2a GUIA DE POO 1a parte

(Roberto Tecla Parra)

VÁZQUEZ MORENO MARCOS OSWALDO

2CM3

**Excepciones**

1. ***¿Qué llamadas a métodos se ponen dentro de un bloque try?***

R. Las llamadas a métodos que se ponen dentro del bloque try son aquellas que podrían lanzar una o mas excepciones.

1. ***El código dentro del bloque try como se escribe?***

R. try{  
 //código que podría lanzar una excepción  
}

1. ***Si una excepción es lanzada en cualquier parte del bloque try que instrucciones se ejecutan***

R. Se ejecutan las instrucciones que estaban antes de encontrar la excepción, y se pasa directamente al catch que recibe la excepción correspondiente, se ejecuta lo que haya dentro del catch y se pasa al finally realizando las instrucciones que se encuentren dentro de este.

1. ***Para lanzar una excepción que operador se usa?***

R. El operador throw

1. ***¿Cuál es el argumento para el operador throw?***

R. **throw** excepción o excepciones que puede lanzar;

1. ***¿Qué instrucción aborta el método y lanza la excepción a lo largo de la pila de llamadas?***

R. throw

***VII.- ¿Qué pasa sin ningún método en la pila de llamadas (ni siquiera main) maneja la***

***excepción.***

R. La máquina virtual java finaliza su ejecución con un mensaje de error

***a) ¿Cuál es el esquema que usa Java para el manejo de errores?***

R. try{  
 //código que podría lanzar una excepción  
}catch (tipoExcepción a){  
 //procesar el tipo de excepción  
 } catch (otroTipoExcepción a){  
 //procesar el tipo de excepción  
 }finally{  
 //código que se ejecuta después del try, se haya producido o no una excepción  
 }

b) ***¿Cuál es el esquema que usa Java para sincronizar el acceso a datos compartidos?***

R. Synchronized (objeto){

// En este método solo puede acceder un hilo a la vez

}

***b1) Cual es el modelo Cliente-Servidor?***

R. Es donde a través de dos programas, un cliente y un servidor, pueden intercambiar información, para esto, el cliente debe de conocer la ip del servidor y simplemente mandar una solicitud para establecer conexión con este.

***c) ¿Cuál es el modelo de componentes en Java?***

R. Component es una clase abstracta, que solo se pueden crear objetos pertenecientes a sus subclases y no al ella misma. Las subclases que representan los elementos comunes GUI son: Button,Checkbox, Choice, Label, List, Scrollbar, TextArea, y TextField, y cada uno de ellos tiene un comportamiento predefinido

***d) ¿Cuál es el modelo de eventos en Java?***

R. Propone la existencia de 3 tipos de objetos:

* **Events:** que pueden suceder y para los cuales es de interés registrar información de estado que detalle para cada tipo de evento los aspectos particulares del caso. Ejemplos de eventos de interés podrían ser: un evento de mouse o uno de teclado.
* **Sources:** eventos que los disparan, en general frente a una acción del usuario. Fuentes de eventos puede ser: la ventana de diálogo donde se visualiza la imagen
* **Listeners:** subscriben a las fuentes de eventos de su interés y son comunicados oportunamente por estas con el evento adecuado al caso.

***e) En Java ¿cuál es el equivalente a RPC (Remote procedure Call)***

R. Su equivalente es el Java RMI (Java Remote Method Invocation)

**Hilos**

***f) Que es un quantum?***

R. Es la cantidad de tiempo que un programa permite correr a un hilo

***g) Que es un cambio de contexto?***

R. Es el paso de la ejecución de un hilo al siguiente

***h) Que es una condición de competencia?***

R. Es la condición que permite una relativa prioridad de un hilo sobre otro.

***i) Que es una región critica?***

R. Porción de un programa en la que se accede a un recurso compartido que no debe ser accedido por más de un hilo en ejecución

***j) Que es exclusión mutua?***

R. Es la propiedad de permitir sólo a un subproceso adquirir los recursos a la vez.

***k) Que es espera ocupada?***

R. Estado donde un proceso repetidamente verifica la condición, como esperar la entrada por teclado o que el ingreso de una sección crítica este habilitado.

***l) Que es un multitarea?***

***R. Se*** define como la propiedad de que varios procesos compartan recursos comunes.

***m) Que es un planificador?***

R. Se refiere a la política a seguir de que hilo toma el control del procesador y cuando.

***Que es un método …***

***p) de clase?***

***R. Es*** la lógica de una clase, es decir, contienen el código que manipula el estado del objeto.

***q) de instancia***

***R. Es*** aquel que se invoca siempre sobre un mismo objeto de una clase.

***r) abstracto***

R. Es un método declarado pero no implementado, es decir, es un método del que sólo se escribe su nombre, parámetros y tipo devuelto pero no su código.

