1 <- ESP
2 <- ENG
3 <- FRA
De ...? 1
A ...? 0
fruta
fruita
silla
cadira
manzana
manzan
ERROR -  poma
puerta
porta
silla
EXIT
```


I'LL WAIT
FOR YOU HERE

	A	B	C	D
1	CAT	ESP	ENG	FRA
2	cadira	silla	chair	chaise
3	llit	cama	bed	lit
4	porta	puerta	door	porte
5	fruita	fruta	fruit	fruit
6	poma	manzana	apple	pomme
7	casa	casa	house	maison
8	orella	oreja	ear	oreille
9	sol	sol	sun	soleil

idiomas.csv

```
CAT;ESP;ENG;FRA
cadira;silla;chair;chaise
llit;cama;bed;lit
porta;puerta;door;porte
fruita;fruta;fruit;fruit
poma;manzana;apple;pomme
casa;casa;house;maison
orella;oreja;ear;oreille
sol;sol;sun;soleil
```

```
0 <- CAT
1 <- ESP
2 <- ENG
3 <- FRA
De ...? 1
A ...? 0
fruta
fruita
silla
cadira
manzana
manzan
ERROR -  poma
puerta
porta
silla
EXIT
```

```
import random

r = open("idiomas.csv")
ls = r.readlines()
r.close()
```

...

```
import random

r = open("idiomas.csv")
ls = r.readlines()
r.close()

#Que elija los idiomas
idiomas = ls[0].strip().split(";")
for i in range(0,len(idiomas)):
    print(i,"<-",idiomas[i])
de = int(input("De ...? "))
a = int(input("A ...? "))
```

...

```
import random

r = open("idiomas.csv")
ls = r.readlines()
r.close()

#Que elija los idiomas
idiomas = ls[0].strip().split(";")
for i in range(0,len(idiomas)):
    print(i,"<-",idiomas[i])
de = int(input("De ...? "))
a = int(input("A ...? "))


continuar = True
while continuar:
    #Palabra al azar
    az = random.randint(1,len(ls)-1)
    cs = ls[az].strip().split(";")
    pde = cs[de]
    pa = cs[a]

    print(pde)
    res = input()

    if res == "EXIT":
        continuar = False
    elif res != pa:
        print("ERROR - ",pa)
```

MEJORAS

Llevar estadísticas de acierto y de aparición

Que salgan las palabras que más has fallado y que más tiempo hace que no salen

Hacer niveles (como de 10 palabras) e ir pasando de nivel

…

ASISTENTES A LA FIESTA

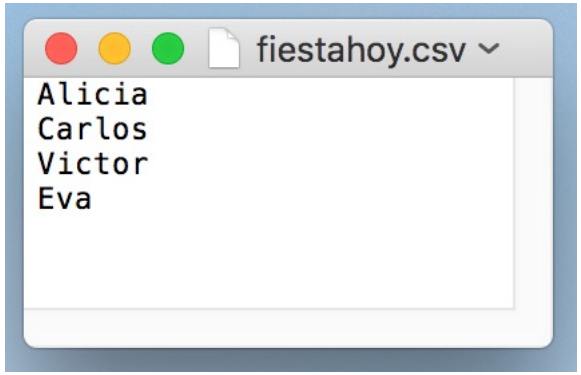
INTERESES

“organizar eventos”

Estimado Jorge:

¿Cómo estás? Espero que se encuentre muy bien en estos días. Soy [redacted], ex-alumno de usted en Introducción a la Programación (el que le dijo el programa para calcular la asistencia a una fiesta si le sirve para recordar jajaja).

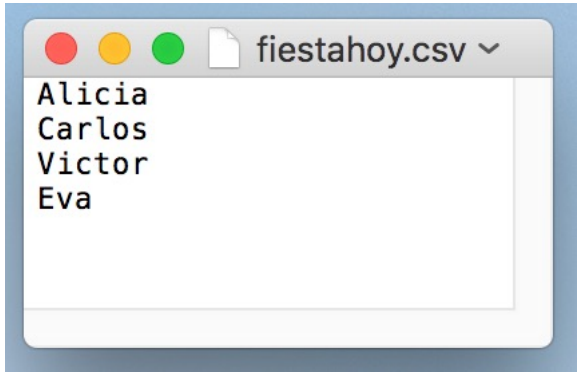
	A	B	C	D	E	F	G	H	I	J	K	L
1	Alicia	X	X	X	X	X	X	X	X			
2	Bob	-	X	-	-	-	-					
3	Carlos	-	X									
4	David	X	X	X	X	-	-	X				
5	Eva	-	-	-	-	-	-	-	-	-	-	-



Num previsto: 2.5


I'LL WAIT
FOR YOU HERE

	A	B	C	D	E	F	G	H	I	J	K	L
1	Alicia	X	X	X	X	X	X	X	X			
2	Bob	-	X	-	-	-	-					
3	Carlos	-	X									
4	David	X	X	X	X	-	-	X				
5	Eva	-	-	-	-	-	-	-	-	-	-	-



Num previsto: 2.5

```
er = open("fiestaest.csv")
els = er.readlines()
er.close()

hr = open("fiestahoy.csv")
hls = hr.readlines()
hr.close()

#Calcular estadisticas
stats = []
for l in els:
    cs = l.strip().split(";")
    n = cs[0]
    tics = cs[1:]
    suma = 0
    for tic in tics:
        if tic == "X":
            suma += 1
    pro = suma / len(tics)
    stat = [n,pro]
    stats.append(stat)
```

...

...

```
er = open("fiestaest.csv")
els = er.readlines()
er.close()

hr = open("fiestahoy.csv")
hls = hr.readlines()
hr.close()

#Calcular estadisticas
stats = []
for l in els:
    cs = l.strip().split(";")
    n = cs[0]
    tics = cs[1:]
    suma = 0
    for tic in tics:
        if tic == "X":
            suma += 1
    pro = suma / len(tics)
    stat = [n,pro]
    stats.append(stat)
```

```
#Calcular el numero de personas
res = 0
for l in hls:
    n = l.strip()

