

Grupo Miércoles 17:00-19:00 semanas A

– Práctica 1 –

Autor: Javier Martínez Virto

NIP:930853

Autor: Víctor Marteles Martínez

NIP:928927

Ejercicio 5:

1.Resumen

Hemos creado el patrón “[0-9]+:[0-9]+” para reconocer valores que son duplas de números separados por “:”.

Hemos creado el patrón “[A-Za-z:][A-Za-z:][A-Za-z:]*[A-Za-z:]” para reconocer valores formados únicamente por letras y el símbolo “:”, con espacios intermedios pero nunca al inicio ni al final.

Hemos creado el patrón “[A-Za-z0-9:][A-Za-z0-9:][A-Za-z0-9:]*[A-Za-z0-9:]” para reconocer valores generales compuestos por letras, números o “:”, permitiendo espacios internos pero no en los extremos.

Hemos creado el patrón “\n” para reconocer saltos de línea, lo que nos permite llevar la cuenta del número total de líneas.

Hemos creado el patrón “^[]*\$” para reconocer líneas vacías que contienen únicamente espacios.

Hemos creado el patrón “^\n” como caso particular para detectar líneas vacías sin espacios, es decir, aquellas que solo contienen un salto de línea.

Hemos creado el patrón “.” como regla comodín que captura cualquier carácter no reconocido por expresiones anteriores, de manera que todos los símbolos queden contabilizados en el total de caracteres.

2.Pruebas

Entrada:

Salida:

Nombre,Edad,Ciudad,Clave¶ Ana,23,Madrid,111:222¶ Luis,45,Sevilla,abc¶ ,, ,¶ Pedro,30,Valencia,999:1¶ Maria, ,Zaragoza,so1¶ ¶	C: 113 L: 7 LV: 1 V: 19 M: 8 D: 2 N: 14
--	---

Ejercicio 6:

1.Resumen

Hemos creado el patrón “`^[^0-9].*[^0-9]$`” para reconocer líneas que no empiezan ni terminan con un dígito. Con este patrón podemos filtrar las líneas que empiezan y acaban en cualquier otro carácter válido.

Hemos creado el patrón “`^[]*[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][]+[A-Za-z][A-Za-z0-9]*$`” para reconocer líneas que contienen un identificador, seguido de una dirección IP válida y un alias, separados por espacios en blanco (admitiendo también espacios al inicio y al final de la línea). Con esta expresión controlamos que el identificador y el alias empiecen por una letra y que la dirección IP esté formada por cuatro bloques de entre 1 y 3 dígitos separados por puntos.

Hemos creado el patrón “`[13579][^0-9][0-9]*[13579](^[^0-9])$`” para reconocer líneas que contienen al menos un número impar en decimal. Para ello se buscan secuencias de dígitos consecutivos y se verifica que al menos uno termine en un dígito impar.

2.Pruebas

Para la entrada “t.txt” he obtenido la salidas:

fermin37 25.255.21.4 ¶ fermin37 25.255.21.4 pepe ¶ 38fermin 20.258.28.4 pepe ¶ s721s¶	1. S721s 2. fermin37 25.255.21.4 pepe 3. fermin37 25.255.21.4 fermin37 25.255.21.4 pepe
--	--

Ejercicio 7:

1.Resumen

Hemos creado el patrón “B(R*UR*UR*UR*)*E” para reconocer cadenas que hacen que el robot termine en una posición (n,m) donde m es múltiplo de 3. En este caso, la acción asociada añade ++ al inicio y al final de la cadena.

Hemos creado el patrón “B(R*UR*UR*)*E” para reconocer cadenas que llevan al robot a una posición (n,m) donde m es par pero no múltiplo de 3. En este caso, se encierra la cadena entre :.

Hemos creado el patrón B(R*UR*UR*)*(R*UR*)E para reconocer cadenas que finalizan en una posición (n,m) donde m es impar y no múltiplo de 3. En este caso, se encierra la cadena entre guiones -.

2.Pruebas

Entrada:

hola robot: aquí tienes más cadenas BRUE BUUE BRRUE BUUUUE BUUUUUE BRRRUUUE BUUUUUUE saludos	hola robot: aquí tienes más cadenas -BRUE- :BUUE: -BRRUE- :BUUUUE: -BUUUUUE- ++BRRRUUUE++ ++BUUUUUUE++ saludos
---	---