***s) sincronizado***

R. Es un tipo de método que permite la exclusión mutua de secciones críticas de código.

***t) remoto***

R. Objeto llamado por una clase que reside en otra.

***1 .- ¿Dentro de que método esta el cuerpo de un hilo?***

R. Run

***2.-¿Cuales son los estados por los que puede pasar un hilo durante su ciclo de vida?***

R. Recién nacido, listo, ejecutable, espera, dormido, suspendido, muerto, bloqueado.

***3.- ¿Como pasa un hilo del recién nacido estado al estado listo?***

R. Llamando el método start()

***4.- ¿Cómo pasa un hilo del estado listo al estado ejecutable?***

R. Al otorgarle el recurso del procesador para que pueda ejecutarse.

***5.- ¿Cómo pasa un hilo del estado ejecutable al estado en espera?***

R. Al llamar el método wait()

***6.- ¿Cómo pasa un hilo del estado ejecutable al estado dormido?***

R. Al invocar el método sleep()

***7.- ¿Cómo pasa un hilo del estado ejecutable al estado suspendido?***

R. Al invocar el método suspend()

***8.- ¿Cómo pasa un hilo del estado ejecutable al estado dormido?***

R. Al invocar el método sleep()

***9.- ¿Cómo pasa un hilo del estado ejecutable al estado muerto?***

R. Al invocar el método stop()

***10.- ¿Cómo pasa un hilo del estado en espera al estado listo?***

R. Al ejecutarse un notify() o un notifyAll()

***11.- ¿Cómo pasa un hilo del estado dormido al estado listo?***

R. Al acabarse el tiempo de siesta

***12.- ¿Cómo pasa un hilo del estado suspendido al estado listo?***

R. Al llamarse mediante resume()

***13.- ¿Cómo pasa un hilo del estado bloqueado al estado listo?***

R. Al completar la solicitud de E/S

***14.-Para que sirve el método start()?***

R. Permite a un hilo recién nacido ser inicializado con lo cual cambia a estado “listo”

***15.-Para que sirve el método stop()?***

R. Detiene un hilo poniéndolo es estado “muerto”

***16.- ¿Para qué sirve el método suspend()?***

R. Pasa un hilo de ejecutable a suspendido. Un hilo no consume recursos en este estado.

***17.- ¿Para qué sirve el método resume()?***

R. Permite cambiar el estado de un hilo de suspendido a ejecutable

***18.- ¿Para qué sirve el método wait()?***

R. Cambia el estado de un hilo de ejecutable a en espera, pasando a esperar en un pool de hilos.

***19.- ¿Para qué sirve el método notify()?***

R. Permite cambiar el estado de un hilo de en espera a ejecutable

***20.- ¿Para qué sirve el método notifyall()?***

R. Cambia a ejecutable todos los hilos que se encontraban en espera en ese momento de la ejecución

***21.- Si un hilo se lleva del estado ejecutable al estado bloqueado llamando el método suspend. Como puede llevarse de nuevo al estado ejecutable?***

R. Con el método resume()

***22.- ¿Que método debe llamar un hilo para pasar al estado bloqueado cierto numero de milisegundos y transcurridos estos volver al estado ejecutable?***

R. sleep()

***23.- ¿Cuáles son las dos causas por las que un hilo llega al estado muerto?***

R. Por la finalización del método run() o por la llamada del método stop()

***24.- Que método se use pare finalizar la vida de un hilo?***

R. stop()

***25.- Que método coloca el primer hilo de la cola de espera de un objeto en el estado ejecutable?***

R. notify()

***27.-Bajo que circunstancias se implementa en una clase la interfaz Runnable?***

R. Cuando la clase ya ha heredado de otra atributos

***28.-Que método(s) tiene que implementar una clase que implements la interfaz Runnable?***

R. run()

***29.-En Java que es un monitor?***

R. Son objetos que pueden ser utilizados por más de un hilo de ejecución. Son ejecutados con exclusión mutua.

***30.-Por que es necesario sincronizar el acceso a datos compartidos?***

R. Para garantizar la integridad de los datos.

**Red**

***¿Qué es un protocolo de comunicaciones?***

***32.1 ¿En comunicaciones que es un protocolo?***

R. Define la forma en la que los mensajes circulan en una red

***32.2 ¿Por qué su usan pilas de protocolos?***

R. Porque regulan la comunicación entre los participantes de esta

***32.3 -¿Qué es TCP/IP?***

R. Son las siglas de Protocolo de Control de Transmisión/Protocolo de Internet (en inglés Transmission Control Protocol/Internet Protocol), un sistema de protocolos que hacen posibles servicios Telnet, FTP, E-mail, y otros entre ordenadores que no pertenecen a la misma red.

***32.4 -¿Qué es una dirección IP?***

R. Es una etiqueta numérica que identifica, de manera lógica y jerárquica, a una interfaz (elemento de comunicación/conexión) de un dispositivo (habitualmente una computadora) dentro de una red que utilice el protocolo IP.

* 1. ***¿Qué es un puerto?***

R. Un puerto es una interfaz a través de la cual se pueden enviar y recibir los diferentes tipos de datos.

* 1. ***¿Qué números de puertos están reservados?***

R. Los puertos inferiores al 1024 son puertos reservados por los sistemas operativos y usados por "protocolos bien conocidos" como por ejemplo HTTP. Los comprendidos entre 1024 (0400 en hexadecimal) y 49151 (BFFF en hexadecimal) son denominados "registrados" y pueden ser usados por cualquier aplicación.

**Sockets**

***33.- ¿Qué es un socket?***

R. Un socket es un punto de comunicación por el cual un proceso puede emitir o recibir información. Los sockets son una innovación del UNIX de Berkeley y permiten tratar una conexión a la red como otro flujo de bytes del que se puede leer o escribir. Los sockets esconden del programador los detalles de bajo nivel de la red.

***35.- ¿Cuál es el nombre de la clase de un socket cliente?***

R. Socket

***36.- ¿Cuáles son los 2 argumentos mas usuales del constructor de un socket cliente?***

R. El nombre de la máquina en donde estamos intentando abrir la conexión(host) y el puerto (número) del servidor que está corriendo sobre el cual nos queremos conectar.

***37.- ¿Cuál es el Ciclo de Vida de un socket cliente?***

R.

1. El nuevo socket es creado usando un constructor Socket().
2. El socket intenta conectarse a un host remoto.
3. Una vez que la conexión es establecida los hosts local y remoto obtienen flujos de entrada y salida para el socket y usan esos flujos para enviarse datos uno a otro. Esta conexión es full-duplex, ambos hosts pueden enviar y recibir datos simultáneamente.

Cuando la transmisión de datos está completa, uno o ambos cierran la conexión.

***38.- ¿Cuál es el nombre de la clase de un socket servidor?***

R. ServerSocket

***39.- ¿Cuál es el argumento mas usual del constructor de un socket servidor?***

R. El puerto a que el cliente establecerá la conexión. Cuando el cliente solicite una conexión, el servidor abrirá la conexión socket con el método accept().

