This exam is **closed book**. You can have **one page** of notes. **UH expels cheaters.**

1. **Short questions.** Answer in a single sentence to each of the following questions: (6×5 points)

   a) What is the main advantage of **modular kernels**?

   They are extensible: new features can be added to the kernel without having to recompile it.

   b) What is the function of the UNIX **signal()** system call?

   The signal system call specifies what a process should do when it receives a signal from another process: this is called "catching" the signal.

   c) What is one of the main purposes of **timer interrupts**?

   They are used to prevent CPU-bound processes from monopolizing the kernel by assigning to each running process a time slice of CPU time after which the process is interrupted.

   d) What is the main advantage of **preemptive schedulers** over non-preemptive ones?

   They are used to prevent CPU-bound processes from monopolizing the kernel by assigning to each running process a time slice of CPU time after which the process is interrupted.

   e) What is the main disadvantage of the **master/slave organization** for multiprocessor operating systems?

   It presents a potential bottleneck as all OS requests must be handled by the master CPU.

   f) Which feature of UNIX made it **more portable** than previous operating systems?

   It was written in a high-level language instead of assembly language.
2. Which of the following statements apply to (a) kernel-supported threads, (b) user level threads and (c) all threads? (5 points per correct line)

<table>
<thead>
<tr>
<th>Statement</th>
<th>Kernel-supported</th>
<th>User-level</th>
<th>Both types</th>
</tr>
</thead>
<tbody>
<tr>
<td>They allow users to write threaded applications that can run on different operating systems.</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>They share the address space of their parent.</td>
<td></td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>They often require the use of non-blocking system calls.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>They let the kernel allocate several processors to threads sharing an address space.</td>
<td>X</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

3. Which values of \( i \) will be printed by the parent process and the child process of this program? (2×5 points)

```c
main() {
    int i = 0;
    fork();
    i++;
    printf("i = %d\n", i);
} // main
```

The parent process will print \( i = \) ___1____ and the child process will print \( i = \) ___1____

4. What is the outcome of the following code sequence? (5 points)

```c
int fd;
fd = open(thisfile, O_RDWR);
close(0);
dup(fd);
```

_____stdin__________ will be redirected to __the file "thisfile"_________________
5. Which processes are good candidates for being swapped out?(5 points)

The processes that **have been a long time in the waiting state**

---

*Explaining why.* (2×5 points).

a) Why do most operating systems on the market continue to use *monolithic kernels*?

Because they are faster than microkernels

b) Why should we *prevent* users of a multi-user system from *rebooting* the OS from their own CD-ROM?

Because they might boot up an unsafe kernel that would let them access, modify or delete the files and the processes of other users.

6. Consider the following System V Release 4 scheduler where the question marks represent new priority levels: (2×10 points)

<table>
<thead>
<tr>
<th>ts_quantum</th>
<th>ts_tqexp</th>
<th>ts_slpret</th>
<th>ts_maxwait</th>
<th>ts_lwait</th>
<th>LEVEL</th>
</tr>
</thead>
<tbody>
<tr>
<td>800</td>
<td>?</td>
<td>?</td>
<td>16000</td>
<td>?</td>
<td># 0</td>
</tr>
<tr>
<td>400</td>
<td>?</td>
<td>?</td>
<td>8000</td>
<td>?</td>
<td># 1</td>
</tr>
<tr>
<td>200</td>
<td>?</td>
<td>?</td>
<td>4000</td>
<td>?</td>
<td># 2</td>
</tr>
<tr>
<td>100</td>
<td>?</td>
<td>?</td>
<td>2000</td>
<td>?</td>
<td># 3</td>
</tr>
</tbody>
</table>

a) What is a *good value* for the *ts_tqexp* parameter at *priority level 0* and *why*?

The ts_tqexp parameter at priority level 0 should be equal to 0 because we do not want to increase the priority of a process that exceed its CPU time slice

b) What is a *good value* for the *ts_slpret* parameter at *priority level 2* and *why*?

The ts_slpret parameter at priority level 2 should be equal to 3 because we should increase the priority of processes that return to the ready state from the wait state as it indicates they have just completed a system call.