	Ingeniería en Informática Facultad de Ciencias Fisicomatemáticas e Ingeniería Universidad Católica Argentina		
Cátedra	Protocolos de Internet - Examen Parcial 2020	**Prof. Titular** **Prof. Asistente**	Ing. Javier Ouret Ing. Ignacio Parravicini

Parcial a desarrollar en forma remota: criterios generales y requisitos para su aprobación.

- El tema del parcial estará basado en conceptos y técnicas ya desarrollados durante las clases teóricas y prácticas.
- Cada alumno debe desarrollar el parcial en forma individual.
- Dado que es un parcial a libro abierto no está permitido el trabajo en grupos ni la interconsulta con otros profesores o alumnos.
- El parcial consistirá de:
    - Primera parte: desarrollo de programas cliente-servidor de acuerdo a la consigna que se entregará el 7/6 a partir de las 8.00 horas.
    - Segunda parte: cuestionario de opciones múltiples (multiple-choice) que será habilitado el 8/6 a las 20.30 en el EVA.
    - 80% de la nota del parcial corresponde a la primera parte y 20% a la segunda parte.
- La primera parte del parcial deberá ser entregada antes de las 20.30 hs del 8/6.
- Formato de archivo fuente en .c ó .py. Instrucciones o notas complementarias en .txt.
- Luego de ese día y hora no se admitirán más entregas, y el parcial se considerará reprobado.
- La primera parte del parcial deberá subirse a la plataforma del EVA de acuerdo a las instrucciones que se indiquen al momento de comenzar el parcial.
- En caso de problemas con la plataforma EVA con el acceso a internet, el parcial deberá enviarse por correo electrónico a javierouret@uca.edu.ar.
- El código de cada programa deberá ser original, no copiado de sitios de internet, libros, etc.
- Deberá estar escrito en español, a excepción de las funciones propias del lenguaje, palabras protegidas, etc.
- Deberá estar íntegramente documentado dentro del mismo código explicando cada paso y por qué se usa cada función, estructura, bucle, etc.
- El lenguaje a utilizar debe ser C o Python.
- Deberá incluirse el instructivo para la compilación y ejecución de cada programa desarrollado.
- Cada programa deberá funcionar correctamente y respetando la consigna propuesta.
- IMPORTANTE: es imposible que dos personas programen exactamente de la misma manera, que estructuren el código en forma similar, que documenten de la misma forma o que usen los mismos nombres de funciones o variables. Por lo tanto cualquier similitud entre dos desarrollos implicará que el parcial no se apruebe y posibles sanciones de acuerdo al reglamento de la universidad. Como se trata de un parcial en modalidad virtual se solicita trabajar con honestidad.

- **DESCRIPCION DEL PROGRAMA A DESARROLLAR**

Implementar un programa servidor y un programa cliente, para que atienda solicitudes de clientes y que realice lo siguiente:

- **Protocolo para iniciar la conexión.**
- **Mensajes entre clientes a través del servidor.**
- **Log de mensajes guardados en el servidor.**
- **Opción para ver el log de mensajes desde el cliente indicado la opción msj-log.**
- **Cálculo del tiempo de ida y vuelta al final de cada mensaje utilizando un timestamp.**
- **Protocolo para cerrar la conexión en forma ordenada con cada cliente utilizando shutdown.**

El cliente y el servidor se comunican intercambiando líneas de caracteres ASCII por medio de TCP. La interacción entre el cliente y el servidor para este caso se ejecuta de la siguiente manera:

1. El servidor "escucha" las conexiones en algún puerto (superior a 1024). Puede usar el puerto 15001.

2. El cliente abre una conexión al socket del servidor.

3. El servidor acepta la conexión y espera recibir una solicitud del cliente.

4. Una vez que el cliente está conectado al servidor, inmediatamente envía una cadena (string) de solicitud. El formato de la cadena de solicitud es:

```
ParcialTL2020 <EB> <tipo de solicitud> <EB> <nombre de usuario>
<EB> <extremo> <TIV> <fin de línea>
```

dónde:

- <EB> es "espacio en blanco", uno o más caracteres en blanco o tabulados.
- <tipo de solicitud> es una cadena.
    - msj: para iniciar y continuar una conversación con mensajes.
    - log: para recuperar el msj-log con todas las conversaciones con ese cliente.
    - fin: para cerrar el msj.

- <nombre de usuario> es el nombre-registro del alumno. Debe contener sólo caracteres ASCII visualizables y no puede tener más de 64 caracteres en longitud.
- <extremo> tiene la forma <dirección IP del cliente> - <número de puerto del cliente> (El <extremo> en este caso se refiere al socket del cliente).
- <TIVms> Tiempo estimado para el ida y vuelta del mensaje en milisegundos.
- <fin de línea> es el marcador de fin de línea, es decir, la secuencia de dos caracteres "\ r \ n".

5. Por lo tanto, si el socket del cliente está vinculado al puerto 12981 en 128.163.1.163, la cadena de solicitud se verá como:

`ParcialTL2020 msj juan40-23456 128.163.1.163-12981 45ms \ r \ n`

6. Tener en cuenta que la longitud máxima de la cadena de solicitud, excluyendo espacios en blanco y la terminación \ r \ n secuencia, tiene 96 caracteres.

7. Al recibir y analizar la cadena de solicitud, el servidor responde enviando un saludo terminado por "\ r \ n". Si la cadena de solicitud se formó correctamente, la línea se ve así:

Hola `juan40-23456` del 128.163.1.163-12981. Bienvenido al servidor de mensajes del `ParcialTL2020` \ r \ n

8. Si la solicitud no se ha formado correctamente o no contiene el identificador de extremo final adecuado, la línea contendrá un mensaje de error.

9. Después de recibir la línea de saludo, el CLIENTE realizará un bucle o loop hasta la lectura de fin de archivo (end-of-file) desde stdin y la conexión de red. Todo lo que ve en stdin (teclado), lo copia al servidor; todo lo que ve en la conexión de red desde el servidor, lo escribe en stdout (pantalla).

10. Después de enviar la línea de saludo, el SERVIDOR realizará un bucle o loop hasta la lectura de fin de archivo desde la conexión de red. Todo lo que ve en la conexión de red del cliente, escribe de vuelta a la conexión de red (operación de eco). Cuando el cliente cierra la conexión, el servidor espera una nueva conexión.

11. Para el msj-log grabar los mensajes de cada cliente desde el servidor en un archivo llamado `<nombre de usuario>.txt.` Cuando el servidor recibe como mensaje la palabra “log” sola recupera el archivo de msj-log y le envía el contenido al cliente.

12. Para cerrar la conexión con el cliente (implementada con shutdown()) cuando el servidor recibe como mensaje la palabra “fin” sola, cierra la conexión con el siguiente aviso:

`Conexión con juan40-23456 del 128.163.1.163-12981 finalizada.`

13. Para el cálculo del TIV usar la función time() para marcar el tiempo de la solicitud desde el cliente y para la respuesta desde el servidor. La diferencia dará el tiempo de viaje.