This exam is closed book. You can have one page of notes.
Please answer every part of every question.

1. Match each of the following features with the single sentence that describes it best: (10×3 points) (Hint: Several of the choices offered are plain wrong.)

<table>
<thead>
<tr>
<th>Feature</th>
<th>Single Sentence</th>
</tr>
</thead>
<tbody>
<tr>
<td>timer interrupts</td>
<td>i) Allow their users to submit several jobs to be processed in sequence</td>
</tr>
<tr>
<td>UNIX</td>
<td>k) Always implemented in hardware.</td>
</tr>
<tr>
<td>layered kernel organization</td>
<td>m) Implemented as a user-level process on Windows NT.</td>
</tr>
<tr>
<td>swapping</td>
<td>f) Much slower than microkernels.</td>
</tr>
<tr>
<td>memory protection</td>
<td>b) Occasions at least two context switches.</td>
</tr>
<tr>
<td>dual-mode CPU</td>
<td>j) Provides a good way to make space in main memory.</td>
</tr>
<tr>
<td>non-volatile RAM</td>
<td>n) Sends a signal to another process.</td>
</tr>
<tr>
<td>batch systems</td>
<td>a) Still the fastest kernel organization.</td>
</tr>
<tr>
<td>system call</td>
<td>e) Used to prevent the running process from monopolizing the CPU.</td>
</tr>
<tr>
<td>monolithic kernels</td>
<td>h) Used to prevent unauthorized access to users' files.</td>
</tr>
<tr>
<td></td>
<td>i) Would be a good idea if it was working.</td>
</tr>
<tr>
<td></td>
<td>k) Would allow a much more intensive usage of delayed disk writes.</td>
</tr>
</tbody>
</table>

2. What are the respective advantages of microkernels and monolithic kernels? (2×5 points)

Microkernels are smaller, more manageable and easier to debug than monolithic kernels. Unfortunately, they are much slower as system calls involving a server process require four context switches instead of two.
3. For each of the statements below, indicate in one sentence whether the statement is true or false (2 points), and why (3 points).

a) A process waiting on a disk I/O is in the suspended state.
   False, it is in the waiting state.

b) It makes no sense to have memory protection on a single user system.
   False, it prevents user processes from interfering with each other or damaging the kernel.

c) Processes never interrupt themselves.
   False, processes interrupt themselves every time they do a system call.

d) `execve()` system calls are often followed by a `fork()` system call.
   False, the `fork()` normally precedes the `execve()`.

e) In a multiprogramming system, there can be many programs in the system but there is never more than a single process.
   False, multiprogramming means having several processes in the system at the same time.

f) A UNIX process shares the address space of its parent.
   False, each process has its own private address space.

4. What is the default action that a Unix process takes when it receives a signal from another process? (5 points) What can it do to prevent that from happening? (5 points)

   By default, A UNIX process terminates when it receives a signal. To prevent that process can catch it by using the `signal()` system call. The ninth signal, SIGKIL, cannot be caught.

5. How many lines of output will the following program print out? (5 points)?

```c
main(){
    fork(); fork(); printf("Hi!\n");
}
```

   The program will print __4________ lines.

6. What is happening to stdin, stdout, and stderr after the following lines are executed. (3×5 points)

```c
int fd;
fd = open("data.txt", O_RDWR | O_CREAT, 0640);
close (0);
close (1);
dup(fd);
```

   stdin is now __ reading from "data.txt" ____________________________

   stdout is now __ closed ____________________________

   stderr is now __ unchanged ____________________________