

Ejercicio 1. Teoría (5 puntos) Indica la respuesta correcta en la siguiente tabla.

- **No se valorarán** las respuestas fuera de la tabla, cualquier otra indicación o las que estando dentro de la tabla no sean claramente legibles.
- **Valor** respuesta correcta 0,50 puntos. Valor respuesta errónea -0,15 puntos.

1	2	3	4	5	6	7	8	9	10
B	C	A	A	A	C	B	A	D	

1. ¿Dónde se almacenan las claves de seguridad?

- a) Keysave
- ☒ b) Keystore
- c) Securesave
- d) Securestore

2. ¿Qué es una excepción?

- a) Una determinada condición que siempre se cumple
- b) Un evento que ocurre durante la ejecución de un programa e interrumpe su ejecución
- ☒ c) Un evento que ocurre durante la ejecución de un programa e interrumpe el flujo normal de las instrucciones

3. ¿Qué máscara de entrada se corresponde con el código postal?

- ☒ a) [0-9]{5}
- b) [0-9]{3}
- c) 0[0-9]{5}
- d) [0-9]*{6}

4. ¿Qué medida de seguridad garantiza que la información sólo pueda ser modificada por las entidades autorizadas?

- ☒ a) Confidencialidad
- b) Integridad
- c) No repudio
- d) Disponibilidad

5. Todos son protocolos seguros.

- ☒ a) SSL, HTTPS, FTPS
- b) Telnet, socket, TLS
- c) SFTP, socket, SSL

6. El método que devuelve un algoritmo criptográfico de un proveedor dado es:

- a) accept()
- b) init()
- ☒ c) getInstance()
- d) getAlgorithm()

7. El método URL.openConnection() realiza la siguiente tarea:

- a) Devuelve un objeto InputStream
- ☒ b) Abre una conexión con el recurso
- c) Devuelve un objeto URLConnection

8. Es una característica del protocolo SMTP:

- ☒ a) Comunicarse por el puerto 25
- b) Permite la descarga de ficheros
- c) Organiza la recepción de mensajes.

9. La criptografía asimétrica:

- a) utiliza claves diferentes para encriptar y desencriptar
- b) Una clave es conocida.
- c) Puede encriptar con la clave pública y privada.
- ☒ d) Todas son correctas

10. Un resumen (función Hash) de mensaje permite: garantizar la integridad de un mensaje. ¿Verdadero o falso?

- a) Garantizar la integridad de un mensaje
- b) Hacer el mensaje más pequeño
- c) Garantizar la confidencialidad de un mensaje
- d) Garantizar que el mensaje será eliminado después de su recepción.

Ejercicio 2. Servidor HTTP. (5 puntos) Crea un servidor HTTP con las siguientes condiciones:

- El servidor aceptará peticiones en el puerto 8066.
- Contestará a las siguientes URLs
 - <http://localhost:8066>. “Bienvenido a mi web”
 - <http://localhost:8066/inscripcion> “Inscripción – En proceso de construcción”
 - Cualquier otra dentro de <http://localhost:8066> Mensaje de error
- Acceso concurrente al servidor (multihilo)
- Tratamiento adecuado de posibles excepciones
- Estados y cabeceras a incluir: Línea de estado, tipo de contenido (Content-Type) y longitud del mensaje (Content-Length)
- **Escribe el código y por medio de breves comentarios su funcionamiento.**

```

class ServidorHTTP { // Definimos la clase principal que definirá y ejecutará nuestro servidor HTTP

    ServidorHTTP() {

    } // Creamos un constructor de clase vacío

    public static void main(String[] args) throws IOException, Exception { // Definimos el programa principal y
        // controlamos las posibles excepciones, en este caso la excepción de los datos de entrada y de salida y otra
        // excepción genérica para controlar el resto de errores posibles

        ServerSocket socServidor = new ServerSocket(8066); // Definimos el puerto permitido para recibir peticiones
        imprimeDisponible(); // Imprimimos las diferentes opciones permitidas por pantalla/console

        while(true) { // Creamos un bucle infinito mediante el cual nos notificará cada petición que vaya procesando
            Socket socCliente = socServidor.accept();
            System.out.println("Atendiendo al cliente ");
            procesaPetición(socCliente);
            socCliente.close();
            System.out.println("cliente atendido");
        }
    }

    /* Dentro de la presente clase añadiremos los métodos "imprimeDisponible()" y "procesaPetición()", los cuales se
    detallarán y explicarán a continuación */

}