***40.- ¿Cuál es el ciclo de vida de un socket servidor?***

R.

1. Un ServerSocket nuevo es creado en un puerto particular usando el constructor ServerSocket().
2. El ServerSocket() escucha los intentos de conexión entrantes en ese puerto usando su método accept(). Accept() se bloquea hasta que un cliente intenta realizar una conexión, en dicho punto accept() regresa un socket que conecta el cliente al servidor.
3. Dependiendo del tipo de servidor, el método getInputStream() o el método getOutputStrem del socket será llamado. Para obtener flujos de entrada y salida que comunican con el cliente.
4. El servidor y el cliente interactúan de acuerdo a un protocolo establecido hasta que es tiempo de cerrar la conexión.
5. El servidor el cliente, o ambos, cierran la conexión.

El servidor regresa al paso II, y espera la siguiente conexión.

***41.- ¿Para qué sirve el método accept?***

R. accept() es una rutina que crea un nuevo Socket para atender a un cliente que se ha conectado a ese servidor. Accept() se bloquea hasta que un cliente intenta realizar una conexión.***42.- ¿Qué tipo de valor regresa el método accept y para que sirve dicho valor?***

R. accept() regresa un Socket que conecta el cliente al servidor.

***43 .- ¿Que métodos se usan para obtener flujos de E/S de un socket?***

R. Los métodos getInputStream() o getOutputStrem

**RMI**

***44.-¿Para que sirve RMI?***

R. permite que objetos Java en diferentes hosts se comuniquen unos con otros.

***45.- ¿Con qué se programa a más alto nivel con sockets o con RMI?***

R. Sockets

***46.- ¿Con RMI la programación está centrada en la invocación de métodos remotos, en el caso de los sockets en que se centra la programación?***

R. En el uso de IP’s y de host, los cuales crean un flujo de datos para poder comunicar al cliente y al servidor.

***47.- Escriba los nombres de las capas del protocolo RMI del lado del cliente.***

R.

1. Programa Cliente
2. Stub
3. Nivel de referencia remota
4. Nivel de transporte

4***8.-Escriba los nombres de las capas del protocolo RMI del lado del servidor.***

R.

1. Programa Cliente
2. Esqueleto
3. Nivel de referencia remota
4. Nivel de transporte

***49.- ¿Para qué sirve la capa de referencia remota?***

R. Es utilizada por la capa Stub y skeleton, es la responsable del funcionamiento independientemente de la capa de transporte que se esté utilizando. Esta capa proporciona un objeto RemoteRef que representa la conexión con el objeto remoto. El objeto stub utiliza el método invoke() de RemoteRef para enviar las llamadas a los métodos del objeto remoto.

***50.- ¿Para qué sirve la capa Stub?***

R. es responsable de:

* -Iniciar una llamada al objeto remoto
* -Ordenar los argumentos del flujo
* -Informar a la capa de referencia remota que la llamada debe ser hecha
* -Retornar el valor o excepción de un flujo
* -Informar a la capa de referencia remota que la llamada está completa.

***51.- ¿Para qué sirve la capa del esqueleto?***

R. es responsable de:

* -Decodificar los datos del flujo de datos
* -Hacer la llamada de la implementación del objeto remoto actual
* -Retornar el estado de la llamada o excepción dentro de un flujo.

***56.- ¿Para qué sirven los métodos bind y rebind de la clase Naming?***

R. bind(url, object) Liga un nombre a un objeto remoto.

Rebind(url, object) Similar al bind() pero reemplaza la asociación hecha.

***57.- ¿Para qué sirve el método lookup de la clase Naming?***

R. lookup(url) Retorna un objeto remoto (un stub) asociado con el url.

***58.- ¿Cual es el nombre del programa que se usa para generar los stubs y los esqueletos para los objetos remotos?***

R. Para generar estos "Stubs" y "Skeletons" Java ofrece una herramienta llamada rmic

***59.- ¿Cuál es el nombre del programa que se usa para registrar un objeto remoto?***

R. rmiregistry

**HTTP**

***60.- ¿Qué es HTTP?***

R. HTTP (Hipertext Transport Protocol) es el protocolo para comunicación entre servidores y navegadores de web. Este es un protocolo sin estado (stateless), cuyas conexiones usan el protocolo TCP\IP para transferir datos los cuales codifica usando MIME (Multiporpose Internet Mail Extensions).

***61.- ¿Para qué sirven las formas de html?***

R. Procesar la entrada del usuario y captar diferentes tipos de datos.

***62.- ¿Qué es CGI?***

R. CGI (Common GateWay Interface) se usa para generar páginas web dinámicamente, el navegador invoca un programa en el servidor que crea una nueva página.

***63.- ¿Cuándo se usa el método POST en una petición http?***

R. Con POST el navegador envía los encabezados usuales y los termina con una línea en blanco, y entonces envía la cadena de consulta. La cadena de consulta se le pasa al programa CGI en la entrada estándar.

***64.- ¿Cuándo se usa el método GET en una petición http?***

R. Si la forma usa GET el navegador agrega la cadena de consulta al url y lo envía al servidor.

***64.1- ¿En CGI qué es una cadena de consulta?***

R. Este término generalmente se utiliza para hacer referencia a una interacción con una base de datos. Es la parte de una URL que contiene los datos que deben pasar a aplicaciones web como los programas CGI.

***65.- ¿Qué es un Servlet?***

R. Los servlets son el equivalente de los applets, pero del lado del servidor al cuál hacen extensible. El servlet API es fruto de la evolución del sistema de programación CGI. Un servidor se comunica con los servlet invocando métodos en la interfaz de java.servlet.Servlet: init(), service(), y destroy().

