Expresiones regulares con grep y sed

PEC3

Jorge Aguilera

Ejercicio 1

En este primer ejercicio tendréis que encontrar una expresión regular que resuelva una posible pregunta inicial de exploración de un conjunto de datos. Se pide:

Encontrar todas las combinaciones ganadoras en diagonal del tres en línea del conjunto de datos “Tic-Tac-Toe Endgame Data Set“, el cual se puede encontrar aquí:

https://archive.ics.uci.edu/ml/datasets/tic-tac-toe+Endgame

Demostrad todas las operaciones escribiendo los comandos y mediante capturas de pantalla, añadiendo las explicaciones necesarias

```
$ awk '/x\,.\,.\,.\,x\,.\,.\,.\,x\,positive/ || /.\,.\,x\,.\,x\,.\,x\,.\,.\,positive/
{print;}' pec3/tic-tac-toe.data
```

Explicación:

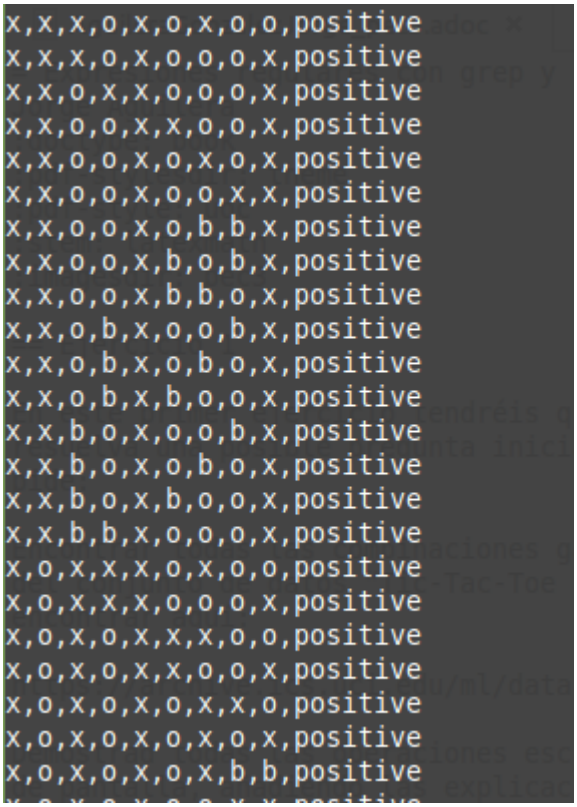
Buscamos todas las lineas cuya matriz sea

x	-	-
-	x	-
-	-	x

o bien

-	-	x
-	x	-
x	-	-

y que terminen en "positive". El comando anterior produce la siguiente salida



```
x,x,x,o,x,o,x,o,o,positive
x,x,x,o,x,o,o,o,x,positive
x,x,o,x,x,o,o,o,x,positive
x,x,o,o,x,x,o,o,x,positive
x,x,o,o,x,o,x,o,x,positive
x,x,o,o,x,o,o,x,x,positive
x,x,o,o,x,o,b,b,x,positive
x,x,o,o,x,b,o,b,x,positive
x,x,o,o,x,b,b,o,x,positive
x,x,o,b,x,o,o,b,x,positive
x,x,o,b,x,o,b,o,x,positive
x,x,o,b,x,b,o,o,x,positive
x,x,b,o,x,o,o,b,x,positive
x,x,b,o,x,o,b,o,x,positive
x,x,b,o,x,b,o,o,x,positive
x,x,b,b,x,o,o,o,x,positive
x,o,x,x,x,o,x,o,o,positive
x,o,x,x,x,o,o,o,x,positive
x,o,x,o,x,x,x,o,o,positive
x,o,x,o,x,x,o,o,x,positive
x,o,x,o,x,o,x,x,o,positive
x,o,x,o,x,o,x,o,x,positive
x,o,x,o,x,o,x,b,b,positive
```

Ejercicio 2

Dominar las expresiones regulares incluye también la capacidad de comprender y modificar expresiones regulares creadas por otros programadores. Este ejercicio está centrado en la comprensión y modificación, si es el caso, de expresiones regulares ya dadas. Este ejercicio no se tiene que demostrar con capturas de pantalla, sino que hay que explicarlo de forma detallada.