    #Buscar si esta
    esta = False
    for stat in stats:
        if stat[0] == n:
            res += stat[1]
            esta = True

    if not esta:
        res += 1

print("Num previsto:",res)
```

NBA (Parte 1)

INTERESES

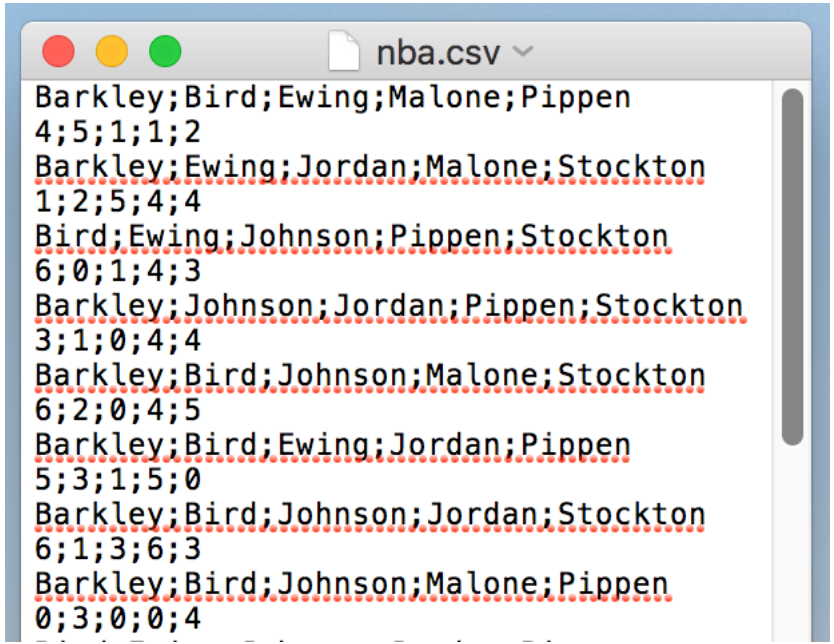
“deportes”

“Videojuegos, baloncesto y física”

“Voluntariados, basketball, música”

Situación Real

	A	B	C	D	E
1	Barkley	Bird	Ewing	Malone	Pippen
2	4	5	1	1	2
3	Barkley	Ewing	Jordan	Malone	Stockton
4	1	2	5	4	4
5	Bird	Ewing	Johnson	Pippen	Stockton
6	6	0	1	4	3
7	Barkley	Johnson	Jordan	Pippen	Stockton
8	3	1	0	4	4
9	Barkley	Bird	Johnson	Malone	Stockton
10	6	2	0	4	5



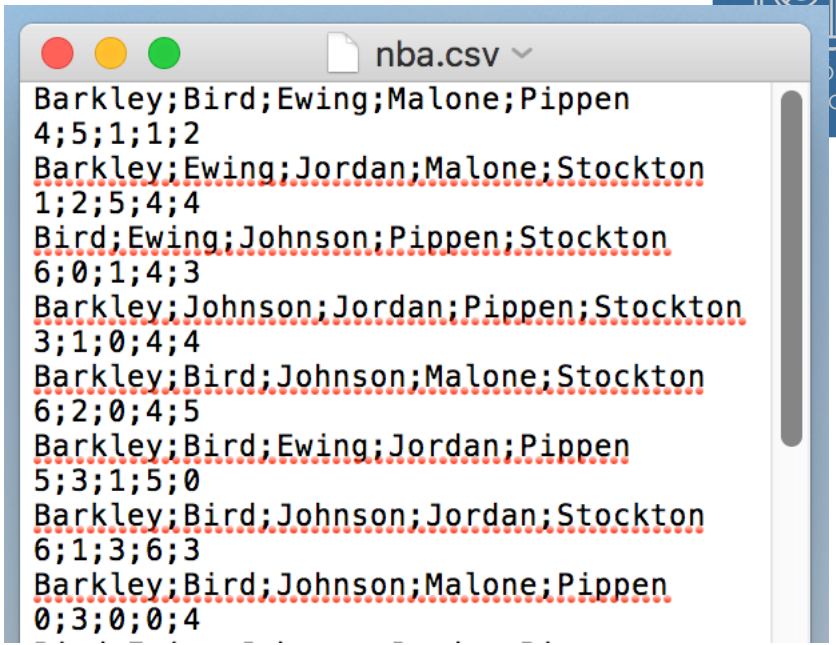
123 Barkley

Situación Real


BARCELONA'92
USA
13
5
9
12
11
15
6
7


I'LL WAIT
FOR YOU HERE

	A	B	C	D	E
1	Barkley	Bird	Ewing	Malone	Pippen
2	4	5	1	1	2
3	Barkley	Ewing	Jordan	Malone	Stockton
4	1	2	5	4	4
5	Bird	Ewing	Johnson	Pippen	Stockton
6	6	0	1	4	3
7	Barkley	Johnson	Jordan	Pippen	Stockton
8	3	1	0	4	4
9	Barkley	Bird	Johnson	Malone	Stockton
10	6	2	0	4	5



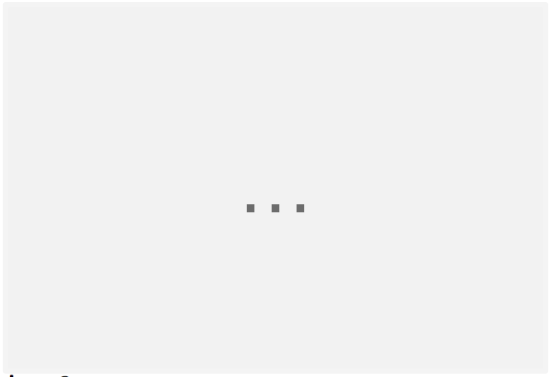
El jugador que más estrellas tiene

```
123 Barkley
```

Resolución Problema

	A	B	C	D	E
1	Barkley	Bird	Ewing	Malone	Pippen
2	4	5	1	1	2
3	Barkley	Ewing	Jordan	Malone	Stockton
4	1	2	5	4	4
5	Bird	Ewing	Johnson	Pippen	Stockton
6	6	0	1	4	3
7	Barkley	Johnson	Jordan	Pippen	Stockton
8	3	1	0	4	4
9	Barkley	Bird	Johnson	Malone	Stockton
10	6	2	0	4	5

```
r = open("nba.csv")
ls = r.readlines()
r.close()

#List de [n,s] con el nombre jugador
#y la suma de sus estrellas
ss = []
i = 0
while i < len(ls):
    pcs = ls[i].strip().split(";")
    ics = ls[i+1].strip().split(";")
```



```
    i+=2


#Maximo
maxn = -1
maxp = ""
for s in ss:
    if s[1] > maxn:
        maxp = s[0]
        maxn = s[1]

print(maxn,maxp)
```

Resolución Problema

	A	B	C	D	E
1	Barkley	Bird	Ewing	Malone	Pippen
2	4	5	1	1	2
3	Barkley	Ewing	Jordan	Malone	Stockton
4	1	2	5	4	4
5	Bird	Ewing	Johnson	Pippen	Stockton
6	6	0	1	4	3
7	Barkley	Johnson	Jordan	Pippen	Stockton
8	3	1	0	4	4
9	Barkley	Bird	Johnson	Malone	Stockton
10	6	2	0	4	5