**JavaBeans**

***71. ¿Cuáles son las tres partes de un JavaBean?***

R. Propiedades: Los atributos que contiene. Métodos: Se establecen los métodos get y set para acceder y modificar los atributos. Eventos: Permiten comunicar con otros JavaBeans.

***72. ¿Qué es una propiedad de un JavaBean?***

R. Una propiedad es un atributo del JavaBean que afecta a su apariencia o a su conducta. Por ejemplo, un botón puede tener las siguientes propiedades: el tamaño, la posición, el título, el color de fondo, el color del texto, si está o no habilitado, etc.

***73. ¿Qué es una propiedad indexada?***

R. Una propiedad indexada representa un array de valores.

***74. ¿Qué es una propiedad encadenada?***

R. Los objetos de una clase que tiene una propiedad ligada notifican a otros objetos (listeners) interesados, cuando el valor de dicha propiedad cambia, permitiendo a estos objetos realizar alguna acción. Cuando la propiedad cambia, se crea un objeto (event) que contiene información acerca de la propiedad (su nombre, el valor previo y el nuevo valor), y lo pasa a los otros objetos (listeners) interesados en el cambio.

***75. ¿Qué es una propiedad restringida?***

R. Una propiedad restringida es similar a una propiedad encadenada salvo que los objetos (listeners) a los que se les notifica el cambio del valor de la propiedad tienen la opción de vetar (veto) cualquier cambio en el valor de dicha propiedad. (Este tipo de propiedad no se estudiará en este capítulo).

***76. ¿Qué es un un método getter?***

R. getter method: lee el valor de la propiedad.

***77. ¿Qué es un un método setter?***

R. Setter method: cambia el valor de la propiedad.

***78- ¿Qué es el mecanismo de introspección?***

Permite analizar a la herramienta de programación o IDE como trabaja el bean.

**JDBC**

***¿Qué es un SGBD?***

R.Un Sistema de Gestión de Bases de Datos (SGBD) es un conjunto de programas que permiten el almacenamiento, modificación y extracción de la información en una base de datos, además de proporcionar herramientas para añadir, borrar, modificar y analizar los datos.

***¿Qué es una tabla?***

R. Una tabla es una relación (modelo relacional) que está compuesta por registros (filas o también llamadas tuplas) y columnas (campos).

***¿Qué es un registro?***

R. Un registro (también conocido como tupla o fila) es un conjunto de elementos (campos) que tienen un significado coherente en la relación o tabla.

***¿Qué es un campo?***

R. Un campo es la mínima unidad de información a la que se puede acceder; un campo o un conjunto de ellos forman un registro, donde pueden existir campos en blanco.

***¿Qué es un metadato?***

R. Un metadato define a un tipo de dato o tipos de datos que se va a ocupar en la base de datos.

***¿Qué es un diccionario de datos?***

R. Es un listado organizado de todos los datos pertinentes al sistema con definiciones precisas y rigurosas para que tanto el usuario como el analista tengan un entendimiento en común de todas las entradas, salidas, componentes y cálculos de la base de datos.

***¿Qué significan las siglas CRUD?***

R. Create, Read, Update y Delete.

***¿Qué significan las siglas SQL?***

R. Structured Query Language.

***¿Para qué sirve SQL?***

R. Permite especificar diferentes tipos de operaciones en bases de datos relacionales.

***En SQL ¿qué es resultado de una consulta?***

R. La recuperación, de forma sencilla, de la información de la base de datos sobre la cual se esté trabajando, así como hacer cambios en ella.

***79.- ¿Qué es ODBC?***

Open DataBase Connectivity (ODBC) es un estándar de acceso a las bases de datos desarrollado por SQL Access Group (SAG) en 1992. El objetivo de ODBC es hacer posible el acceder a cualquier dato desde cualquier aplicación, sin importar qué sistema de gestión de bases de datos (DBMS) almacene los datos.

***80.- ¿Qué es JDBC?***

R. Java Database Connectivity, más conocida por sus siglas JDBC,es una API que permite la ejecución de operaciones sobre bases de datos desde el lenguaje de programación Java, independientemente del sistema operativo donde se ejecute o de la base de datos a la cual se accede, utilizando el dialecto SQL del modelo de base de datos que se utilice.

***81.- ¿Cuáles son las partes de URL en JDBC?***

R. El driver, el dominio, el puerto y finalmente la base de datos a utilizar.

***82.- ¿Cómo se realiza la conexión con JDBC a una Base de Datos?***

R. Mediante el método getConnectio(url) de la clase Connection.

***83.- ¿Cómo se envía una instrucción de SQL a la Base de Datos?***

R. Con los métodos de la clase Statement.

***84.- ¿Para qué sirve la clase DriveManager?***

R. Para cargar un driver (SGBD) a utilizar.

***85.- ¿Para qué sirve la clase Statement?***

R. Para enviar instrucciones de SQL a la BD.

***86.- ¿Para qué sirve la clase ResultSet?***

R. Para almacenar los resultados de una consulta.

**2a GUIA de POO 2a parte**

**(Roberto Tecla Parra)**

**Hilos**

***1.-Codificar una clase llamada CuentaSegundos que implemente la interfaz Runnable:***

***-Que cada segundo incremente en una unidad un contador y muestre el valor de dicho contador en una etiqueta.***

***-Que cuando llegue a un valor límite (que se establece cuando se crea un objeto) deje de incrementar el contador y de actualizar el valor mostrado.***

|  |
| --- |
| import java.awt.\*;  import javax.swing.\*;  import java.awt.event.\*;  import java.util.\*;  import java.io.\*;  import java.lang.Thread.\*;  import java.lang.\*;  public class CuentaSegundos extends JFrame implements Runnable{  Panel p;  JLabel contador;  Thread hilo;  int i=0;  public CuentaSegundos(){  p=new Panel();  contador=new JLabel("0");  p.add(contador);  add(p,"Center");  setDefaultCloseOperation(WindowConstants.EXIT\_ON\_CLOSE);  setSize(100, 100);  setVisible(true);  hilo = new Thread(this);  hilo.start();  }    public void run(){  while(i<10){  i++;  contador.setText(""+i);  try{  Thread.sleep(400);  } catch (Exception e) {  return;  }  }  }    public static void main(String[] args) {  new CuentaSegundos();  }  } |