Partiendo del conjunto de datos de kaggle.com (https://www.kaggle.com/), "ISOTC213-General Product Specification Standards" (https://www.kaggle.com/giacomomarchioro/isotc213), se ha utilizado, para encontrar unos determinados estándares, la siguiente orden:

```
$ grep ".*,.*,Geometrical.*,.*,20.*,Published.*,.*,.*,0" ISO_TC213.csv
```

Buscar todos los registros cuyo titulo empiecen por "Geometrical" de cualquier año superior a 1999, que su status sea "Published" y que sean gratuitos

a) Describid con vuestras propias palabras cuáles son los estándares que parece que se pretendían buscar y cómo está formada la expresión regular. (1 punto)

La expresión busca separar por comas las diferentes partes de cada línea, de tal forma que los dos primeros no nos importa su contenido y del tercero buscamos que empieze por Geometrical. Así mismo se busca que el año de publicación sea superior al 1999 y que esté publicado

Posteriormente, se ha empleado como expresión equivalente:

```
$ grep ".*,Geometrical.*,.*,20.*,Published.*,0" ISO_TC213.csv
```

b) Se observa que esta segunda expresión no devuelve el mismo número de resultados que la anterior y se ha acabado descartando. Explicad hasta qué punto son equivalentes las dos expresiones regulares y el motivo del descarte. (1 punto)

Las dos expresiones son parecidas porque intentan buscar el mismo patrón (El texto Geometrical, el año 20xx seguido del texto Published) sin embargo la segunda expresión es menos restrictiva al no separar todos los campos por la coma mientras que la primera particiona la cadena en todos sus campos.

Ejercicio 3

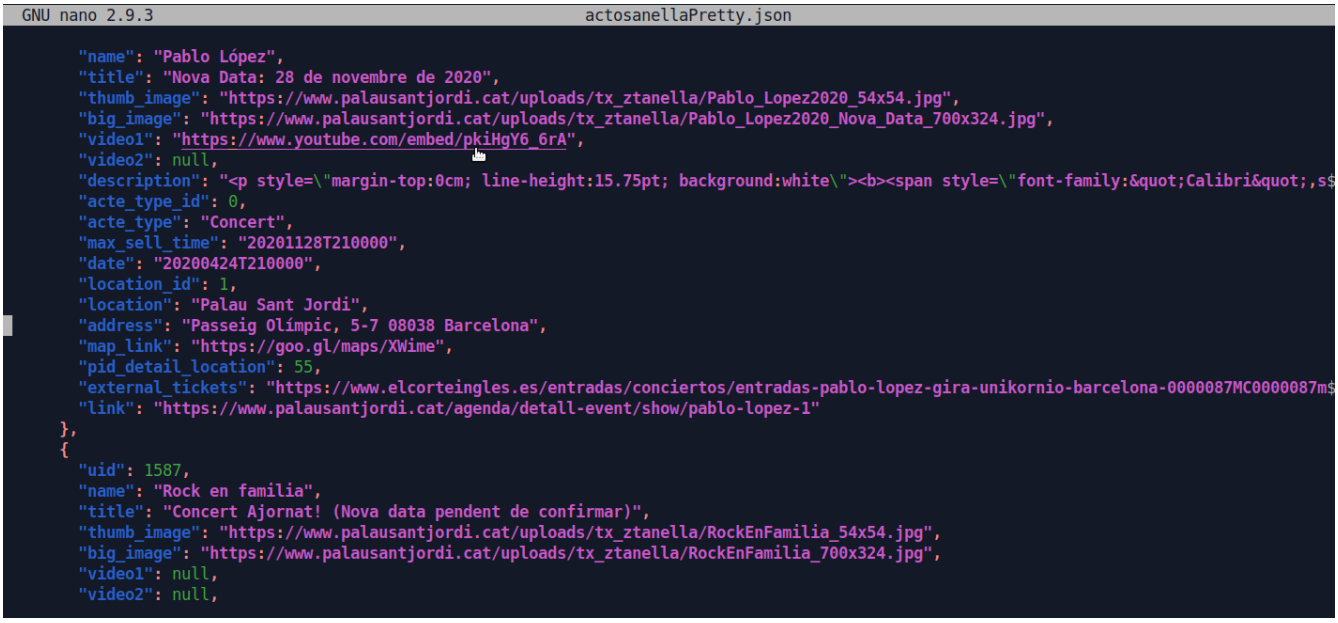
Este ejercicio sirve para ilustrar y mostrar el uso del editor sed. Éste es especialmente útil para hacer sustituciones de expresiones regulares, aunque sus funcionalidades no se limitan a éso. En este ejercicio tendréis que poner en práctica el uso de sed para diferentes usos.