```
r = open("nba.csv")
ls = r.readlines()
r.close()

#List de [n,s] con el nombre jugador
#y la suma de sus estrellas
ss = []
i = 0
while i < len(ls):
    pcs = ls[i].strip().split(";")
    ics = ls[i+1].strip().split(";")

    for j in range(0,5):
        nom = pcs[j]
        star = int(ics[j])

        esta = False
        for s in ss:
            if s[0] == nom:
                s[1] += star
                esta = True

        if not esta:
            ss.append([nom,star])
    i+=2


#Maximo
maxn = -1
maxp = ""
for s in ss:
    if s[1] > maxn:
        maxp = s[0]
        maxn = s[1]

print(maxn,maxp)
```

NBA (Parte 2)

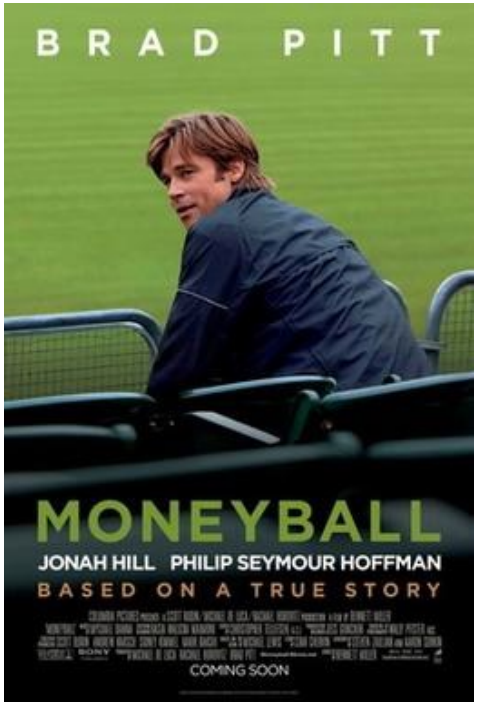
INTERESES

“deportes”

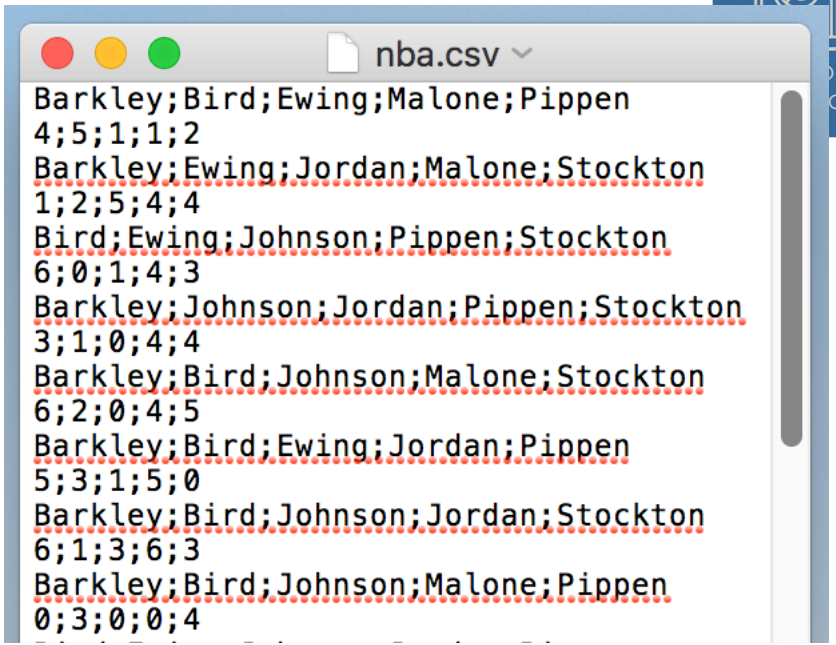
“Videojuegos, baloncesto y física”

“Voluntariados, basketball, música”


EXIT


BRAD PITT
MONEYBALL
JONAH HILL PHILIP SEYMOUR HOFFMAN
BASED ON A TRUE STORY
COMING SOON

	A	B	C	D	E
1	Barkley	Bird	Ewing	Malone	Pippen
2	4	5	1	1	2
3	Barkley	Ewing	Jordan	Malone	Stockton
4	1	2	5	4	4
5	Bird	Ewing	Johnson	Pippen	Stockton
6	6	0	1	4	3
7	Barkley	Johnson	Jordan	Pippen	Stockton
8	3	1	0	4	4
9	Barkley	Bird	Johnson	Malone	Stockton
10	6	2	0	4	5

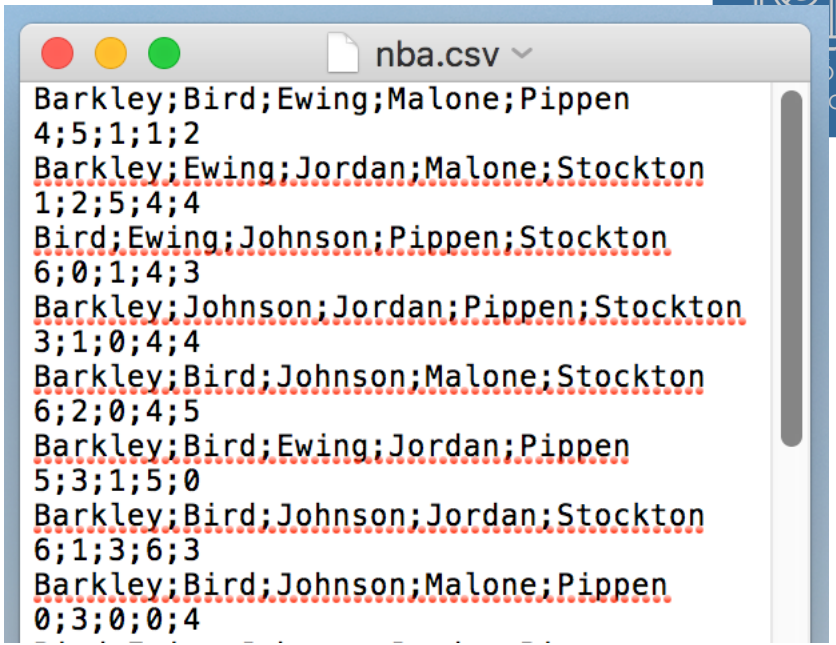


El jugador que más estrellas consigue para sus compañeros

