

<b>(66.62) REDES DE COMPUTADORAS</b>		<b>1999</b>	
<b>PRÁCTICA :</b>	<b>5</b>	<b>TEMA :</b>	<b>Berkeley Sockets</b>

**Responda las preguntas y justifique cada respuesta.**

1. ¿Qué es la interface Berkeley Sockets?. ¿Para que sirve y en qué capa del modelo ISO/OSI se ubica?.
2. Escriba la sintaxis y explique el funcionamiento de las 8 principales llamadas de la interfaz socket.
3. Explique cuál es el orden típico de utilización de las llamadas en un cliente y en un servidor.
4. ¿En qué estructuras de datos se almacenan las direcciones IP y los números de ports TCP/UDP?. ¿Qué es un internet end-point address y en qué estructura de datos se almacena?.
5. ¿Detalle la estructura conceptual de datos de un socket?. ¿Qué campos determina el programa y cuáles el Sistema Operativo?. Ayuda: Diferenciar entre cliente y servidor, y recordar el concepto de “well known ports”.
6. Escriba las líneas de código en lenguaje C para convertir y almacenar un nombre de servicio (e.g. FTP, SMTP) o su número correspondiente (e.g. 21, 25) al campo correspondiente en un internet endpoint address. Ayuda: Revise el código del módulo connectsock() en el Cap. 7 del libro de Comer (Vol. III).
7. ¿Qué tipo de servidores existen en cuanto a concurrencia y si están o no orientados a la conexión?.
8. ¿Cómo se utiliza la llamada fork() para implementar un servidor concurrente?. Escriba el código mínimo para implementar un servidor concurrente con fork().
9. Escriba el código mínimo para implementar un servidor concurrente iterativo.
10. ¿Cómo se resuelve con la llamada select(), el problema de “deadlock” que puede ocurrir si un server ejecuta las llamadas read() en el servidor y el cliente no escribe, o si un server iterativo ejecuta la llamada accept() y no se producen pedidos nuevos de conexión?.
11. Implemente un cliente TCP de echo.
12. Implemente un servidor concurrente de HTTP 1.0.