A partir de la Agenda de actas de la "Anella Olímpica" de la ciudad de Barcelona que se puede encontrar aquí:

https://opendata-ajuntament.barcelona.cat/data/es/dataset/actes-anella/resource/b0314d7c-9743-4b90-aee2-a2d6a0ae9d48

Mostrad los comandos necesarios para:

a) Dado que todos los datos están en una sola línea, usando la línea de comandos, cread un nuevo archivo "pretty printed". Observad la imagen para saber cómo debería de quedar. (0.5 puntos)




```
$ cat actosanella.json | jq . > actosanella_pretty.json
```

b) Buscad con grep todas las líneas donde aparece el datetime en el formato indicado, como 20200426T200000. (0.5 puntos)

```
$ grep -Eo '[0-9]{8}T[0-9]{6}' actosanella_pretty.json

    20210123T210000
    20200417T210000
    20200418T210000
    20200418T210000
    20201008T221500
    20200418T221500
    20201128T210000
    20200424T210000
    20200425T173000
    20200425T173000
    20200426T200000
    20200426T200000
    20210214T210000
    20200428T200000
    20200515T210000
    20200515T210000
    20200705T213000
    20200515T213000
    20201114T210000
    20200516T210000
    20200523T200000
    20200523T200000
    20200606T213000
    20200606T213000
    20200612T220000
    20200612T220000
...
```

c) Usando sed, borrad las líneas donde aparezca “null”. (0.5 puntos)

```
$ sed '/null/d' actosanella_pretty.json  > actosanella_sin_null.json
```

d) Usando sed, cambiar el formato de la fecha por uno más amigable, donde sólo aparezca la fecha, sin la hora. Por ejemplo, en lugar de 20200426T200000 debería aparecer 26-04-2020. (1.5 puntos)

```
$ sed -i -E 's/([0-9]{4})([0-9]{2})([0-9]{2})T([0-9]{6})/\3-\2-\1/g'
actosanella_sin_null.json
$ grep '"date":' actosanella_sin_null.json
"date": "17-04-2020",
        "date": "18-04-2020",
        "date": "18-04-2020",
        "date": "24-04-2020",
        "date": "25-04-2020",
        "date": "26-04-2020",
        "date": "28-04-2020",
        "date": "15-05-2020",
        "date": "15-05-2020",
        "date": "16-05-2020",
        "date": "23-05-2020",
        "date": "06-06-2020",
        "date": "12-06-2020",
        "date": "17-06-2020",
        "date": "20-06-2020",
        "date": "30-06-2020",
        "date": "07-07-2020",
        "date": "25-07-2020",
```

Demostrad todas las operaciones escribiendo los comandos y mediante capturas de pantalla, añadiendo las explicaciones necesarias.

Ejercicio 4

Como se ha visto en los materiales docentes, awk es una herramienta con muchas posibilidades para filtrar datos, generar informes, etc. Además, se pueden crear scripts awk para automatizar tareas. En este ejercicio se tendrá que crear un script de awk para tratar un conjunto de datos.

