Material permitido: **Calculadora NO programable.** Tiempo: **2 horas.** N	**Aviso 1:** Todas las respuestas deben estar razonadas. **Aviso 2:** Escriba sus respuestas con una letra **lo más clara posible.** **Aviso 3: No use *Tipp-ex*** o similares (atasca el escáner).

ESTE EXAMEN CONSTA DE 5 PREGUNTAS

1. (*1.5 p*). Responda a las siguientes cuestiones:

a) (*0.75 p*). Explique el resultado de ejecutar la orden:

```
$ cat < /etc/host.conf
```

b) (*0.75 p*). Explique el funcionamiento de la siguiente llamada al sistema y el significado de sus parámetros.

```
resultado=nice(incremento);
```

2. (*1.5 p*). Explique razonadamente si las siguientes afirmaciones son verdaderas o falsas:

a) (*0.75 p*). Un proceso hijo puede acceder a una tubería sin nombre creada por su proceso padre.

b) (*0.75 p*). El núcleo puede acceder directamente a los campos del *área U* del proceso que se está ejecutando pero no al *área U* de otros procesos.

3. (*2 p*). Conteste a las siguientes cuestiones:

a) (*0.5 p*). Defina qué son las hebras de usuario.

b) (*1.5 p*). Describa sus ventajas e inconvenientes.

4. (*2 p*). Escriba un programa en C que cree una zona de memoria compartida privada de tamaño 1024 bytes, donde solo el usuario va a tener permisos de lectura y escritura.

Pregunta 5 en la página siguiente

5. (*3 p*) Conteste razonadamente a los siguientes apartados:

a) (*1.5 p*) Explicar el significado de las sentencias enumeradas (**[ ]**) del siguiente programa.

b) (*1.5 p*) El programa es compilado enlazando la librería `lpthread` y creándose el ejecutable `Examen`. Describir el **funcionamiento** así como la **salida** obtenida salida cuando se invoca la orden "`$ ./Examen`". Supóngase que el sistema asigna PIDs a los procesos consecutivamente comenzando con pid=1000 y que no se produce ningún error.

```
        #include <pthread.h>
        #include <stdlib.h>
        #include <stdio.h>
        #include <string.h>

        char cadena1[]="Problema 5", cadena2[]="incorrecto.";

[1]     void *fun1(void *arg)
        {
[2]     wait();
        printf("PidB=%d\n", getpid());
[3]     strcpy(cadena2,"correcto.");
        }

        void main(void)
        {
        int pid;
[4]     if ((pid=fork())==-1){perror("Fallo en fork:"); exit(1);}
        if (pid==0)
             {
[5]          sleep(1);
             strcpy(cadena1,"Ejercicio imposible");
             printf("PidA=%d\n", getpid());
             }
        else
             {
             pthread_t hilo;
[6]          if (pthread_create(&hilo,NULL,fun1,NULL))
                  {
                  printf("Error creando pthread");
                  exit(1);
                  }
[7]          pthread_join(hilo,NULL);
[8]          printf("PidC=%d\n", getpid());
             printf("%s del examen DyASO %s\n",cadena1,cadena2);
             }
        }
```