```
445 Bird
```


I'LL WAIT
FOR YOU HERE

	A	B	C	D	E
1	Barkley	Bird	Ewing	Malone	Pippen
2	4	5	1	1	2
3	Barkley	Ewing	Jordan	Malone	Stockton
4	1	2	5	4	4
5	Bird	Ewing	Johnson	Pippen	Stockton
6	6	0	1	4	3
7	Barkley	Johnson	Jordan	Pippen	Stockton
8	3	1	0	4	4
9	Barkley	Bird	Johnson	Malone	Stockton
10	6	2	0	4	5



El jugador que más estrellas consigue para sus compañeros

```
445 Bird
```

Resolución Problema

```
r = open("nba.csv")
ls = r.readlines()
r.close()

#List de [n,s] con el nombre jugador
#y la suma de sus estrellas
ss = []
i = 0
while i < len(ls):
    pcs = ls[i].strip().split(";")
    ics = ls[i+1].strip().split(";")
```

```
#Maximo
maxn = -1
maxp = ""
for s in ss:
    if s[1] > maxn:
        maxp = s[0]
        maxn = s[1]
print(maxn,maxp)
```

Resolución Problema

```
r = open("nba.csv")
ls = r.readlines()
r.close()

#List de [n,s] con el nombre jugador
#y la suma de sus estrellas
ss = []
i = 0
while i < len(ls):
    pcs = ls[i].strip().split(";")
    ics = ls[i+1].strip().split(";")

    #Por cada jugador el quinteto
    for j in range(0,5):
        nom = pcs[j]

        #Por cada jugador del quinteto que
        #no es el que estoy mirando
        for k in range(0,5):
            if j != k:
                star = int(ics[k])

                esta = False
                for s in ss:
                    if s[0] == nom:
                        s[1] += star
                        esta = True

                if not esta:
                    ss.append([nom,star])
    i+=2

#Maximo
maxn = -1
maxp = ""
for s in ss:
    if s[1] > maxn:
        maxp = s[0]
        maxn = s[1]
print(maxn,maxp)
```

Situación Real

MANAGING AND LIMITED PARTNERS	
MANAGING PARTNER	Josh Harris
CO-MANAGING PARTNER	David Blitzer
LIMITED PARTNER	Adam Aron
LIMITED PARTNER	Martin Geller
LIMITED PARTNER	Travis Hennings
LIMITED PARTNER	Tony Ignaczak
LIMITED PARTNER	James Lassiter
LIMITED PARTNER	Marc Leder
LIMITED PARTNER	Michael Rubin
LIMITED PARTNER	Will Smith & Jada Pinkett Smith
LIMITED PARTNER	Art Wrubel

BASKETBALL OPERATIONS | EXECUTIVE LEADERSHIP	
GENERAL MANAGER	Elton Brand
EXECUTIVE VP, BASKETBALL OPERATIONS	Alex Rucker
ASSISTANT GENERAL MANAGER	Ned Cohen
VP, PLAYER DEVELOPMENT	Annelie Schmittel
VP, SCOUTING	Vince Rozman
VP, STRATEGY	Sergi Oliva
DIRECTOR, BASKETBALL OPERATIONS	Zach Sogolow

NBA.com Equipos Global Selecciona edición favorita

NBA Noticias y artículos Resultados Vídeos En clave española Clasificación Estac

14/11/2018 **Philadelphia 76ers**

Baloncesto a la sombra: la figura de Sergi Oliva en los Philadelphia 76ers

En su quinta temporada en las entrañas de la NBA, el catalán Sergi Oliva nos concede unos minutos para hablar sobre analytics, los 76ers y la NBA.

Situación Real

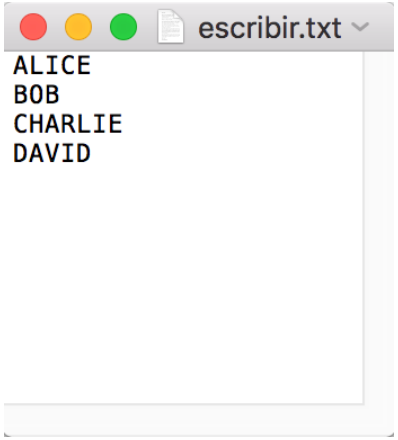
ESCRIBIR

```
ns = ["ALICE","BOB","CHARLIE","DAVID"]
```

```
ALICE
BOB
CHARLIE
DAVID
```

```
ns = ["ALICE","BOB","CHARLIE","DAVID"]

for e in ns:
    print(e)
```

```
ns = ["ALICE","BOB","CHARLIE","DAVID"]
```




```
ns = ["ALICE","BOB","CHARLIE","DAVID"]

#OPEN (con "w")
e = open("escribir.txt","w")

#PRINT
for n in ns:
    print(n,file=e)

#CLOSE
e.close()
```

RUT

Situación Real

Mi voz es mi herramienta de trabajo ¿Cómo puedo cuidarla?

Fecha: Viernes 01 de junio de 2018
Hora: 10:00 hrs a 13:00 hrs.

Sala: 401, Edificio CDDoc, Cuarto piso, Campus San Joaquín

* Required

Nombres *

Your answer

Apellidos *

Your answer

RUT o DNI (Sin puntos, con guión) *

Your answer

rutssucios.txt

```
12345678k
12.345.678.K
12 345 678 K
12345678-k
12-345-678-k
```

rutslimpios.txt

```
12.345.678-K
12.345.678-K
12.345.678-K
12.345.678-K
12.345.678-K
```


I'LL WAIT
FOR YOU HERE

rutssucios.txt

```
12345678k
12.345.678.K
12 345 678 K
12345678-k
12-345-678-k
```

rutslimpios.txt

```
12.345.678-K
12.345.678-K
12.345.678-K
12.345.678-K
12.345.678-K
```

```
def limpiar(sucio):
    r = ""
    for c in sucio:
        if c in "0123456789":
            r += c
        elif c in "kK":
            r += "K"
    s = r[:2]+"."+r[2:5]+"."+r[5:8]+"-"+r[8:]
    return s
```

...