A partir del fichero csv "Personal docent en centres públics titularitat del Departament d'Educació" (Personal docente en centros públicos titularidad del Departamento de Educación), disponible en:

https://analisi.transparenciacatalunya.cat/es/Educaci-/Personal-docent-en-centres-p-blics-titularitat-del/2ip7-jdgh

el cual se puede encontrar en el portal de transparencia de la Generalitat de Catalunya, vamos a estudiar la proporción de mujeres (campo número 7) y hombres (campo número 8) trabajando como docentes según el tipo de centro.

a) Encontrad todos los tipos de centros que hay. Para hacerlo, se debe emplear la primera palabra del tercer campo. Indicad el comando y una captura de pantalla. (0.5 puntos)

```
$ awk -F"," 'NR>1 {split($3,c," "); print c[1]}'
Personal_docent_en_centres_p_blics_titularitat_del_Departament_d_Educaci_.csv |sort
|uniq
AFA
CEE
CFA
EASD
EOI
ESCOLA
ESCRBC
INST-ESC
INSTITUT
PARVULARI
ZER
```

b) Haced un recuento del número de centros que hay de cada tipo. Indicad el comando y una captura de pantalla. (0.5 puntos)

```
$ awk -F"," 'NR>1 {split($3,c," "); print c[1]}'
Personal_docent_en_centres_p_blics_titularitat_del_Departament_d_Educaci_.csv |sort
|uniq -c
      2 AFA
     25 CEE
    125 CFA
      8 EASD
     45 EOI
   1664 ESCOLA
      1 ESCRBC
      1 INST-ESC
    603 INSTITUT
      1 PARVULARI
     85 ZER
```

c) Cread un script awk que reciba el nombre del tipo de centro como parámetro y devuelva, tanto para hombres como para mujeres, el total, el porcentaje, la media y la desviación típica. El script tendrá el Shebang de awk, tal como se explica en los materiales docentes. Debe mostrar al final el nombre del autor. Por ejemplo, para el tipo de centro EOI, la salida podría ser (los valores pueden no ser correctos):

```
Personal docente en centros públicos, titularidad del
Departamento de Educación
--------------- Tipología del centro: EOI --------------
Mujeres
Total mujeres = 421(75%). Media mujeres = 9.356, Desviación
típica mujeres = 7.874.
Hombres
Total hombres = 134(24%). Media hombres = 2.978, Desviación
típica hombres = 2.266.
2020, Informe creado por Guillem Lluch Moll
```

Incluid el código en el documento, una captura de su ejecución y entregad también el script. (2 puntos)

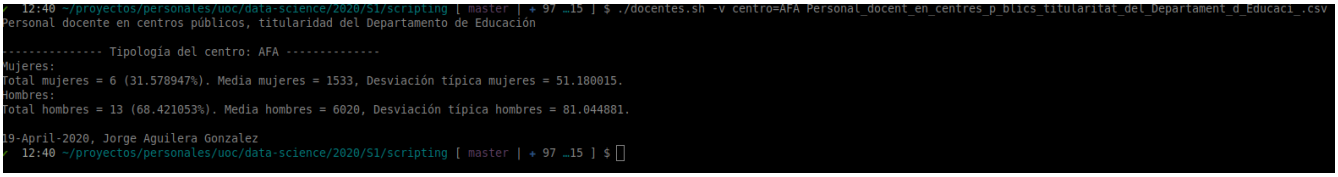
docentes.sh

```
#!/usr/bin/awk -f

BEGIN {
 if(centro == ""){
    print "argumento Centro requerido"
    error=1
 }

 FS=","
}
```

```
function abs(value) {
    return value < 0 ? -value : value
}

{
 split($3,nombre," ")
 tipo=nombre[1]

 tmujeres = tmujeres+$8
 thombres = thombres+$9

 marray[tipo] = marray[tipo]+$8
 harray[tipo] = harray[tipo]+$9
}

END {
 if(error==1){
  exit -1
 }
 cmujeres = marray[centro]
 chombres = harray[centro]

 pmujeres = (cmujeres / (cmujeres+chombres))*100
 phombres = 100-pmujeres

 mmujeres = tmujeres / length(marray)
 mhombres = thombres / length(harray)

 msum=0
 for (i in marray){
    msum = msum + abs(marray[i]-mmujeres)
 }
 dmujeres = sqrt(msum/(length(marray)-1))

 hsum=0
 for (i in harray){
    hsum = hsum + abs(harray[i]-hmujeres)
 }
 dhombres = sqrt(hsum/(length(harray)-1))

 printf "Personal docente en centros públicos, titularidad del Departamento de
Educación\n\n"
 printf "--------------- Tipología del centro: %s --------------\n", centro
 printf "Mujeres:\n"
 printf "Total mujeres = %d (%f%). Media mujeres = %d, Desviación típica mujeres =
%f.\n", cmujeres,pmujeres,mmujeres,dmujeres
 printf "Hombres:\n"
 printf "Total hombres = %d (%f%). Media hombres = %d, Desviación típica hombres =
%f.\n", chombres,phombres,mhombres,dhombres
 printf "\n%s, Jorge Aguilera Gonzalez\n", strftime("%d-%B-%Y")
}
```

ejemplo AFA

```
./docentes.sh -v centro=AFA
Personal_docent_en_centres_p_blics_titularitat_del_Departament_d_Educaci_.csv
```



Ejercicio 5

El formato de este ejercicio es un poco más abierto que el resto y, de hecho, enlazará con uno de los ejercicios de la última PEC4 (que consistirá en la elaboración de un proyecto propio de interés personal).

- Buscad y elegid un conjunto de datos para un análisis posterior. Podéis escoger el conjunto que queráis, pero debe de ser lo suficientemente grande como para que tratarlo con herramientas informáticas sea imprescindible.
- Elaborad un pequeño informe sobre el conjunto de datos escogido. En concreto, describid el formato de los ficheros que almacenan el conjunto de datos, su tamaño (por ejemplo, número de ficheros, columnas y registros) y formato de los distintos campos (texto, numérico, etc..). Para realizar esta descripción es necesario utilizar herramientas de Bash.

Incidencias en M30

El conjunto de datos a tratar se refiere al detalle de las incidencias habidas en la vía de circunvalidación de Madrid M30 (más info en https://datos.madrid.es/sites/v/index.jsp?vgnextoid=6e1ce0f3e8e22610VgnVCM1000001d4a900aRCRD&vgnextchannel=374512b9ace9f310VgnVCM100000171f5a0aRCRD) en concreto los ficheros CSV para los años 2016, 2017 y 2018 (2019 pendiente de publicación)

Los ficheros CSV se encuentran en las siguientes URLs:

- 2016 https://datos.madrid.es/egob/catalogo/300201-2-calle30-accidentes-historico.csv
- 2017 https://datos.madrid.es/egob/catalogo/300201-0-calle30-accidentes-historico.csv
- 2018 https://datos.madrid.es/egob/catalogo/300201-4-calle30-accidentes-historico.csv

Si nos centramos en el año 2018 podemos ver que el fichero tiene las siguientes características:

- Tamaño (858k)

```
$ ls -lhat *.csv
-rw-rw-r-- 1 jorge jorge *858K* Apr 19 12:59 Listado_accidentes_2018.csv
```

- registros (2232)

```
$ wc -l Listado_accidentes_2018.csv
2233 Listado_accidentes_2018.csv
```

- campos 40

```
awk -F";" 'NR==1 {print NF}' Listado_accidentes_2018.csv
40
```

Respecto de la tipología de los datos podemos analizar los 13 primeros:

```
awk -F";" 'NR<3 {print $1; print $2; print $3; print $4; print $5; print $6; print $7;
print $8; print $9; print $10; print $11; print $12; print $13 }'
Listado_accidentes_2018.csv
Fecha
Hora
Dia
N Incidencia
SITREM
Carretera
MC30
CALZADA
Localizacion
Enlace
Tipo de accidente
Tipo de colision
N Vehiculos implicados

01/01/2018
18:49
lunes
20180101-0015
 C-E-ACC-01
M30
04NC70
INTERIOR
Tronco
Superficie
Salida de via
Lateral
2
```

De lo que podemos deducir que dispondremos de 2 campos iniciales que unidos dan un campo fecha-hora, identificadores de incidencias y lugares de tipo texto, así como contadores numéricos de vehículos implicados, etc.