11 – ARCHIVOS

Jorge Muñoz

IIC1103 – Introducción a la Programación


LISTAS
ARCHIVOS
STRINGS
What the hell is this?
imgflip.com

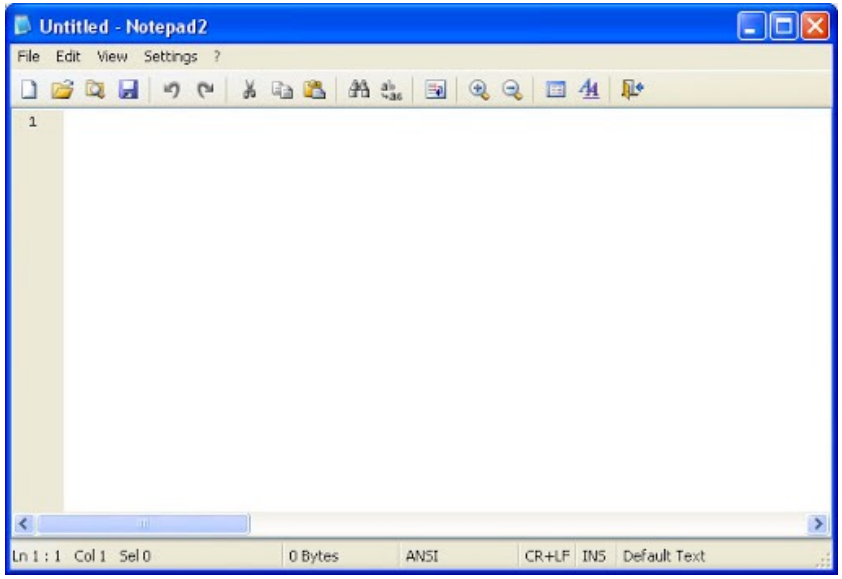
Archivos SIMPLES

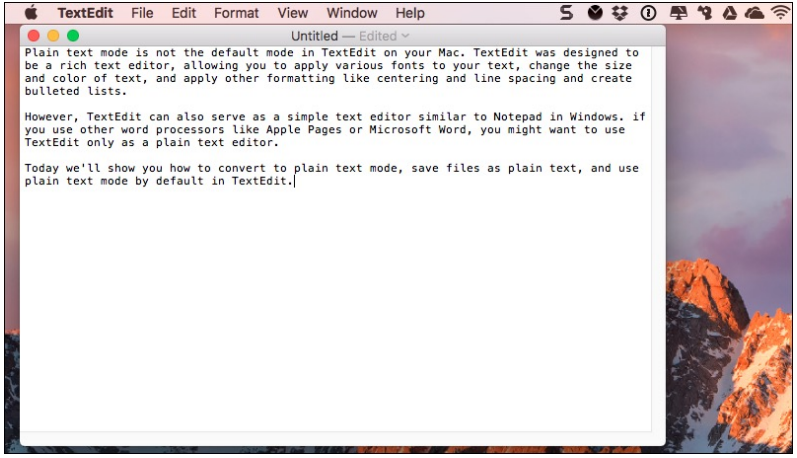
- Leer
  - strip
- Escribir

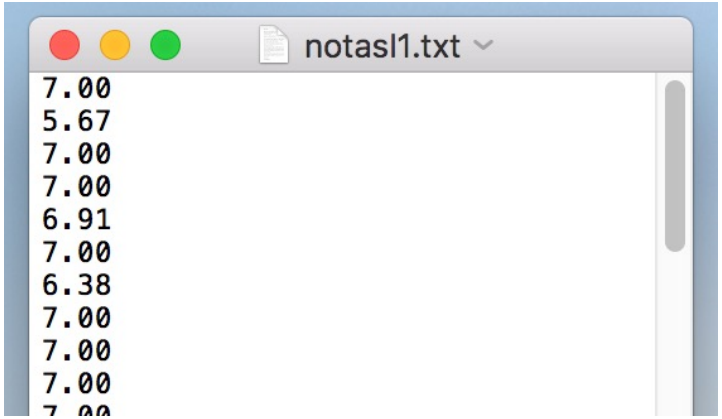
Archivos CVS

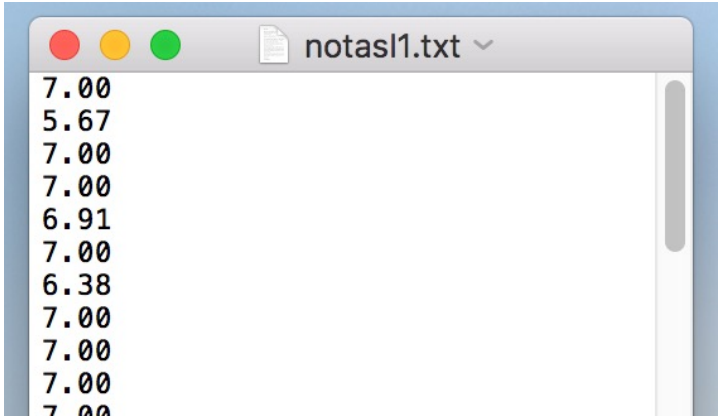
- Leer CSV
  - cabecera
- Escribir CSV

ARCHIVOS SIMPLES (Leer)


Untitled - Notepad2
File Edit View Settings ?
1
Ln 1 : 1 Col 1 Sel 0
0 Bytes
ANSI
CR+LF
INS
Default Text


TextEdit File Edit Format View Window Help
Untitled — Edited
Plain text mode is not the default mode in TextEdit on your Mac. TextEdit was designed to be a rich text editor, allowing you to apply various fonts to your text, change the size and color of text, and apply other formatting like centering and line spacing and create bulleted lists.
However, TextEdit can also serve as a simple text editor similar to Notepad in Windows. if you use other word processors like Apple Pages or Microsoft Word, you might want to use TextEdit only as a plain text editor.
Today we'll show you how to convert to plain text mode, save files as plain text, and use plain text mode by default in TextEdit.