***2.-Codificar una clase llamada Ticker***

***Que mueva las letras de un texto de izquierda a derecha (o de derecha a izquierda). Se puede quitar la primer letra y pegarla (con +) al final y repetir esto de forma periódica.***

|  |
| --- |
| import java.awt.\*;  import javax.swing.\*;  import java.awt.event.\*;  import java.util.\*;  import java.io.\*;  import java.lang.Thread.\*;  import java.lang.\*;  public class ticker extends JFrame implements Runnable{  Panel p;  JLabel p\_movida;  Thread hilo;  public ticker(){  p=new Panel();  p\_movida=new JLabel("Holi Crayoli");  p.add(p\_movida);  add(p,"Center");  setDefaultCloseOperation(WindowConstants.EXIT\_ON\_CLOSE);  setSize(100, 100);  setVisible(true);  hilo = new Thread(this);  hilo.start();  }    public void run(){  String dat = p\_movida.getText();  int l=dat.length();  for(; ; ){  int i=0;  while(i < l){  String pal=dat.substring(0, (l-i));  String pal2=dat.substring((l-i), l);  i++;  p\_movida.setText(""+pal2+pal);  try{  Thread.sleep(600);  } catch (Exception e) {  return;  }  }  }  }    public static void main(String[] args) {  new ticker();  }  } |

**Sockets**

***Problema 1.-Codificar un servidor que cuando un cliente se conecte le envié la cadena “Hola Mundo”***

***Cliente***

//package cliente;

import java.io.ObjectInputStream;

import java.io.ObjectOutputStream;

import java.util.logging.Level;

import java.util.logging.Logger;

import java.io.IOException;

import java.net.Socket;

import java.applet.\*;

import java.awt.event.\*;

import java.awt.\*;

import javax.swing.\*;

import java.util.\*;

public class Cliente extends JPanel implements ActionListener {

Label l;

Button b;

public Cliente(){

b=new Button ("Conectar");

l=new Label ("\*\*\*\*\*\*\*\*");

add(b);

add(l);

b.addActionListener(this);

}

public void actionPerformed (ActionEvent e){

try{

Socket cliente =new Socket("localhost",4500);

ObjectInputStream entrada=new ObjectInputStream(cliente.getInputStream());

String mensaje2 =(String)entrada.readObject();

l.setText(mensaje2);

System.out.println("Mensaje desde el servidor:"+mensaje2);

}

catch(ClassNotFoundException ex){

Logger.getLogger(Cliente.class.getName()).log(Level.SEVERE,null,ex);

}

catch(IOException ex){

Logger.getLogger(Cliente.class.getName()).log(Level.SEVERE,null,ex);

}

}

public static void main(String[] args){

JFrame f=new JFrame("Oraculo");

f.add("Center", new Cliente());

f.setDefaultCloseOperation(WindowConstants.EXIT\_ON\_CLOSE);

f.setSize(900, 200); f.setVisible(true);

}

}

***Servidor***

//package Servidor;

import java.net.ServerSocket;

import java.io.ObjectInputStream;

import java.io.ObjectOutputStream;

import java.io.IOException;

import java.net.Socket;

import java.util.logging.Level;

import java.util.logging.Logger;

import java.applet.\*;

import java.awt.event.\*;

import java.awt.\*;

import javax.swing.\*;

import java.util.\*;

public class Servidor{

public static void main(String[] args){

String cadena="Hola mundo";

for ( ; ; ){

try{

ServerSocket servidor2=new ServerSocket(4500);

Socket clienteNuevo = servidor2.accept();

ObjectOutputStream resp = new ObjectOutputStream(clienteNuevo.getOutputStream());

resp.writeObject(cadena);

clienteNuevo.close();

servidor2.close();

}

catch(Exception ex){

Logger.getLogger(Servidor.class.getName()).log(Level.SEVERE, null, ex);

}

}

}

}

***Problema 2.-Codificar un servidor que cuando un cliente se conecte lea el nombre del cliente y le envié a dicho cliente el nombre del ultimo usuario que se conecto.***

**Cliente**

//package cliente;

import java.io.ObjectInputStream;

import java.io.ObjectOutputStream;

import java.util.logging.Level;

import java.util.logging.Logger;

import java.io.IOException;

import java.net.Socket;

import java.applet.\*;

import java.awt.event.\*;

import java.awt.\*;

import javax.swing.\*;

import java.util.\*;

public class ClienteS extends JPanel implements ActionListener {

Label l;

Button b;

TextField t;

String texto;

public ClienteS(){

t=new TextField(15);

b=new Button ("Conectar");

l=new Label ("\*\*\*\*\*\*\*\*\*\*");

add(t);

add(b);

add(l);

b.addActionListener(this);

}

public void actionPerformed (ActionEvent e){

texto=t.getText();

try{

Socket cliente =new Socket("localhost",4500);

ObjectOutputStream mensaje= new ObjectOutputStream(cliente.getOutputStream());

mensaje.writeObject(texto);

ObjectInputStream entrada=new ObjectInputStream(cliente.getInputStream());

String mensaje2 =(String)entrada.readObject();

l.setText(mensaje2);

System.out.println("Mensaje desde el servidor:"+mensaje2);

}

catch(ClassNotFoundException ex){

Logger.getLogger(ClienteS.class.getName()).log(Level.SEVERE,null,ex);

}

catch(IOException ex){

Logger.getLogger(ClienteS.class.getName()).log(Level.SEVERE,null,ex);

