# Ejercicio 1

Crea un programa que al ejecutarse reciba dos números enteros como parámetros. El primer entero es el puerto de escucha, el segundo es el puerto de envío.

En cada Datagrama UDP espera encontrar otros dos números y un carácter separados por espacios, el primer número representa la altura, el segundo el ancho y el carácter es el carácter utilizado a continuación.

Habrá una clase Receptor que recibe Datagramas en el puerto de escucha, implementa el patrón Observer. Con cada datagrama avisa a los que observan con la información para generar el cuadrado y la información para enviar la respuesta.

Habrá una clase GeneradorCuadrado con un método estático que genera el cuadrado.

Habrá una clase Enviador que observa la recepción de Datagramas en Receptor, cuando llega un nuevo Datagrama usa la clase GeneradorCuadrado y la información que ha recibido para enviar por Broadcast el cuadrado.

Al recibir cada Datagrama el programa generará un cuadrado con el alto y el ancho recibidos y lo enviará en el segundo puerto a través de Broadcast UDP.

El programa estará leyendo hasta que se le envíe la cadena ‘fin’, ‘Fin’, o ‘FIN’, ‘fiN’, etc o cualquier combinación de mayúsculas y minúsculas que sean igual a ‘fin’ Envío:

3 5 a

aaaaa a a aaaaa

Cuando lo tengas ejecuta dos comandos nc para verificar que funciona. Haz capturas de pantalla de los comandos.

3 puntos puntos:

0.75 Escucha y recepción de datagramas

0.75 Generación de cuadrado

0.5 Envío de cuadrado

0.5 Finalización con fin y cierre de comunicaciones.

0.5 Captura de comandos nc funcionando (Se tiene que ver el comando)

NOTA: No se producen errores en la entrada, el programa siempre manda números mayores que 1 y todos los datos están bien.

# Ejercicio 2

Crea un programa que reciba como parámetro dos rutas y dos cadenas de texto

La primera ruta es la ruta de un fichero con mensajes

La segunda ruta es la ruta de un fichero con direcciones de correo

Las dos últimas cadenas representan el usuario de envío smtp y la contraseña. Ejemplo de ejecución

java examen.E2 data/mensajes.txt data/dirs.txt jorge.duenas@educa.org abc1234

El programa enviará el mensaje de la línea 1 del fichero mensajes a la dirección 1 del fichero dirs, , desde la cuenta jorge.duenas@educa.org y password abc1234.

Luego lo mismo con la línea dos, línea tres, etc.

Crea una clase LectorParaSpam que reciba en el constructor las dos rutas. El programa leerá la información de los dos ficheros en el método comenzarLectura. Esta clase implementa el patrón Observer. Al leer cada línea avisará a sus observadores con el correo y el mensaje (Produce un aviso por cada pareja email-mensaje).

Crea una clase EnviadorSpam, en el constructor recibe la cuenta que envía el correo y la contraseña. También recibe en el constructor un LectorParaSpam, observará la información que le pasa el LectorParaSpam en sus notificaciones, cada vez le notificará una dirección de destino y un mensaje.

Ten en cuenta que antes de llamar al método comenzarLectura de LectorParaSpam los observadores tienen que estar creados y observando.

3 puntos puntos:

0.5 Lectura de parámetros

0.5 Lectura de información de ficheros

0.5 Relacionar cada línea con la línea del otro fichero.

1.0 Observer

0.5 Envío de correo

NOTA: Utiliza funciones auxiliares para hacer cada subtrabajo.

NOTA: Los ficheros tienen el mismo número de líneas. Mételos en un ArrayList y recórrelos.

# Ejercicio 3

Crea un programa que reciba como parámetro un número de puerto por el que escuchar.

El programa será un servidor **mono**thread web.

Cuando el programa reciba una petición en la URL habrá dos parámetros. Ejemplo visito en el navegador http://localhost:8000/30/6/ Petición:

GET /30/6/ HTTP/1.1

Host: localhost:8000

El programa extraerá de la petición get los dos números, el primero será n y el segundo m.

El primer número representa el entero inicial, el segundo número representa la cantidad de primos.

El programa devolverá una página web con ul-li con los m (6 en el ejemplo) siguientes primos a n (30 en el ejemplo).

Ejemplo:

<ul>

<li>31</li> <li>37</li> <li>41</li> <li>43</li> <li>47</li>

<li>53</li> </ul>

Cada petición será atendida por una instancia de una clase PrimosHTTP que recibe en su constructor el socket ya asociado, el número inicial y la cantidad de primos. Esta clase se puede observar. Cada vez que encuentre un primo avisará a sus observadores.

Crea una clase Logger que se instancia una única vez en el main y observa a cada una de las instancias PrimosHTTP. Cada vez que se encuentre un primo, PrimosHTTP comunicará el evento, con esta informaciónla instancia de la clase Logger guarda en el fichero /var/log/primos.txt los números que se han ido encontrando.

Puntos

1 servidor multithread arranca en puerto pasado como parámetro

1 recibir petición GET y procesar los parámetros

1 generar cadena representado la web con los primero

1 construir petición HTTP de respuesta con el HTML

NOTA: Ejemplo de respuesta HTTP/1.1 200 OK