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



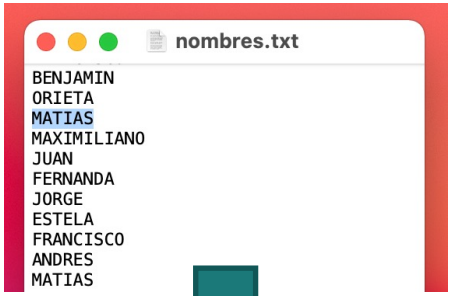
Promedio: 6.268080260303691

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

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

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

print("Promedio:",suma/len(ls))
```




```python
# OPEN - READLINE - CLOSE
uwu = open("nombres.txt")
ls = uwu.readlines()
uwu.close()

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

num = 0
for l in ls:
    if l == "MATIAS":
        num += 1

print("Hay", num, "Matias")
```

Hay 0 Matias



ALICE\n
BOB\n
CHARLIE\n
BOB\n
BOB\n
DAVID\n
EVA\n
BOB\n
FERNANDO\n

“BOB” != “BOB\n”



Hay 5 Matias

```python
# OPEN - READLINE - CLOSE
uwu = open("nombres.txt")
ls = uwu.readlines()
uwu.close()

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

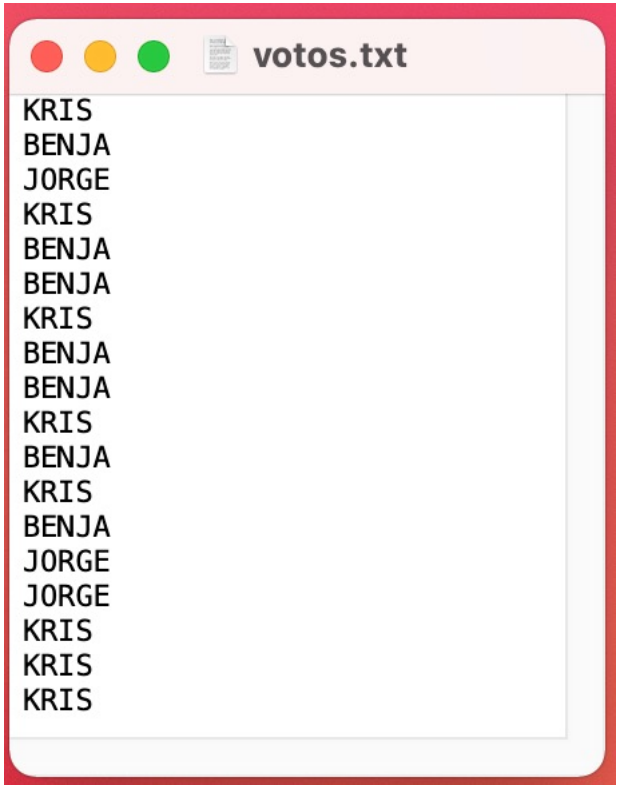
num = 0
for l in ls:

    #Metodo strip de str quita los \n del final
    nl = l.strip()

    if nl == "MATIAS":
        num += 1

print("Hay", num, "Matias")
```

KRIS, BENJA, JORGE



La mas popular es KRIS con 8 votos

votos.txt

```
KRIS
BENJA
JORGE
KRIS
BENJA
BENJA
KRIS
BENJA
BENJA
KRIS
BENJA
KRIS
BENJA
JORGE
JORGE
KRIS
KRIS
KRIS
```

La mas popular es KRIS con 8 votos

```python
# OPEN - READLINE - CLOSE
uwu = open("votos.txt")
ls = uwu.readlines()
uwu.close()

# Donde ahora 'ls' es una list de str
# donde cada elemento es una linea del archivo
nkris = 0
nbenja = 0
njorge = 0

for l in ls:
    nom = l.strip()
    if nom == "KRIS":
        nkris += 1
    elif nom == "BENJA":
        nbenja += 1
    elif nom == "JORGE":
        njorge += 1