}

}

public static void main(String[] args){

JFrame f=new JFrame("Conectar");

f.add("Center", new ClienteS());

f.setDefaultCloseOperation(WindowConstants.EXIT\_ON\_CLOSE);

f.setSize(900, 200); f.setVisible(true);

}

}

**RMI**

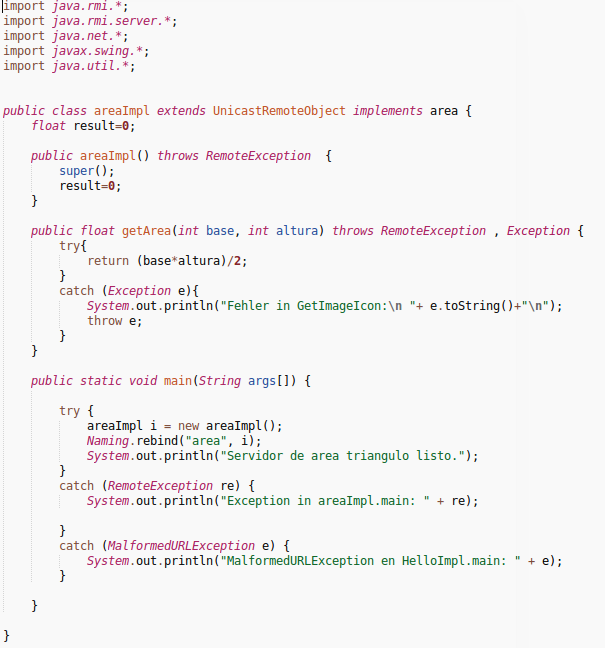
En cada problema codificar la interfaz remota, la clase del objeto remoto (servidor), la clase del cliente y en los métodos remotos poner los parámetros que se necesiten.

***Problema 1.-Escribir el código de un método remoto que calcule el área de un triangulo.***

**Interfaz remota:**



**Servidor:**

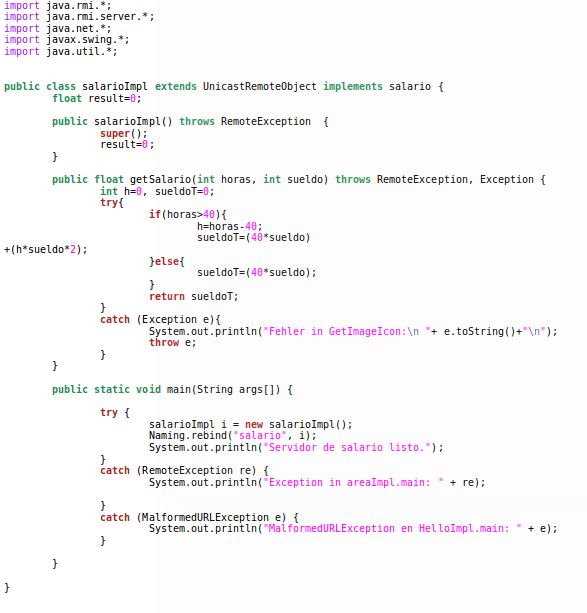


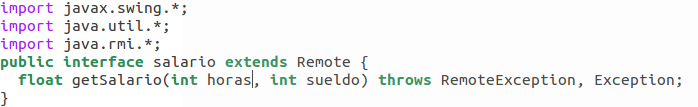
**Cliente:**



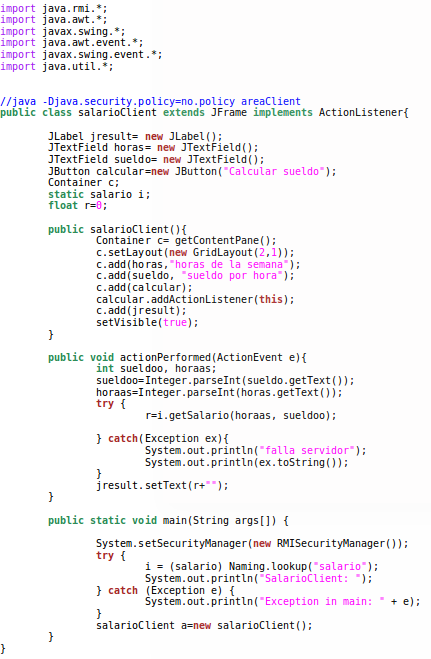
***Problema 2.-Escribir el código de un método remoto que calcule el salario semanal de un trabajador que gana el salario mínimo en base a los horas trabajadas por dicho trabajador durante la semana. Considere 40 horas normales de trabajo a la semana y que las horas extras (las que se trabajan después de transcurridas las 40) se pagan dobles.***

**Servidor:**



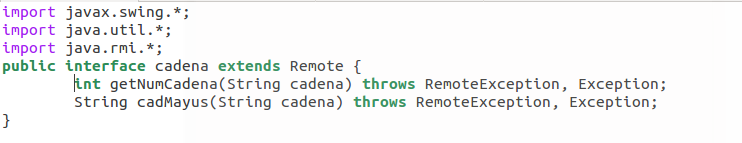
**Interfaz remota:**

**Cliente:**

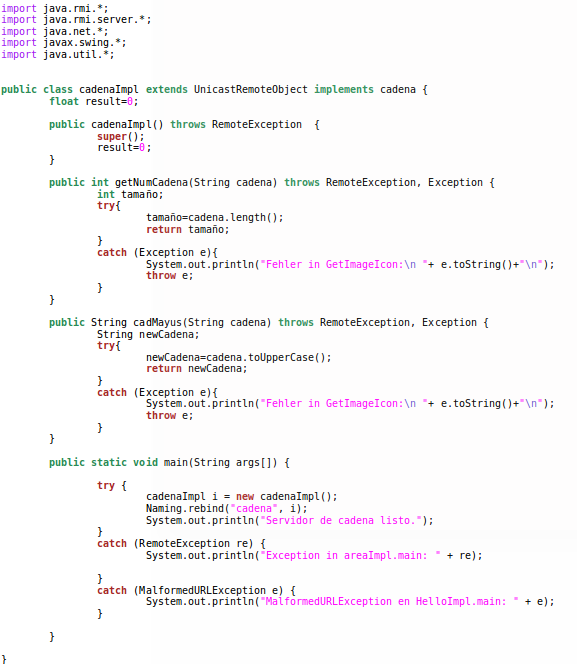


***Problema 3.-Escribir el código de un método remoto que convierta una cadena a mayúsculas y otro método remoto que obtenga la longitud de una cadena.***

**Interfaz remota:**



**Servidor:**



**Cliente:**

**Javabeans**

Para resolver los problemas de abajo tenga en cuenta lo siguiente:

-Un Javabean tiene que tener un **constructor por defecto** (sin argumentos)

-Un Javabean tiene que tener **persistencia**, es decir, debe implementar la interfaz Serializable

***Problema 1-Codificar un JavaBean llamado Viaje que tenga las siguientes propiedades:***

***Origen (lectura/escritura), Destino (lectura/escritura), y Costo (lectura/escritura)***

import java.io.Serializable;

public class Viaje implements Serializable{

private String origen;

private String destino;

private String Costo;

public Viaje () {

}

public Viaje(String origen, String destino, String costo){

this.origen=origen;

this.destino=destino;

this.costo=costo;

}

public String getOrigen(){

return origen;

}

public String getDestino(){

return destino;

}

public String getCosto(){

return costo;

}

public void setOrigen(String origen){

this.origen=origen;

}

public void setDestino(String destino){

this.destino=destino;

}

public void setCosto(String costo){

this.costo=costo;

}  
}

***Problema 2-Codificar un JavaBean llamado Película que tenga las siguientes propiedades:***

***Titulo ( lectura/escritura), Director ( lectura/escritura), y Año ( lectura/escritura)***

import java.io.Serializable;

public class Pelicula implements Serializable{

private String titulo;

private String director;

private int año;

public Pelicula () {

}

public Pelicula(String titulo, String director, int año){

this.titulo=titulo;

this.director=director;

this.año=año;

}

public String getTitulo(){

return titulo;

}

public String getDirector(){

return director;

}

public int getAño(){

return año;

}

public void setTitulo(String titulo){

this.titulo=titulo;

}

public void setDirector(String director){

this.director=director;

}

public void setAño(int año){

this.año=año;

}  
  
}

***Problema 3.-Codificar un JavaBean llamado Perro que tenga las siguientes propiedades:***

***Nombre ( lectura/escritura), Raza ( lectura/escritura), Edad ( lectura/escritura), Dueño ( lectura/escritura) , Genero ( lectura/escritura), y AñosRestantesDeVida ( solo lectura). Suponga que si la raza es:***

***Pastor Aleman el perro vive 15 años***

***Boxer el perro vive 11 años***

***Terrier el perro vive 8 años,***

***Doberman el perro vive 13 años***

***Para las otras razas vive 14 años***

import java.io.Serializable;

public class Perro implements Serializable{

private String nombre;

private String raza;

private int edad;

private String dueño;

private String genero;

public Perro () {

}

public Pelicula(String nombre, String raza, int edad, String dueño, String genero){

this.nombre=nombre;

this.raza=raza;

this.edad=edad;

this.dueño=dueño;

this.genero=genero;

}

public String getNombre(){

return nombre;

}

public String getRaza(){

return raza;

}

public int getEdad(){

return edad;

}

public String getDueño(){

return dueño;

}

public String getGenero(){

return genero;

}

public void setNombre(String nombre){

this.nombre=nombre;

}

public void setRaza(String raza){

this.raza=raza;

}

public void setEdad(int edad){

this.edad=edad;

}

public void setDueño(String dueño){

this.dueño=dueño;

}

public void setGenero(String genero){

this.genero=genero;

}

public int getAñosRestantesDeVida(){

if(raza==”Pastor Aleman”)

return 15-edad;

else if(raza==”Boxer”)

return 11-edad;

else if(raza==”Terrier”)

return 8-edad;

else if(raza==”Doberman”)

return 13-edad;

else

return 14-edad;

}  
  
}

***Problema 4.-En el siguiente JavaBean (Car) determinar cuáles son sus propiedades y si son de lectura o escritura o ambas cosas.***

import java.awt.\*;

import java.awt.event.\*;

import java.io.\*;

public class Car implements Serializable{

private String make, model; //propiedades de lectura y escritura

double price;//propiedad solo de lectura

public Car() { this ("", "", 0.0); } //constructor

public String getMake() { return make; } //lectura

public void setMake (String m) { make = m; } //escritura

public String getModel() { return model; } //lectura

public void setModel (String mo) { model = mo; } //escritura

public double getPrice() { return price; } //lectura

}

**Servlets**

Para cada problema de abajo escribir el código **HTML** del **formulario** con los campos de entradas que sean necesarios y el botón de enviar.