```
def limpiar(sucio):
    r = ""
    for c in sucio:
        if c in "0123456789":
            r += c
        elif c in "kK":
            r += "K"
    s = r[:2]+"."+r[2:5]+"."+r[5:8]+"-"+r[8:]
    return s

r = open("rutssucios.txt")
ls = r.readlines()
r.close()

e = open("rutslimpios.txt","w")
for l in ls:
    sucio = l.strip()
    limpio = limpiar(sucio)
    print(limpio,file=e)
e.close()
```

CODIGOS IMDB

Situación Real

	A	B	C	D	E	F	G	H	I
1	USER	TIPO	CODE	ORIGINAL	SPAIN	LATAM			
2	0	movie	119217	Good Will Hunting	El indomable Will Hunting (Spain)	En busca del destino (Argentina)			
3	0	movie	97165	Dead Poets Society	El club de los poetas muertos (Spain)	La sociedad de los poetas muertos (Argentina)			
4	0	movie	118799	Life Is Beautiful	La vida es bella (Spain)	La vida es bella (Argentina)			
5	0	show	4270492	Billions	Billions (Spain)	Billions (Argentina)			
6	0	show	108778	Friends	Friends (Spain)	Friends (Argentina)			
7	0	show	3530232	Last Week Tonight with John Oliver	Last Week Tonight with John Oliver (Spain)	ND			
8	1	movie	4154796	Avengers: Endgame	Vengadores: Endgame (Spain)	Avengers: Endgame (Chile)			
9	1	movie	137523	Fight Club	El club de la lucha (Spain)	El club de la pelea (Argentina)			
10	1	movie	119654	Men in Black	Homes de negre (Spain) (Catalan title)	Hombres de negro (Argentina)			
11	1	show	903747	Breaking Bad	Breaking Bad (Spain)	ND			
12	1	show	1355642	Fullmetal Alchemist: Brotherhood	Fullmetal Alchemist: Brotherhood (Spain)	ND			

IMDbPY is a Python package for retrieving and managing the data of the IMDb movie database about movies and people.

DOWNLOADS READ THE DOCS

Retrieving

If you know the IMDb id of a movie, you can use the `get_movie` method to retrieve its data. For example, the movie “The Untouchables” by Brian De Palma has the id “0094226”:

```
>>> movie = ia.get_movie('0094226')
>>> movie
<Movie id:0094226[http] title:_The Untouchables (1987)_>
```

https://www.imdb.com/title/tt0113568/?ref_=nv_sr_srsg_3
https://www.imdb.com/title/tt0109830/?ref_=nv_sr_srsg_0
https://www.imdb.com/title/tt4154796/?ref_=nv_sr_srsg_0
https://www.imdb.com/title/tt0167260/?ref_=ttls_li_tt
https://www.imdb.com/title/tt1375666
https://www.imdb.com/title/tt1560139
https://www.imdb.com/title/tt0120794/?ref_=nv_sr_srsg_0
https://www.imdb.com/title/tt0289879/?ref_=nv_sr_srsg_0
https://www.imdb.com/title/tt1431045/?ref_=nv_sr_srsg_0
imdb.com/title/tt0816711/?ref_=nv_sr_srsg_0
https://www.imdb.com/title/tt1375666/?ref_=nv_sr_srsg_0
peaky blinders
vikings"
https://www.imdb.com/title/tt6751668/?ref_=hm_fanfav_tt_3_p
https://www.imdb.com/title/tt10431500/?ref_=nv_sr_srsg_0
https://www.imdb.com/title/tt0299658/?ref_=fn_al_tt_1
https://www.imdb.com/title/tt1596343/?ref_=fn_al_tt_1
https://www.imdb.com/title/tt0780504/?ref_=fn_al_tt_1
https://www.imdb.com/title/tt0110912/?ref_=nv_sr_srsg_0
https://www.imdb.com/title/tt0110912/?ref_=fn_al_tt_1
https://www.imdb.com/title/tt2119532/?ref_=nv_sr_srsg_0

imdbsucio.txt

```
https://www.imdb.com/title/tt4154796/?ref_=nv_sr_srsg_0
https://www.imdb.com/title/tt0167260/?ref_=ttls_li_tt
https://www.imdb.com/title/tt1375666
https://www.imdb.com/title/tt1560139
https://www.imdb.com/title/tt0120794/?ref_=nv_sr_srsg_0
https://www.imdb.com/title/tt0289879/?ref_=nv_sr_srsg_0
https://www.imdb.com/title/tt1431045/?ref_=nv_sr_srsg_0
imdb.com/title/tt0816711/?ref_=nv_sr_srsg_0
https://www.imdb.com/title/tt1375666/?ref_=nv_sr_srsg_0
peaky blinders
vikings"
https://www.imdb.com/title/tt6751668/?ref_=hm_fanfav_tt_3_pd_fp1
https://www.imdb.com/title/tt10431500/?ref_=nv_sr_srsg_0
https://www.imdb.com/title/tt0299658/?ref_=fn_al_tt_1
https://www.imdb.com/title/tt1596343/?ref_=fn_al_tt_1
https://www.imdb.com/title/tt0780504/?ref_=fn_al_tt_1
```

imdblimpio.txt

```
4154796
0167260
1375666
1560139
0120794
0289879
1431045
0816711
1375666
6751668
10431500
0299658
1596343
0780504
0110912
0110912
2119532
```


I'LL WAIT
FOR YOU HERE

imdbsucio.txt

```
https://www.imdb.com/title/tt4154796/?ref_=nv_sr_srsg_0
https://www.imdb.com/title/tt0167260/?ref_=ttls_li_tt
https://www.imdb.com/title/tt1375666
https://www.imdb.com/title/tt1560139
https://www.imdb.com/title/tt0120794/?ref_=nv_sr_srsg_0
https://www.imdb.com/title/tt0289879/?ref_=nv_sr_srsg_0
https://www.imdb.com/title/tt1431045/?ref_=nv_sr_srsg_0
imdb.com/title/tt0816711/?ref_=nv_sr_srsg_0
https://www.imdb.com/title/tt1375666/?ref_=nv_sr_srsg_0
peaky blinders
vikings"
https://www.imdb.com/title/tt6751668/?ref_=hm_fanfav_tt_3_pd_fp1
https://www.imdb.com/title/tt10431500/?ref_=nv_sr_srsg_0
https://www.imdb.com/title/tt0299658/?ref_=fn_al_tt_1
https://www.imdb.com/title/tt1596343/?ref_=fn_al_tt_1
https://www.imdb.com/title/tt0780504/?ref_=fn_al_tt_1
```

imdblimpio.txt

```
4154796
0167260
1375666
1560139
0120794
0289879
1431045
0816711
1375666
6751668
10431500
0299658
1596343
0780504
0110912
0110912
2119532
```

```
r = open("imdbsucio.txt")
ls = r.readlines()
r.close()

codes = []
for l in ls:
    cs = l.strip().split("/")
    for c in cs:
        if c[0:2] == "tt":
            codes.append(c[2:])

e = open("imdblimpio.txt","w")
for code in codes:
    print(code,file=e)
e.close()
```

PRINT (SEP)

¿Eres capaz de acertar cuántas facultades tienen estudiantes en Intro a la Progra este semestre?

Situación Real

14

¿Sabrías decir cuales?