```


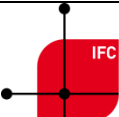
*/*Comentario: He ido introduciendo comentarios entre las diferentes líneas de código para así tratar de explicarlo*/*

/* Definimos el método procesaPetición() con las diferentes vistas y cabeceras a mostrar según la petición introducida */

```
private static void procesaPetición(Socket socketCliente) throws IOException { // Creamos el método para
// procesar y controlar las diferentes peticiones que reciba nuestro servidor HTTP
// Definimos el método de entrada
InputStreamReader inSR = new InputStreamReader(socketCliente.getInputStream());
// Inicializamos el 'buffer' o memoria temporal donde se irán guardando los diferentes datos introducidos
// a través de la consola o terminal
BufferedReader bufLeer = new BufferedReader(inSR);
// Imprime el texto predefinido
PrintWriter printWriter = new PrintWriter(socketCliente.getOutputStream(), true);
// Leemos la petición introducida a través del teclado
String petición = bufLeer.readLine();
// Eliminamos los posibles espacios que pueda contener
petición = petición.replaceAll(" ", "");
if (petición.startsWith("GET")) { // Creamos un condicional IF para averiguar de qué clase de
// petición se trata y asegurarse de que resulta accesible y está permitida
// Extraemos la petición introducida, resultando la última parte de la dirección URL completa
petición = petición.substring(3, petición.lastIndexOf("HTTP"));
String html;
if (petición.length() != 0 && !petición.equals("/")) {
// Página de Inscripción
if (petición.equals("/inscripcion")) {
// Construimos la estructura HTML a mostrar
html = "<html><head><title>Inscripción</title></head><body><h1>" +
"Inscripción – En proceso de construcción</h1></body></html>";
// Cabeceras definidas
printWriter.println("HTTP/1.1 200 OK"); // Línea de estado
printWriter.println("Content-Type:text/html;charset=UTF-8"); // Tipo de contenido
printWriter.println("Content-Length: " + html.length() + "1"); // Longitud del contenido
printWriter.println("\n");
// Imprimimos la página HTML por pantalla
printWriter.println(html);
} else { // Página de error cuando no se puede procesar la petición introducida
// Construimos la estructura HTML a mostrar
html = "<html><head><title>noEncontrado</title></head><body>"
+ "<h1>¡ERROR! Página no encontrada</h1>"
+ "<p>Mensaje de error: La página que solicitaste no existe en nuestro servidor</p></body></html>";
// Cabeceras definidas
printWriter.println("HTTP/1.1 404 Not Found"); // Línea de estado
printWriter.println("Content-Type:text/html;charset=UTF-8"); // Tipo de contenido
printWriter.println("Content-Length: " + html.length() + "1"); // Longitud del contenido
printWriter.println("\n");
// Imprimimos la página HTML por pantalla
printWriter.println(html);
}
} else {
// Página principal
html = "<html><head><title>Inicio</title></head><body><h1>¡Bienvenido!</h1>"
+ "<p>Bienvenido a mi web</p></body></html>";
// Cabeceras definidas
printWriter.println("HTTP/1.1 200 OK"); // Línea de estado
printWriter.println("Content-Type:text/html;charset=UTF-8"); // Tipo de contenido
printWriter.println("Content-Length: " + html.length() + "1"); // Longitud del contenido
printWriter.println("\n");
// Imprimimos la página HTML por pantalla
printWriter.println(html);
}
}
}
```

/* Creamos el método imprimeDisponible() para mostrar al/la usuario/a las diferentes rutas permitidas y disponibles. Este resulta un método auxiliar, el cual se mostrará por consola/terminal de comandos tras poner en funcionamiento nuestro servidor HTML */

```
private static void imprimeDisponible() {
    System.out.println("El Servidor WEB se está ejecutando y permanece a la escucha por el puerto 8066.\n"
    + "Escribe en la barra de direcciones de tu explorador preferido:\n\n"
    + "http://localhost:8066\n para solicitar la página de bienvenida\n\n"
    + "http://localhost:8066/inscripcion\n para solicitar una página de Inscripción,\n\n"
    + "http://localhost:8066/q\n para simular un error");
}
```

 cpifp Bajo Aragón	Programación de Procesos y Servicios	 IFC INFORMÁTICA Y COMUNICACIONES
	2ª Evaluación - 2025	
Fecha: 15/01/2024	Alumno:	Nota:

