# Teoría de la computación - Práctica 1

**Grupo Lunes 8:00-10:00 semanas B**
**— Práctica 1 —**
**Autor:** Fabrizio Duarte
**NIP:736857**
**Autor:** Fernando Pastor Peralta
**NIP:897113**

**Ejercicio 1:**

**1. Resumen:**
Se trata de un ejercicio básico, en el que usamos la regla @hotmail e imprimimos en su lugar el @gmail.

**2.Pruebas:**

**ENTRADA:**

```
1 perico@hotmail.com ana@unizar.es
2 sab@hotmail.com javier@garcia.com
3 maria@hotmail.com modrego@rm.edu
4 jose@unizar.es
5
```

**SALIDA:**

```
perico@gmail.com ana@unizar.es
sab@gmail.com javier@garcia.com
maria@gmail.com modrego@rm.edu
jose@unizar.es
hendrix01:~/home/kali/Desktop/ProgramasP1-TComp1/ej1_p1_Tcomp/ pwd
/home/a897113/home/kali/Desktop/ProgramasP1-TComp1/ej1_p1_Tcomp
```

**Ejercicio 2:**

**1.Resumen:**
Otro ejercicio trivial, en el que declaramos un contador en la zona de declaraciones, en la zona de reglas aumentamos el contador por cada aparición de la regla @hotmail y en la zona del código imprimimos el número de @hotmail's encontrados.

**2.Pruebas:**

**ENTRADA:**

```
1 perico@hotmail.com ana@unizar.es
2 sab@hotmail.com javier@garcia.com
3 maria@hotmail.com modrego@rm.edu
4 jose@unizar.es
5
```

**SALIDA:**

```
perico@hotmail.com ana@unizar.es
sab@hotmail.com javier@garcia.com
maria@hotmail.com modrego@rm.edu
jose@unizar.es
 Total de @hotmail : 3
hendrix01:~/home/kali/Desktop/ProgramasP1-TComp1/ej2_p1_Tcomp/ pwd
/home/a897113/home/kali/Desktop/ProgramasP1-TComp1/ej2_p1_Tcomp
```

**Ejercicio 3:**

**1.Resumen:**
Es parecido al ejercicio 1, con el único detalle de que el carácter punto se trata de forma especial y comenzamos a utilizar las herramientas que nos proporciona flex en este caso las comillas con el punto ".".

**2.Pruebas:**

**ENTRADA:**

```
1 perico@hotmail.com ana@unizar.es
2 sab@hotmail.com javier@garcia.com
3 maria@hotmail.com modrego@rm.edu
4 jose@unizar.es
5 jrg@unizaroes.es
6
```

**SALIDA:**

```
perico@hotmail.com ana@gmail.com
sab@hotmail.com javier@garcia.com
maria@hotmail.com modrego@rm.edu
jose@gmail.com
jrg@unizaroes.es
hendrix01:~/home/kali/Desktop/ProgramasP1-TComp1/ej3_p1_Tcomp/ pwd
/home/a897113/home/kali/Desktop/ProgramasP1-TComp1/ej3_p1_Tcomp
```

**Ejercicio 4:**

**1.Resumen:**
Nuestra regla se asegura de encontrar un número, con ello hacemos uso de la función atoi para transformar la variable de tipo carácter guardado en yytext en un entero. Posteriormente sumamos uno al anterior número y lo imprimimos. Además por cada aparición de un salto de línea (\n), pondremos 2 de ellos.

**2.Pruebas:**

**ENTRADA:**

```
1 Mi numero de telefono es 548271210
2 ponte en contacto con
3 el asistente 59 en 04 minutos.
4
```

**SALIDA:**

```
Mi numero de telefono es 659382321

ponte en contacto con

el asistente 610 en 15 minutos.

hendrix02:~/home/kali/Desktop/ProgramasP1-TComp1/ej4_p1_Tcomp/ pwd
/home/a897113/home/kali/Desktop/ProgramasP1-TComp1/ej4_p1_Tcomp
```

**Ejercicio 5:**

**1.Resumen**

Se trata de un ejercicio donde utilizamos 6 contadores:
C (carácteres) = Utilizamos el . para las situaciones en las que no se aplica ninguna de las otras reglas e yyleng en cada una de las demás reglas para contabilizar el número de carácteres que se han consumido en dicha regla.
L (cantidad de líneas): contamos los fines de línea (que sólo contabilizan en esta regla, en el caso de líneas con texto) y sumamos cada una de las líneas en blanco que encontramos.
LV (líneas en blanco): aplicamos una regla en la que buscamos únicamente espacios en una línea y si esto ocurre, contabilizamos una nueva LV.
V (valores): cada vez que encontramos un valor ya sea numérico o alfabético, contabilizamos un nuevo V (en sus respectivas reglas) .
M (mayor cantidad de carácteres de un valor): aplicamos if(yyleng > M) M = yyleng; en las reglas de valores numéricos y alfabéticos para encontrar el mayor número de carácteres.
N (valores numéricos): su regla específica, suma un nuevo N al aplicarse.

**2.Pruebas:**

**ENTRADA:**

```
1 Ciudad ,Habitantes,Comida
2 Zaragoza, 600000, migas
3
4 Barcelona,1000000,pan y tomate
5 Huesca,30000,trenza de Huesca
6
```

(Solo 1 espacio en la línea 3.)

**SALIDA:**

```
C: 108
L: 5
LV: 1
V: 12
M: 16
N: 3
hendrix01:~/home/kali/Desktop/ProgramasP1-TComp1/ej5_p1_Tcomp/ pwd
/home/a897113/home/kali/Desktop/ProgramasP1-TComp1/ej5_p1_Tcomp
```

**Ejercicio 6:**

**1.Resumen:**

Usaremos la orden egrep de linux:

-Para las líneas que tienen que empezar y finalizar con un dígito, permitiremos que existan espacios al inicio y fin de la línea y obligaremos a que haya 2 números al inicio y fin de una serie de carácteres cualesquiera.

*egrep '^[ ]*[0-9].*[0-9][ ]*$' t.txt*

-Para las líneas donde contengan únicamente un identificador y una dirección ip, comenzaremos con una serie de espacios indefinidos, después una letra obligatoria y una cantidad indefinida de carácteres alfanuméricos, luego, debe haber por lo menos un espacio y la IP será de 4 números de entre 1 y 3 cifras separadas por puntos, al final permitimos una cantidad de espacios cualesquiera.

*egrep '^[ ]*[a-zA-Z][a-zA-Z0-9]*[ ]+[0-9][0-9]?[0-9]?[.][0-9][0-9]?[0-9]?[.][0-9][0-9]?[0-9]?[.][0-9][0-9]?[0-9]?[ ]*$' t.txt*

-Para líneas que contengan un número impar decimal ,el número puede tener cuantos dígitos sean, pero estará obligado a terminar en un número que haga que el número sea impar, entonces no permitiremos que haya un número después de acabar nuestro número.

*egrep '[0-9]*[13579]([^0-9]|$)' t.txt*

**2.Pruebas:**

**ENTRADA:**

```
1 AUTORES: Fernando Pastor Peralta (897113) Fabrizio Bryan Duarte Jáuregui (736857)
2 egrep '^[ ]*[0-9].*[0-9][ ]*$' t.txt
3 egrep '^[ ]*[a-zA-Z][a-zA-Z0-9]*[ ]+[0-9][0-9]?[0-9]?[.][0-9][0-9]?[0-9]?[.][0-9][0-9]?[0-9]?[.][0-9][0-9]?[0-9]?[ ]*$' t.txt
4 egrep '[0-9]*[13579]([^0-9]|$)' t.txt
5
```

t.txt

```
 1 1 Esto tiene que imprimirse en la primera 1
 2  Esto tiene imprimirse en la tercera 3
 3 hola
 4 jajaj
 5     q talll
 6    1936-1939
 7    x2a3
 8    1939_WW2_1945
 9  2_HOLA
10  fabri3
11            69
12  f45f
13   EstoTieneQueImprimirseEnLaSegunda2          1.200.2.67
14   fermin37 25.255.21.4
15  fermin37 25.255.21.4 pepe
16  37 fermin 25.255.21.4
17  fermin37 25.255.21
18  fermin37 25.255.21.8.67
19  fermin37 25.255.21.8.
20  fermin37 25.255..21.8
21  fermin37 25.2556.1.8
22  hola 22dados
23 131hola 212dados
24       21
25      x21x
26      22
27      y22y
28      2121132
29      2121133 h
30      2121133
31
```