if nkris > nbenja and nkris > njorge:
    print("La mas popular es KRIS con", nkris, "votos")
elif nbenja > nkris and nbenja > njorge:
    print("El mas popular es BENJA con", nbenja, "votos")
else:
    print("Obviamente el mas popular es JORGE con", njorge, "votos")
```

LET

INTERESES

“universidad”

“vida cuotidiana”

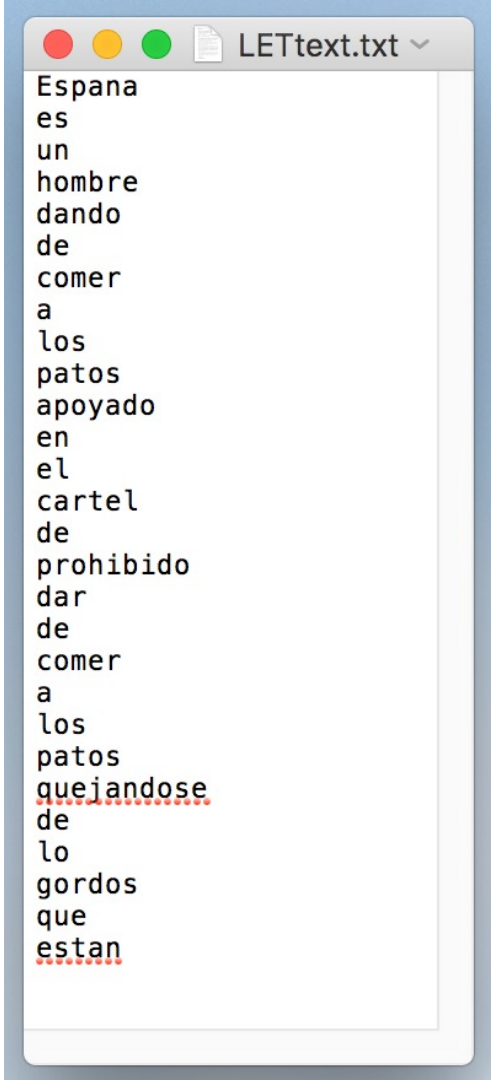
“útiles”

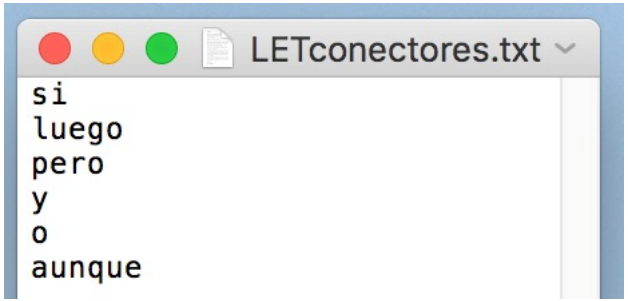
Estimado Profe:

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

```
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)
```

ARCHIVOS SIMPLES (Escribir)

```
print("Estoy fuera")
print("Entrando")
print("Estoy dentro")
print("Saliendo")
print("Estoy fuera")
```



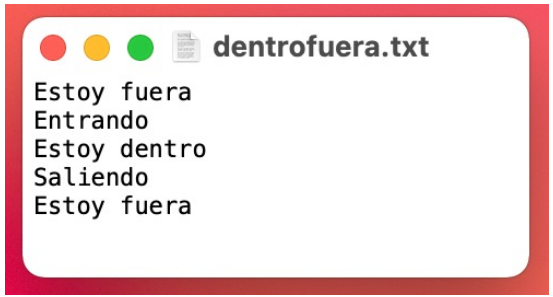
```
Estoy fuera
Entrando
Estoy dentro
Saliendo
Estoy fuera
```

```
# OPEN (con w)
uwu = open("dentrofuera.txt", "w")

#PRINT (con file)
print("Estoy fuera", file=uwu)
print("Entrando", file=uwu)
print("Estoy dentro", file=uwu)
print("Saliendo", file=uwu)
print("Estoy fuera", file=uwu)

# CLOSE
uwu.close()
```





RUT

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
```



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
```

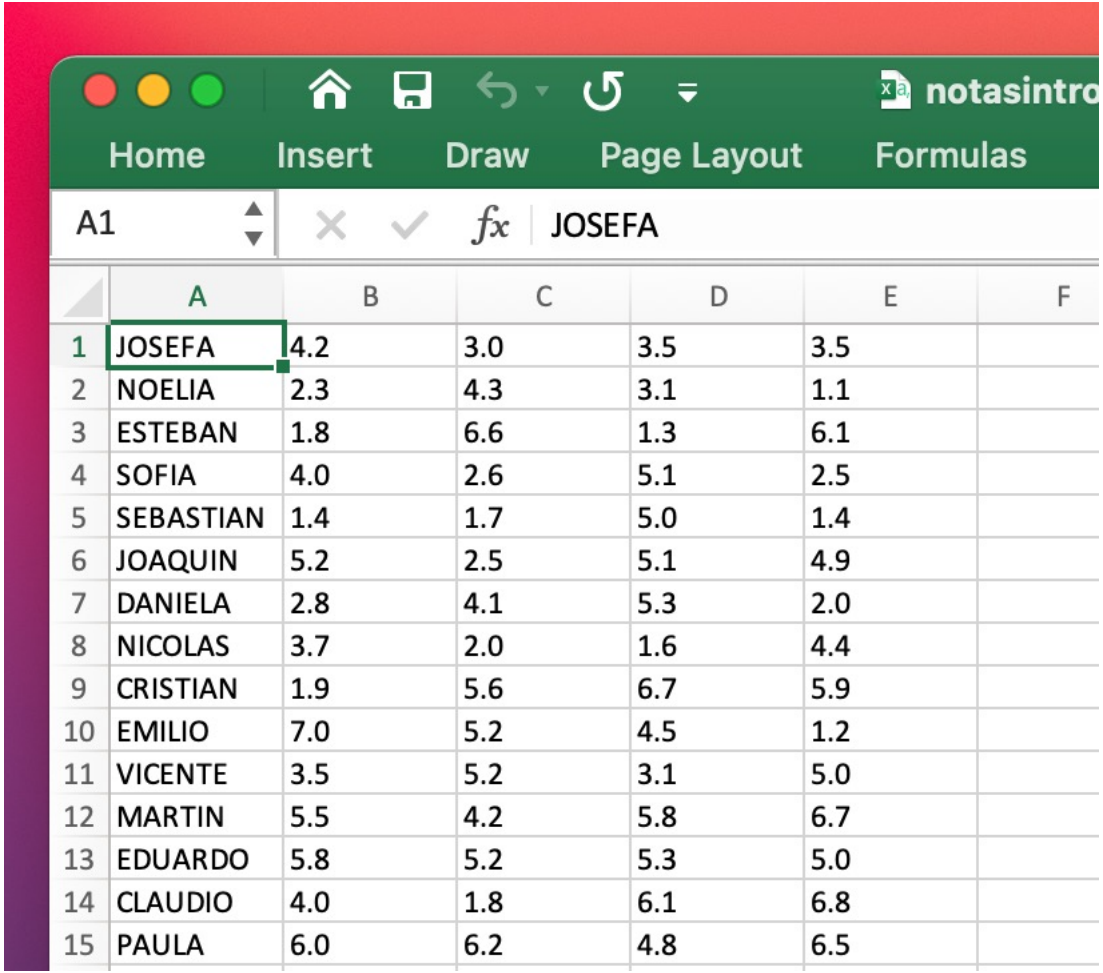