***1.-Codificar un Servlet que calcule el área de un circulo a partir de los datos de un formulario enviados por el navegador y envié el resultado de dicho calculo al cliente como texto plano o HTML. Escribir el código HTML del formulario con un campo de entrada para el radio y el botón de enviar.***

**Código formulario (HTML)**

<html>

<body>

<form method="POST" action="/servlets/Area">

radio=<input type="text" name="radio ">

<input type="submit" value="enviar">

</form>

</body>

</html>

**Código Servlet**

public class Area extends HttpServlet {

public void init(ServletConfig config) throws ServletException {

System.out.println("init");

}

public void doPost(HttpServletRequest req, HttpServletResponse res)

throws ServletException, IOException {

res.setContentType("text/html");

PrintWriter out = res.getWriter();

String n = req.getParameter("radio");

float rad=Float.parseFloat(n);

float area=3.1416\*rad\*rad;

out.println("<HTML>");

out.println("<HEAD><TITLE>Area circulo</TITLE></HEAD>");

out.println("<BODY>");

out.printf("<H1>El area del circulo de radio %d es %.2f </H1>",rad,area);

out.println("</BODY></HTML>");

}

public void destroy() {

System.out.println("destroy");

}

public String getServletInfo() {

return null;

}

public ServletConfig getServletConfig() {

return null;

}

}

***2.-Codificar un Servlet que calcule el perímetro de un pentágono no regular a partir de los datos de un formulario enviados por el navegador y envié el resultado de dicho calculo al cliente como texto plano o HTML Escribir el código HTML del formulario con un campo de entrada para cada lado del pentágono no regular y el botón de enviar.***

**Código formulario (HTML)**

<html>

<body>

<form method="POST" action="/servlets/Perimetro">

lado1=<input type="text" name="lado1 ">

lado2=<input type="text" name="lado2 ">

lado3=<input type="text" name="lado3 ">

lado4=<input type="text" name="lado4 ">

lado5=<input type="text" name="lado5 ">

<input type="submit" value="enviar">

</form>

</body>

</html>

**Código Servlet**

public class Perimetro extends HttpServlet {

public void init(ServletConfig config) throws ServletException {

System.out.println("init");

}

public void doPost(HttpServletRequest req, HttpServletResponse res)

throws ServletException, IOException {

res.setContentType("text/html");

PrintWriter out = res.getWriter();

String n1 = req.getParameter("lado1");

String n2 = req.getParameter("lado2");

String n3 = req.getParameter("lado3");

String n4 = req.getParameter("lado4");

String n5 = req.getParameter("lado5");

int l1=Integer.parseInt(n1);

int l2=Integer.parseInt(n2);

int l3=Integer.parseInt(n3);

int l4=Integer.parseInt(n4);

int l5=Integer.parseInt(n5);

int perimetro=l1+l2+l3+l4+l5;

out.println("<HTML>");

out.println("<HEAD><TITLE>Perimetro pentagono</TITLE></HEAD>");

out.println("<BODY>");

out.printf("<H1>El perimetro del pentágono es de %d </H1>",perimetro);

out.println("</BODY></HTML>");

}

public void destroy() {

System.out.println("destroy");

}

public String getServletInfo() {

return null;

}

public ServletConfig getServletConfig() {

return null;

}

}

***3.-Codificar un Servlet que envié al cliente la capital del país que el usuario escribió en un formulario. Escribir el código HTML del formulario con un campo de entrada para el pais y el botón de enviar.***

**Código formulario (HTML)**

<html>

<body>

<form method="POST" action="/servlets/Capital">

pais=<input type="text" name="pais ">

<input type="submit" value="enviar">

</form>

</body>

</html>

**Código Servlet**

public class Capital extends HttpServlet {

public void init(ServletConfig config) throws ServletException {

System.out.println("init");

}

public void doPost(HttpServletRequest req, HttpServletResponse res)

throws ServletException, IOException {

res.setContentType("text/html");

PrintWriter out = res.getWriter();

String paises[] = {“Mexico”,”Brasil”,”Estados Unidos”,”Francia”,”España”};

String capitales[]={“Mexico,D.F.”, “Brasilia”,”Washington D.C.”, “Paris”,”Madrid”};

String n=req.getParameter("pais");

out.println("<HTML>");

out.println("<HEAD><TITLE>Capital paises</TITLE></HEAD>");

out.println("<BODY>");

out.printf("<H1>La capital del pais %s es: </H1>",n);

for(int i=0; i<paises.length;i++)

if(n.equals(paises[i]))

out.println(“”+capitales[i]);

if(i==(paises.length-1))

out.printl(“No se”);

out.println("</BODY></HTML>");

}

public void destroy() {

System.out.println("destroy");

}

public String getServletInfo() {

return null;

}

public ServletConfig getServletConfig() {

return null;

}

}

***4.-Codificar un Servlet que calcule el promedio de las calificaciones de un alumno en las siguientes materias: física, matemáticas , química , y español a partir de los datos de un formulario enviados por el navegador y envié el resultado de dicho calculo al cliente como texto plano o HTML. Escribir el código HTML del formulario con un campo de entrada para cada materia y el botón de enviar.***

**Codigo formulario (HTML)**

<html>

<body>

<form method="POST" action="/servlets/Promedio">

fisica=<input type="text" name="fisica ">

matematicas=<input type="text" name="matematicas ">

quimica=<input type="text" name="quimica ">

español=<input type="text" name="español ">

<input type="submit" value="enviar">

</form>

</body>

</html>

**Código Servlet**

public class Promedio extends HttpServlet {

public void init(ServletConfig config) throws ServletException {

System.out.println("init");

}

public void doPost(HttpServletRequest req, HttpServletResponse res)

throws ServletException, IOException {

res.setContentType("text/html");

PrintWriter out = res.getWriter()

String n1 = req.getParameter("fisica");

String n2 = req.getParameter("matematicas");

String n3 = req.getParameter("quimica");

String n4 = req.getParameter("español");

float c1=Float.parseFloat(n1);

float c2=Float.parseFloat(n2);

float c3=Float.parseFloat(n3);

float c4=Float.parseFloat(n4);

float c5=Float.parseFloat(n5);

float promedio=(c1+c2+c3+c4+c5)/5;

out.println("<HTML>");

out.println("<HEAD><TITLE>Promedio alumno</TITLE></HEAD>");

out.println("<BODY>");

out.printf("<H1>El promedio del alumno es %.2f </H1>",promedio);

out.println("</BODY></HTML>");

}

public void destroy() {

System.out.println("destroy");

}

public String getServletInfo() {

return null;

}

public ServletConfig getServletConfig() ;