## SALIDA:

```
hendrix02:~/home/kali/Desktop/ProgramasP1-TComp1/ej6_p1_Tcomp/ egrep '^[ ]*[0-9].*[0-9][ ]*$' t.
txt
1 Esto tiene que imprimirse en la primera 1
   1936-1939
   1939_WW2_1945
            69
 37 fermin 25.255.21.4
      21
     22
     2121132
     2121133
hendrix02:~/home/kali/Desktop/ProgramasP1-TComp1/ej6_p1_Tcomp/ pwd
/home/a897113/home/kali/Desktop/ProgramasP1-TComp1/ej6_p1_Tcomp
```

```
hendrix02:~/home/kali/Desktop/ProgramasP1-TComp1/ej6_p1_Tcomp/ egrep '^[ ]*[a-zA-Z][a-zA-Z0-9]*[
][0-9]?[0-9]?[ ]*$' t.txt                       <
  EstoTieneQueImprimirseEnLaSegunda2          1.200.2.67
  fermin37 25.255.21.4
hendrix02:~/home/kali/Desktop/ProgramasP1-TComp1/ej6_p1_Tcomp/ pwd
/home/a897113/home/kali/Desktop/ProgramasP1-TComp1/ej6_p1_Tcomp
```

```
hendrix02:~/home/kali/Desktop/ProgramasP1-TComp1/ej6_p1_Tcomp/ egrep '[0-9]*[13579]([^0-9]|$)' t
.txt
1 Esto tiene que imprimirse en la primera 1
 Esto tiene imprimirse en la tercera 3
   1936-1939
   x2a3
   1939_WW2_1945
 fabri3
          69
 f45f
  EstoTieneQueImprimirseEnLaSegunda2          1.200.2.67
  fermin37 25.255.21.4
 fermin37 25.255.21.4 pepe
 37 fermin 25.255.21.4
 fermin37 25.255.21
 fermin37 25.255.21.8.67
 fermin37 25.255.21.8.
 fermin37 25.255..21.8
 fermin37 25.2556.1.8
131hola 212dados
     21
    x21x
    2121133 h
    2121133
hendrix02:~/home/kali/Desktop/ProgramasP1-TComp1/ej6_p1_Tcomp/ pwd
/home/a897113/home/kali/Desktop/ProgramasP1-TComp1/ej6_p1_Tcomp
```

**Ejercicio 7:**
**1.Resumen:**
1º Caso n (D's) múltiplo de 5: Construimos una regla en la que si solo encontramos A's estaremos en el caso de 5*0, si encontramos D's tendrán que ir en grupos de 5 en 5 separadas por cualesquiera cantidad de A's. En ese caso pondremos al inicio y fin de la cadena ++.

2º Caso n (D's) par y no múltiplo de 5: Nuestra regla comienza exigiendo tener una cantidad de mínimo 2 D's y podremos repetir este proceso 3 veces más pudiendo obtener los rangos de 2, 4, 6 y 8 D`s. La regla no permitirá los números múltiplos de 5, entonces exigirá mínimo que tenga 4 D's más, obteniendo un número nuevo en el rango de D's permitidas de 12, luego se podrán sumar entre 1 y 3 veces más 2 D's por lo tanto cubriremos el rango de 14, 16 y 18, el siguiente número sería múltiplo de 5, por lo tanto se repetirá el proceso sumando 4, y así comenzará el ciclo anterior ((18+4=)22, (+2) 24, 26, 28, 32, 34, 36, 38, 42…).

3º Caso n (D's) impar u no múltiplo de 5: Seguimos el razonamiento anterior de exigencia y bucle, en el que empezamos exigiendo mínimo una D y luego solo se permitirán 3 D's (cubriendo los casos de 1 y 3 D's), el siguiente número será múltiplo de 5, entonces exigimos que haya 4 D's más, obteniendo el caso de (7 D's) y luego podremos obtener los siguientes impares sumando de 1 a 3 veces 2 D's (obteniendo el rango de 9,11 y 13 D`s). El siguiente número sería múltiplo de 5, por lo que el bucle comenzaría de nuevo ((13+4=)17, (+2) 19, 21, 23, 27, 29, 31, 33, 37…).

**2.Pruebas:**

**ENTRADA:**

```
1 querido francisco:
2 me puedes marcar como hemos quedado
3 las cadenas
4 IF, IDAADAAF,
5 IAAAADADADADDF
6 IDDDDDDDDDDF IDAF
7
```

**SALIDA:**

```
querido francisco:
me puedes marcar como hemos quedado
las cadenas
++IF++, +IDAADAAF+,
++IAAAADADADADDF++
++IDDDDDDDDDDF++ -IDAF-
hendrix01:~/home/kali/Desktop/ProgramasP1-TComp1/ej7_p1_Tcomp/ pwd
/home/a897113/home/kali/Desktop/ProgramasP1-TComp1/ej7_p1_Tcomp
```