```
def limpiar(sucio):
    r = ""
    for c in sucio:
        if c in "0123456789":
            r += c
        elif c in "kK":
            r += "K"
    s = r[:2]+"."+r[2:5]+"."+r[5:8]+"-"+r[8:]
    return s

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()
```

ARCHIVOS CSV (Leer)



	A	B	C	D	E	F
1	JOSEFA	4.2	3.0	3.5	3.5	
2	NOELIA	2.3	4.3	3.1	1.1	
3	ESTEBAN	1.8	6.6	1.3	6.1	
4	SOFIA	4.0	2.6	5.1	2.5	
5	SEBASTIAN	1.4	1.7	5.0	1.4	
6	JOAQUIN	5.2	2.5	5.1	4.9	
7	DANIELA	2.8	4.1	5.3	2.0	
8	NICOLAS	3.7	2.0	1.6	4.4	
9	CRISTIAN	1.9	5.6	6.7	5.9	
10	EMILIO	7.0	5.2	4.5	1.2	
11	VICENTE	3.5	5.2	3.1	5.0	
12	MARTIN	5.5	4.2	5.8	6.7	
13	EDUARDO	5.8	5.2	5.3	5.0	
14	CLAUDIO	4.0	1.8	6.1	6.8	
15	PAULA	6.0	6.2	4.8	6.5	

notasintro.txt

```
JOSEFA,4.2,3.0,3.5,3.5
NOELIA,2.3,4.3,3.1,1.1
ESTEBAN,1.8,6.6,1.3,6.1
SOFIA,4.0,2.6,5.1,2.5
SEBASTIAN,1.4,1.7,5.0,1.4
JOAQUIN,5.2,2.5,5.1,4.9
DANIELA,2.8,4.1,5.3,2.0
NICOLAS,3.7,2.0,1.6,4.4
CRISTIAN,1.9,5.6,6.7,5.9
EMILIO,7.0,5.2,4.5,1.2
VICENTE,3.5,5.2,3.1,5.0
MARTIN,5.5,4.2,5.8,6.7
```

A veces

el separador es , (Google)

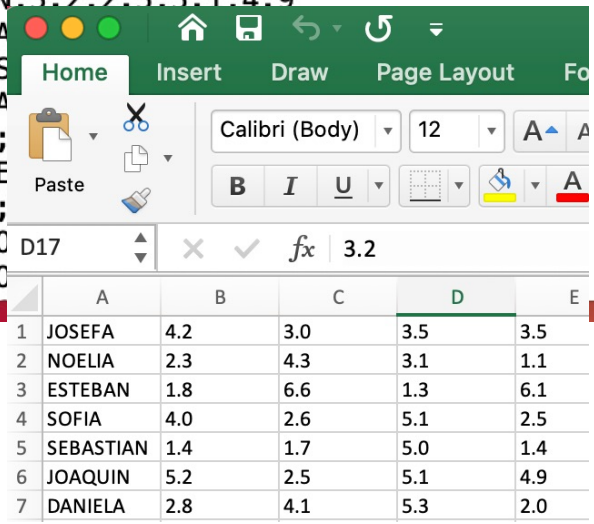
y a veces es ; (Excel)

notasintroPUNTOCOMA.txt