```
500
173
137
40
11
30
38
6
3
1
4
1
1
1
```

¿?

Extendido.csv

```
Year,Semester,Section,Num in Sec,Facultad,Curriculo,Facultad
2020,1,9,1,5,50014,COMERCIAL
2020,1,9,2,5,50004,COMERCIAL
2020,1,9,3,5,50014,COMERCIAL
2020,1,9,4,5,50014,COMERCIAL
2020,1,9,5,5,50014,COMERCIAL
2020,1,9,6,9,90001,COLLEGE
2020,1,9,7,5,50014,COMERCIAL
```

```
CODE FAC NUM
5 COMERCIAL 500
9 COLLEGE 173
4 INGENIERIA 137
2 ASTRONOMIA 40
12 BIOLOGIA 11
3 FISICA 30
6 MATEMATICAS 38
11 AGRONOMIA 6
20 PEDAGOGIA 3
0 ESPECIAL 1
10 QUIMICA 4
70 MUSICA 1
33 ARTE 1
59 DISENO 1
```


I'LL WAIT
FOR YOU HERE

Extendido.csv

```
Year,Semester,Section,Num in Sec,Facultad,Curriculo,Facultad
2020,1,9,1,5,50014,COMERCIAL
2020,1,9,2,5,50004,COMERCIAL
2020,1,9,3,5,50014,COMERCIAL
2020,1,9,4,5,50014,COMERCIAL
2020,1,9,5,5,50014,COMERCIAL
2020,1,9,6,9,90001,COLLEGE
2020,1,9,7,5,50014,COMERCIAL
```

```
CODE FAC NUM
5 COMERCIAL 500
9 COLLEGE 173
4 INGENIERIA 137
2 ASTRONOMIA 40
12 BIOLOGIA 11
3 FISICA 30
6 MATEMATICAS 38
11 AGRONOMIA 6
20 PEDAGOGIA 3
0 ESPECIAL 1
10 QUIMICA 4
70 MUSICA 1
33 ARTE 1
59 DISENO 1
```

Extendido.csv

```
Year,Semester,Section,Num in Sec,Facultad,Curriculo,Facultad
2020,1,9,1,5,50014,COMERCIAL
2020,1,9,2,5,50004,COMERCIAL
2020,1,9,3,5,50014,COMERCIAL
2020,1,9,4,5,50014,COMERCIAL
2020,1,9,5,5,50014,COMERCIAL
2020,1,9,6,9,90001,COLLEGE
2020,1,9,7,5,50014,COMERCIAL
```

```
CODE FAC NUM
5 COMERCIAL 500
9 COLLEGE 173
4 INGENIERIA 137
2 ASTRONOMIA 40
12 BIOLOGIA 11
3 FISICA 30
6 MATEMATICAS 38
11 AGRONOMIA 6
20 PEDAGOGIA 3
0 ESPECIAL 1
10 QUIMICA 4
70 MUSICA 1
33 ARTE 1
59 DISENO 1
```

```
r = open("Extendido.csv")
ls = r.readlines()
r.close()

datos = []
for l in ls[1:]:
    cs = l.strip().split(",")

    esta = False
    for dato in datos:
        if dato[0] == cs[4]:
            dato[2] += 1
            esta = True

    if not esta:
        dato = [cs[4],cs[6],1]
        datos.append(dato)


print("CODE","FAC","NUM")
for d in datos:
    print(d[0],d[1],d[2])
```

Extendido.csv

```
Year,Semester,Section,Num in Sec,Facultad,Curriculo,Facultad
2020,1,9,1,5,50014,COMERCIAL
2020,1,9,2,5,50004,COMERCIAL
2020,1,9,3,5,50014,COMERCIAL
2020,1,9,4,5,50014,COMERCIAL
2020,1,9,5,5,50014,COMERCIAL
2020,1,9,6,9,90001,COLLEGE
2020,1,9,7,5,50014,COMERCIAL
```

facnum.csv

```
CODE,FAC,NUM
5,COMERCIAL,500
9,COLLEGE,173
4,INGENIERIA,137
2,ASTRONOMIA,40
12,BIOLOGIA,11
3,FISICA,30
6,MATEMATICAS,38
11,AGRONOMIA,6
20,PEDAGOGIA,3
0,ESPECIAL,1
10,QUIMICA,4
70,MUSICA,1
33,ARTE,1
59,DISENO,1
```

```
r = open("Extendido.csv")
ls = r.readlines()
r.close()

datos = []
for l in ls[1:]:
    cs = l.strip().split(",")

    esta = False
    for dato in datos:
        if dato[0] == cs[4]:
            dato[2] += 1
            esta = True

    if not esta:
        dato = [cs[4],cs[6],1]
        datos.append(dato)


e = open("facnum.csv","w")

print("CODE","FAC","NUM", file=e)
for d in datos:
    print(d[0],d[1],d[2],file=e)

e.close()
```

facnum.csv

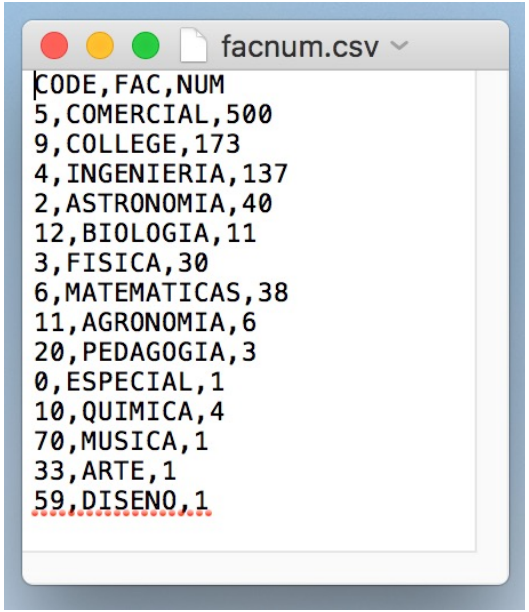
```
CODE,FAC,NUM
5,COMERCIAL,500
9,COLLEGE,173
4,INGENIERIA,137
2,ASTRONOMIA,40
12,BIOLOGIA,11
3,FISICA,30
6,MATEMATICAS,38
11,AGRONOMIA,6
20,PEDAGOGIA,3
0,ESPECIAL,1
10,QUIMICA,4
70,MUSICA,1
33,ARTE,1
59,DISENO,1
```

facnum.csv