```
JOSEFA;4.2;3.0;3.5;3.5
NOELIA;2.3;4.3;3.1;1.1
ESTEBAN;1.8;6.6;1.3;6.1
SOFIA;4.0;2.6;5.1;2.5
SEBASTIAN;1.4;1.7;5.0;1.4
JOAQUIN;5.2;2.5;5.1;4.9
DANIELA;2.8;4.1;5.3;2.0
NICOLAS;3.7;2.0;1.6;4.4
CRISTIAN;1.9;5.6;6.7;5.9
EMILIO;7.0;5.2;4.5;1.2
VICENTE;3.5;5.2;3.1;5.0
MARTIN;5.5;4.2;5.8;6.7
EDUARDO;5.8;5.2;5.3;5.0
CLAUDIO;4.0;1.8;6.1;6.8
PAULA;6.0;6.2;4.8;6.5
TOMAS;4.3;1.3;5.2;4.9
```

notasintroPUNTOCOMA.csv

```
JOSEFA;4.2;3.0;3.5;3.5
NOELIA;2.3;4.3;3.1;1.1
ESTEBAN;1.8;6.6;1.3;6.1
SOFIA;4.0;2.6;5.1;2.5
SEBASTIAN;1.4;1.7;5.0;1.4
JOAQUIN;5.2;2.5;5.1;4.9
DANIELA
NICOLAS
CRISTIA
EMILIO;
VICENTE
MARTIN;
EDUARDO
CLAUDIO
```



	A	B	C	D	E
1	JOSEFA	4.2	3.0	3.5	3.5
2	NOELIA	2.3	4.3	3.1	1.1
3	ESTEBAN	1.8	6.6	1.3	6.1
4	SOFIA	4.0	2.6	5.1	2.5
5	SEBASTIAN	1.4	1.7	5.0	1.4
6	JOAQUIN	5.2	2.5	5.1	4.9
7	DANIELA	2.8	4.1	5.3	2.0

Si cambias .txt por **.csv**

Excel te lo abre bonito

Nombre	Tareas	Interrogaciones	Examen	Participacion

notasintro.txt

```
JOSEFA,4.2,3.0,3.5,3.5
NOELIA,2.3,4.3,3.1,1.1
ESTEBAN,1.8,6.6,1.3,6.1
SOFIA,4.0,2.6,5.1,2.5
SEBASTIAN,1.4,1.7,5.0,1.4
JOAQUIN,5.2,2.5,5.1,4.9
DANIELA,2.8,4.1,5.3,2.0
NICOLAS,3.7,2.0,1.6,4.4
CRISTIAN,1.9,5.6,6.7,5.9
EMILIO,7.0,5.2,4.5,1.2
VICENTE,3.5,5.2,3.1,5.0
MARTIN,5.5,4.2,5.8,6.7
```



```
Promedio de la participacion es 4.2405
```

```python
# OPEN - READLINE - CLOSE
uwu = open("notasintro.txt")
ls = uwu.readlines()
uwu.close()

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

suma = 0
for l in ls:

    nl = l.strip()

    # Metodo split de str, devuelve list of str
    # cortando el str por el separador
    cs = nl.split(",")

    notapart = float(cs[4])
    suma += notapart

print("Promedio de la participacion es",suma/len(ls))
```



	A	B	C	D	E
1	**Nombre**	**Tareas**	**Interrogaciones**	**Examen**	**Participacion**
2	JOSEFA	4.5	4.6	5.8	6.6
3	NOELIA	5.4	3.8	4.9	1.3
4	ESTEBAN	1.4	1.7	1.8	3.7
5	SOFIA	2.3	4.4	5.4	3.1
6	SEBASTIAN	1.4	4.8	6.0	4.1
7	JOAQUIN	4.3	2.4	5.0	1.2
8	DANIELA	6.0	4.1	3.4	5.0
9	NICOLAS	3.8	6.7	3.0	3.7
10	CRISTIAN	2.5	3.7	6.1	4.0
11	EMILIO	6.6	2.6	4.8	6.9
12	VICENTE	3.9	1.1	1.0	1.5
13	MARTIN	1.8	3.0	3.6	6.3
14	EDUARDO	2.0	6.2	2.9	4.4

notasintro.txt

```
Nombre,Tareas,Interrogaciones,Examen,Participacion
JOSEFA,2.4,3.6,6.4,5.6
NOELIA,4.5,4.6,5.8,7.0
ESTEBAN,3.8,4.0,6.8,1.7
SOFIA,3.6,6.2,2.9,5.9
SEBASTIAN,2.6,1.8,5.7,2.8
JOAQUIN,2.9,6.2,2.0,5.1
DANIELA,1.0,3.2,1.2,4.2
NICOLAS,3.1,1.3,6.6,1.4
CRISTIAN,3.2,3.2,1.7,5.4
EMILIO,6.7,6.5,2.3,3.8
VICENTE,3.0,4.4,5.5,3.7
MARTIN,4.4,3.2,4.8,4.9
```

notasintro.txt

```
Nombre,Tareas,Interrogaciones,Examen,Participacion
JOSEFA,2.4,3.6,6.4,5.6
NOELIA,4.5,4.6,5.8,7.0
ESTEBAN,3.8,4.0,6.8,1.7
SOFIA,3.6,6.2,2.9,5.9
SEBASTIAN,2.6,1.8,5.7,2.8
JOAQUIN,2.9,6.2,2.0,5.1
DANIELA,1.0,3.2,1.2,4.2
NICOLAS,3.1,1.3,6.6,1.4
CRISTIAN,3.2,3.2,1.7,5.4
EMILIO,6.7,6.5,2.3,3.8
VICENTE,3.0,4.4,5.5,3.7
MARTIN,4.4,3.2,4.8,4.9
```



```
Promedio de la participacion es 4.2405
```

```python
# OPEN - READLINE - CLOSE
uwu = open("notasintrohead.txt")
ls = uwu.readlines()
uwu.close()

#Head y resto
head = ls[0]
resto = ls[1:]

suma = 0
for l in resto:

    nl = l.strip()
    cs = nl.split(",")

    notapart = float(cs[4])
    suma += notapart

print("Promedio de la participacion es",suma/len(ls))
```

Gran Arcade DCC - A (Ex 2024-1)🌶️

La universidad ha decidido instalar máquinas Arcade de videojuegos por el campus. Los estudiantes te han pedido un sistema para poder visualizar sus puntajes.

Cada máquina cuenta con un archivo **`registro.txt`** que contiene una lista con estudiantes y sus puntajes asociados. Implementa un código que reciba como *input* el nombre de un estudiante e imprima el puntaje que obtuvo, de acuerdo a este archivo. Siempre se entregará el nombre de un estudiante que exista en la lista de puntajes.

Cada línea de **`registro.txt`** representa un puntaje, y se estructura de la siguiente forma:

```
{nombre_estudiante} {puntaje}
```

Donde **`nombre_estudiante`** es el nombre del estudiante, y **`puntaje`** es el puntaje que obtuvo. Puedes asumir que los nombres no van a contener espacios, que los puntajes siempre serán un número positivo, y que no habrán nombres de estudiantes repetidos.

Nota: Podras ver el contenido del archivo **`registro.txt`** junto al *output* habitual. Este se imprimirá automáticamente.

Ejemplo

Input

```
kaatt
```

Output

```
Contenido de registro.txt:
---------------
jorgeGamer 330
peachyhead 270
vale_ria 13
kaatt 455
denis777 999
---------------
Output Estudiante:
455
```

Explicación: Se recibe el nombre de **kaatt**. Como aparece en el archivo **`registro.txt`** con un puntaje de **455**, se imprime este valor.

ARCHIVOS CSV (Escribir)

```
print("A", "Arbol", "Ardilla")
print("B", "Barco", "Bandido")
print("C", "Castor", "Cabeza")
print("D", "Duende", "Dado")
print("E", "Enano", "Eelefante")
```



```
A Arbol Ardilla
B Barco Bandido
C Castor Cabeza
D Duende Dado
E Enano Eelefante
```

```
# OPEN (con w)
uwu = open("palabras.txt", "w")

#PRINT (con file)
print("A", "Arbol", "Ardilla", file=uwu)
print("B", "Barco", "Bandido", file=uwu)
print("C", "Castor", "Cabeza", file=uwu)
print("D", "Duende", "Dado", file=uwu)
print("E", "Enano", "Eelefante", file=uwu)

# CLOSE
uwu.close()
```





```python
# OPEN (con w)
uwu = open("palabrasSEP.txt", "w")

#PRINT (con file)
print("A", "Arbol", "Ardilla", file=uwu, sep=";")
print("B", "Barco", "Bandido", file=uwu, sep=";")
print("C", "Castor", "Cabeza", file=uwu, sep=";")
print("D", "Duende", "Dado", file=uwu, sep=";")
print("E", "Enano", "Eelefante", file=uwu, sep=";")


# CLOSE
uwu.close()
```



palabrasSEP.txt

```
A;Arbol;Ardilla
B;Barco;Bandido
C;Castor;Cabeza
D;Duende;Dado
E;Enano;Eelefante
```

```python
# OPEN (con w)
uwu = open("palabras.txt", "w")

#PRINT (con file)
print("A", "Arbol", "Ardilla", file=uwu)
print("B", "Barco", "Bandido", file=uwu)
print("C", "Castor", "Cabeza", file=uwu)
print("D", "Duende", "Dado", file=uwu)
print("E", "Enano", "Eelefante", file=uwu)


# CLOSE
uwu.close()
```



palabras.txt

```
A Arbol Ardilla
B Barco Bandido
C Castor Cabeza
D Duende Dado
E Enano Eelefante
```


LISTAS
ARCHIVOS
STRINGS
What the hell is this?
imgflip.com

Gran Arcade DCC - B (Ex 2024-1)🌶️🌶️

La universidad ha decidido instalar máquinas Arcade de videojuegos por el campus. Los estudiantes se han vuelto competitivos y quieren que estas cuenten con una tabla de los 10 estudiantes con mayor puntaje. Tu trabajo será implementar esta nueva funcionalidad.

Cada máquina cuenta con un archivo **`registro.txt`** que contiene una lista con estudiantes y sus puntajes asociados. Implementa un código que a partir de los puntajes en **`registro.txt`** genere un archivo **`ranking.txt`** con los 10 mejores puntajes ordenados de forma **decreciente** (del mayor puntaje al menor), en el mismo formato que el archivo **`registro.txt`**.

Cada línea de **`registro.txt`** representa un puntaje, y se estructura de la siguiente forma:

```
{nombre_estudiante} {puntaje}
```

Donde **`nombre_estudiante`** es el nombre del estudiante, y **`puntaje`** es el puntaje que obtuvo. Puedes asumir que los nombres no van a contener espacios, que los puntajes siempre serán un número positivo, y que no habrán estudiantes repetidos.

Nota: No te preocupes de manejar el *output*. Este se generará automáticamente a partir del contenido final de **`ranking.txt`**. Adicionalmente, se te mostrarán los contenidos de **`registro.txt`**. Este problema no cuenta con *input*.

Ejemplo

Output

```
Contenido de ranking.txt:
---------------
denis777 999
jocelyn_64 789
rodrigo_pro 766
dcchampion 605
kaatt 455
Biancaa 350
jorgeGamer 330
peachyhead 270
gato_con_botas 202
juanjo 125
---------------

Contenido de registro.txt:
---------------
rodrigo_pro 766
jorgeGamer 330
jocelyn_64 789
juanjo 125
peachyhead 270
vale_ria 13
kaatt 455
gato_con_botas 202
dcchampion 605
gabriel2 110
denis777 999
Biancaa 350
---------------
```

Explicación: Se recibe el archivo **`registro.txt`** con 12 puntajes. A partir de esto, se seleccionan los 10 mejores y se genera el archivo **`ranking.txt`**. Notar que en este archivo, las lineas se encuentran ordenadas de mayor a menor puntaje. Además, debido a que el archivo **`ranking.txt`** solo puede contener a 10 estudiantes, se descartan los puntajes de **`gabriel2`** (110) y **`vale_ria`** (13).

Gran Arcade DCC - C (Ex 2024-1)🌶️🌶️

La universidad ha decidido instalar máquinas Arcade de videojuegos por el campus. La implementación del sistema de rankings fue un éxito, sin embargo hay un problema: El archivo de ranking no se actualiza. Deberás agregar esta nueva funcionalidad.

Cada máquina cuenta con un archivo **`ranking.txt`** que contiene una lista de los **10 estudiantes con mayor puntaje** en orden decreciente según puntaje.

Implementa un código que reciba como *input* el nombre de un estudiante y un número entero que represente su puntaje, y genere un nuevo archivo **`actualizacion.txt`** con los 10 mejores puntajes **considerando este nuevo puntaje**, asegurando el orden de mayor a menor puntaje y el formato indicado.

En caso de que te entreguen el puntaje de un estudiante que no estaba en el ranking, deberás incorporarlo. Si este estudiante ya estaba en el ranking, deberás **actualizar su puntaje y posición**.

Cada linea de **`ranking.txt`** representa un puntaje, y se estructura de la siguiente forma:

```
{nombre_estudiante} {puntaje}
```

Donde **`nombre_estudiante`** es el nombre del estudiante, y **`puntaje`** es el puntaje que obtuvo. Las entradas del archivo siempre estarán **ordenadas de forma decreciente** por puntaje. Puedes asumir que los nombres no van a contener espacios y que los puntajes siempre serán un número positivo. Puedes considerar que siempre habrá un cambio en el ranking y que en caso de ser un estudiante ya existente, su nuevo puntaje siempre será mayor.

Nota: No te preocupes de manejar el *output*. Este se generará automáticamente a partir del contenido final de **`actualizacion.txt`**. Adicionalmente, se te mostrarán los contenidos de **`ranking.txt`**.

Ejemplo

Input

```
emol
76
```

Output

```
Contenido de actualizacion.txt:
---------------
jorgeGamer 883
rodrigo_pro 766
tomas01 443
violeta 441
Biancaa 350
denis777 300
gabriel2 150
vale_ria 101
emol 76
MasterCristian 57
---------------

Contenido de ranking.txt:
---------------
jorgeGamer 883
rodrigo_pro 766
tomas01 443
violeta 441
Biancaa 350
denis777 300
gabriel2 150
vale_ria 101
MasterCristian 57
Monta 6
---------------
```

Explicación: Se recibe el puntaje de **emol**, que en este caso es 76. Este puntaje se ubica entre el de **vale_ria** (101 puntos) y **MasterCristian** (57 puntos), por lo que se inserta a **emol** en medio, en la novena posición. Como la tabla de posiciones solo contiene a 10 estudiantes, **Monta**, quien originalmente era el décimo, queda desplazado fuera de esta.

I2 2017-2 Pregunta 2 (Teleoperadores)

La empresa *Smooth Operator SpA* está teniendo problemas con los operadores telefónicos de su *call center*. Muchos de los operadores están entregando un mal servicio y hasta ahora no tienen cómo identificarlos. Por esto, decidieron registrar los datos de todas llamadas que se estaban recibiendo. Además, implementaron un sistema de evaluación telefónica, en donde los clientes califican a los operadores luego de terminar su llamada. Luego de acumular meses de datos, se acercaron a los talentosos alumnos de Introducción a la Programación con el objetivo de poder interpretar los datos que han obtenido.

Para esto les entregaron el archivo `registro.csv`. Este archivo contiene información de cada llamada atendida por los operadores en los últimos 6 meses. La primera línea contiene un encabezado (los nombres de cada columna), y las siguientes líneas están escritas de la forma:

```
nombre_cliente;nombre_operador;duracion_llamada;rating_llamada
```

En este archivo:

- `nombre_cliente` es el nombre de quien llamó al call center.
- `nombre_operador` es el nombre del operador que atendió la llamada.
- `duracion_llamada` es la duración total de la llamada en minutos.
- `rating_llamada` es la calificación que dejó el cliente por la atención del operador.

Las primeras 10 líneas de este archivo se ven de la siguiente forma:

```
nombre;operador;tiempo_llamada;rating_llamada
pedro;juan;15;3
valentin;maria;2;5
diego;juan;18;2
jose;maria;5;4
anita;camilo;8;5
sofia;manuel;7;1
esteban;maria;10;3
cristina;maria;3;4
julio;marcos;1;2
```

Escribe un programa que calcule el desempeño de los operadores de este *call center*. Tu programa debe crear un archivo con nombre `reporte.csv`, donde en cada línea tenga los siguientes datos:

```
nombre_operador;duracion_maxima_llamada;rating_promedio_llamadas
```

En donde:

- `duracion_maxima_llamada` es la duración de la llamada más larga que atendió el operador.
- `rating_promedio_llamadas` es el promedio de las calificaciones que dejaron los clientes a las atenciones del operador.

Por ejemplo, el archivo que debería resultar si tomamos como entrada las primeras 10 líneas del archivo `registro.csv`:

```
operador;duracion_maxima_llamada;rating_promedio_llamada
juan;18;2.5
maria;10;4.0
camilo;8;5.0
manuel;7;1.0
marcos;1;2.0
```

Observación: El archivo de salida no necesita estar ordenado.

I2 2016-2 Pregunta 2 (Visitas Carreras)

Pregunta 1 (1/3)

La universidad está organizando visitas de colegios a cada una de las carreras que imparte. Para esto, abrió un formulario que le permite a los escolares inscribirse en la visita de su interés. Los nombres de los inscritos y la carrera de su interés están (en orden de llegada) en el archivo `inscritos.txt`. Pero solo hay un cierto número de cupos para la visita a cada carrera, los que se encuentran en el archivo `carreras.txt`.

inscritos.txt

```
Alan Turing,Ingenieria
Eloisa Diaz,Medicina
Gustave Eiffel,Ingenieria
Ada Lovelace,Ingenieria
Larry King,Periodismo
Jonas Salk,Medicina
Grace Hopper,Ingenieria
...
```

carreras.txt

```
Ingenieria 55
Medicina 1
Periodismo 10
Enfermeria 35
College 40
...
```

Se te pide que escribas un programa que, a partir de los archivos anteriores, genere para cada carrera, un archivo (cuyo nombre debe ser el nombre de la carrera en minúsculas, seguido de .txt) con los estudiantes inscritos para la visita. Si hay más interesados que cupos, solo quedan inscritos los primeros que hayan hecho la inscripción. Si no hay ningún interesado en alguna visita, el archivo debe crearse y quedar vacío. A continuación se muestran, a modo de ejemplo, los archivos resultantes para Ingeniería, Medicina y Periodismo.

ingenieria.txt

```
Alan Turing
Gustave Eiffel
Ada Lovelace
Grace Hopper
...
```

medicina.txt

```
Eloisa Diaz
```

periodismo.txt

```
Larry King
...
```

I2 2017-1 Pregunta 2 (Consumos)

Una compañía de telefonía móvil te pide que hagas un programa para crear un archivo (**"saldos.txt"**) con los saldos de segundos de llamadas y kilobytes de descarga que le quedan a cada uno de sus clientes, de acuerdo al plan que tienen contratado. Para ello, debes leer la información de clientes (**"clientes.txt"**), planes (**"planes.txt"**), y consumos ya efectuados (**"consumos.txt"**).

A continuación, se muestra un ejemplo simple:

clientes.txt

```
Hugo;Normal
Paco;Super
Luis;Platino
```

planes.txt

```
Normal;120;10
Super;240;40
Platino;360;100
```

consumos.txt

```
Hugo;llamada;24
Paco;llamada;12
Hugo;descarga;800000
Paco;descarga;800000
Luis;descarga;800000
Luis;llamada;14
Luis;llamada;21
Paco;llamada;24
Hugo;llamada;12
Luis;llamada;18
Hugo;descarga;500000
Paco;llamada;18
Luis;descarga;300000
Hugo;llamada;14
Paco;descarga;300000
Paco;descarga;500000
Luis;descarga;500000
Hugo;descarga;300000
```



saldos.txt

```
Hugo;7150;8400000
Paco;14346;38400000
Luis;21547;98400000
```

Notar que la información de los planes está en minutos y gigabytes (GB), en tanto que la información de los consumos y saldos está en segundos y kilobytes (KB). Por simplicidad, puedes asumir que 1 GB = 1000000 KB. Puedes también asumir que el separador entre los datos es un ";".

Por ejemplo, **Hugo** ha hecho 3 llamadas por un total de 50 segundos (**24** + **12** + **14**), y por estar en un plan **Normal** tiene 7200 segundos (**120** * 60). Por lo tanto, su saldo es **7150** segundos. A su vez, ha hecho 3 descargas, totalizando 1600000 KB (**800000** + **500000** + **300000**). Dado que su plan **Normal** considera **10** GB, su saldo es de **8400000** KB (**10** * 1000000 − 1600000). Se destaca en "negritas" aquellos datos que se leen desde los archivos o que se escriben al archivo **"saldos.txt"**.