```
CODE FAC NUM
5 COMERCIAL 500
9 COLLEGE 173
4 INGENIERIA 137
2 ASTRONOMIA 40
12 BIOLOGIA 11
3 FISICA 30
6 MATEMATICAS 38
11 AGRONOMIA 6
20 PEDAGOGIA 3
0 ESPECIAL 1
10 QUIMICA 4
70 MUSICA 1
33 ARTE 1
59 DISENO 1
```

```
CODE FAC NUM
5 COMERCIAL 500
9 COLLEGE 173
4 INGENIERIA 137
2 ASTRONOMIA 40
12 BIOLOGIA 11
3 FISICA 30
6 MATEMATICAS 38
11 AGRONOMIA 6
20 PEDAGOGIA 3
0 ESPECIAL 1
10 QUIMICA 4
70 MUSICA 1
33 ARTE 1
59 DISENO 1
```

```
r = open("Extendido.csv")
ls = r.readlines()
r.close()

datos = []
for l in ls[1:]:
    cs = l.strip().split(",")

    esta = False
    for dato in datos:
        if dato[0] == cs[4]:
            dato[2] += 1
            esta = True

    if not esta:
        dato = [cs[4],cs[6],1]
        datos.append(dato)


e = open("facnum.csv","w")

print("CODE","FAC","NUM", sep=",", file=e)
for d in datos:
    #sep nos sirve poner un str entre los
    #elementos a imprimir
    print(d[0],d[1],d[2],sep=",",file=e)

e.close()
```



JOIN

Situación Real

Red.csv

```
Year,Semester,Section,Num in Sec,Facultad,Curriculo
2020,1,9,1,5,50014
2020,1,9,2,5,50004
2020,1,9,3,5,50014
2020,1,9,4,5,50014
2020,1,9,5,5,50014
2020,1,9,6,9,90001
2020,1,9,7,5,50014
2020,1,9,8,5,50014
2020,1,9,9,5,50014
2020,1,9,10,5,50014
2020,1,9,11,5,50014
2020,1,9,12,9,90001
2020,1,9,13,5,50014
2020,1,9,14,5,50014
2020,1,9,15,5,50014
2020,1,9,16,5,50014
2020,1,9,17,5,50014
2020,1,9,18,5,50014
2020,1,9,19,5,50014
2020,1,9,20,5,50014
2020,1,9,21,5,50004
```

Facultades.csv

```
Num Facultad,Nom Facultad
5,COMERCIAL
9,COLLEGE
4,INGENIERIA
2,ASTRONOMIA
12,BIOLOGIA
3,FISICA
6,MATEMATICAS
11,AGRONOMIA
20,PEDAGOGIA
0,ESPECIAL
10,QUIMICA
70,MUSICA
33,ARTE
59,DISENO
```

Extendido.csv

```
2020,1,4,70,11,110020,AGRONOMIA
2020,1,4,71,4,40013,INGENIERIA
2020,1,4,72,2,20101,ASTRONOMIA
2020,1,4,73,2,20101,ASTRONOMIA
2020,1,4,74,9,90001,COLLEGE
2020,1,4,75,4,40013,INGENIERIA
2020,1,4,76,4,40013,INGENIERIA
2020,1,4,77,2,20101,ASTRONOMIA
2020,1,4,78,5,50004,COMERCIAL
2020,1,4,79,4,40013,INGENIERIA
2020,1,4,80,4,40013,INGENIERIA
2020,1,4,81,4,40013,INGENIERIA
2020,1,4,82,4,40013,INGENIERIA
2020,1,4,83,2,20101,ASTRONOMIA
2020,1,4,84,9,90001,COLLEGE
2020,1,4,85,9,90001,COLLEGE
2020,1,4,86,20,207201,PEDAGOGIA
```


I'LL WAIT
FOR YOU HERE

Situación Real

Red.csv

```
Year,Semester,Section,Num in Sec,Facultad,Curriculo
2020,1,9,1,5,50014
2020,1,9,2,5,50004
2020,1,9,3,5,50014
2020,1,9,4,5,50014
2020,1,9,5,5,50014
2020,1,9,6,9,90001
2020,1,9,7,5,50014
2020,1,9,8,5,50014
2020,1,9,9,5,50014
2020,1,9,10,5,50014
2020,1,9,11,5,50014
2020,1,9,12,9,90001
2020,1,9,13,5,50014
2020,1,9,14,5,50014
2020,1,9,15,5,50014
2020,1,9,16,5,50014
2020,1,9,17,5,50014
2020,1,9,18,5,50014
2020,1,9,19,5,50014
2020,1,9,20,5,50014
2020,1,9,21,5,50004
```

Facultades.csv

```
Num Facultad,Nom Facultad
5,COMERCIAL
9,COLLEGE
4,INGENIERIA
2,ASTRONOMIA
12,BIOLOGIA
3,FISICA
6,MATEMATICAS
11,AGRONOMIA
20,PEDAGOGIA
0,ESPECIAL
10,QUIMICA
70,MUSICA
33,ARTE
59,DISENO
```

Extendido.csv

```
2020,1,4,70,11,110020,AGRONOMIA
2020,1,4,71,4,40013,INGENIERIA
2020,1,4,72,2,20101,ASTRONOMIA
2020,1,4,73,2,20101,ASTRONOMIA
2020,1,4,74,9,90001,COLLEGE
2020,1,4,75,4,40013,INGENIERIA
2020,1,4,76,4,40013,INGENIERIA
2020,1,4,77,2,20101,ASTRONOMIA
2020,1,4,78,5,50004,COMERCIAL
2020,1,4,79,4,40013,INGENIERIA
2020,1,4,80,4,40013,INGENIERIA
2020,1,4,81,4,40013,INGENIERIA
2020,1,4,82,4,40013,INGENIERIA
2020,1,4,83,2,20101,ASTRONOMIA
2020,1,4,84,9,90001,COLLEGE
2020,1,4,85,9,90001,COLLEGE
2020,1,4,86,20,207201,PEDAGOGIA
```

```
fr = open("Facultades.csv")
fls = fr.readlines()
fr.close()

rr = open("Red.csv")
rls = rr.readlines()
rr.close()
```

```
fr = open("Facultades.csv")
fls = fr.readlines()
fr.close()

rr = open("Red.csv")
rls = rr.readlines()
rr.close()


#Header
e = open("Extendido.csv","w")
header = rls[0].strip()+",Facultad"
print(header,file=e)

for l in rls[1:]:
    cs = l.strip().split(",")

    #Buscar facultad
    for f in fls[1:]:
        fcs = f.strip().split(",")
        if fcs[0] == cs[4]:
            fac = fcs[1]
```

...

```
fr = open("Facultades.csv")
fls = fr.readlines()
fr.close()

rr = open("Red.csv")
rls = rr.readlines()
rr.close()


#Header
e = open("Extendido.csv","w")
header = rls[0].strip()+",Facultad"
print(header,file=e)

for l in rls[1:]:
    cs = l.strip().split(",")

    #Buscar facultad
    for f in fls[1:]:
        fcs = f.strip().split(",")
        if fcs[0] == cs[4]:
            fac = fcs[1]

    #Imprimir (lo hariamos asi)
    #print(cs[0],cs[1],cs[2],cs[3],cs[4],cs[5],fac)
    #Uffff y si en vez de 7 son 70 uffffff
```

...

```
e.close()
```


JOIN

```
fr = open("Facultades.csv")
fls = fr.readlines()
fr.close()

rr = open("Red.csv")
rls = rr.readlines()
rr.close()


#Header
e = open("Extendido.csv","w")
header = rls[0].strip()+",Facultad"
print(header,file=e)

for l in rls[1:]:
    cs = l.strip().split(",")

    #Buscar facultad
    for f in fls[1:]:
        fcs = f.strip().split(",")
        if fcs[0] == cs[4]:
            fac = fcs[1]

    #Imprimir (lo hariamos asi)
    #print(cs[0],cs[1],cs[2],cs[3],cs[4],cs[5],fac)
    #Uffff y si en vez de 7 son 70 uffffff

    #Si tienes una list de STR puedes crear un str
    #juntando los elementos con un str entre ellos
    s = ",".join(cs)
    s = s + ","+fac
    print(s,file=e)

e.close()
```

SAVE / LOAD

```
1-Impar 2-Par 0-Salir? 1
Dedos (de 0 a 5)? 3
3 + 5 = 8
Punto COMP
USER 0 COMP 1
1-Impar 2-Par 0-Salir? 1
Dedos (de 0 a 5)? 4
4 + 1 = 5
Punto USER
USER 1 COMP 1
1-Impar 2-Par 0-Salir? 0
```


I'LL WAIT
FOR YOU HERE

```
1-Impar 2-Par 0-Salir? 1
Dedos (de 0 a 5)? 3
3 + 5 = 8
Punto COMP
USER 0 COMP 1
1-Impar 2-Par 0-Salir? 1
Dedos (de 0 a 5)? 4
4 + 1 = 5
Punto USER
USER 1 COMP 1
1-Impar 2-Par 0-Salir? 0
```

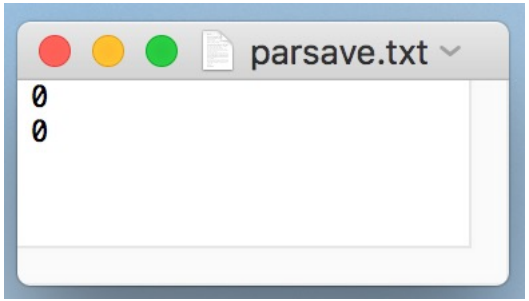
```
import random

u = 0
c = 0

continuar = True
while continuar:
    op = int(input("1-Impar 2-Par 0-Salir? "))
    if op == 0:
        continuar = False
    else:
        du = int(input("Dedos (de 0 a 5)? "))
        dc = random.randint(0,5)
        suma = du + dc
        print(du,"+",dc,"=",suma)

        if (suma%2 == 0 and op == 2) or (suma%2 == 1 and op == 1):
            print("Punto USER")
            u += 1
        else:
            print("Punto COMP")
            c += 1

        print("USER",u,"COMP",c)
```

Cada vez que lo vuelves a ejecutar empieza de nuevo

```
1-Impar 2-Par 0-Salir? 1
Dedos (de 0 a 5)? 3
3 + 5 = 8
Punto COMP
USER 0 COMP 1
1-Impar 2-Par 0-Salir? 1
Dedos (de 0 a 5)? 4
4 + 1 = 5
Punto USER
USER 1 COMP 1
1-Impar 2-Par 0-Salir? 0
>>>
>>>
>>>
>>>
>>> ====================
>>>
1-Impar 2-Par 0-Salir? 2
Dedos (de 0 a 5)? 0
0 + 0 = 0
Punto USER
USER 1 COMP 0
1-Impar 2-Par 0-Salir? 0
```

¿cómo podemos usar un archivo para guardar la partida?

```
1-Impar 2-Par 0-Salir? 2
Dedos (de 0 a 5)? 2
2 + 3 = 5
Punto COMP
USER 0 COMP 1
1-Impar 2-Par 0-Salir? 2
Dedos (de 0 a 5)? 3
3 + 3 = 6
Punto USER
USER 1 COMP 1
1-Impar 2-Par 0-Salir? 0
>>>
>>>
>>>
>>> ====================
>>>
1-Impar 2-Par 0-Salir? 1
Dedos (de 0 a 5)? 5
5 + 3 = 8
Punto COMP
USER 1 COMP 2
1-Impar 2-Par 0-Salir? 0
```



Un archivo para guardar la partida

Yo decido el formato

En este caso:

- Puntos User
- Puntos Comp

Inicialmente lo creo con dos 0

```
import random

#Auto-Load partida
r = open("parsave.txt")
ls = r.readlines()
r.close()

u = int(ls[0].strip())
c = int(ls[1].strip())

continuar = True
while continuar:
    op = int(input("1-Impar 2-Par 0-Salir? "))
    if op == 0:
        continuar = False

        #Auto-save
        #Creo otra vez el archivo en el formato de siempre
        e = open("parsave.txt","w")
        print(u,file=e)
        print(c,file=e)
        e.close()

    else:
        du = int(input("Dedos (de 0 a 5)? "))
        dc = random.randint(0,5)
        suma = du + dc
        print(du,"+",dc,"=",suma)

        if (suma%2 == 0 and op == 2) or (suma%2 == 1 and op == 1):
            print("Punto USER")
            u += 1
        else:
            print("Punto COMP")
            c += 1

        print("USER",u,"COMP",c)
```