1. For legacy systems, explain why we would need a VBR if we already have an MBR.

.....

.....

.....

Score

/ 3

2. What is the difference between a hypervisor and a typical virtual machine monitor.

.....

.....

.....

Score

/ 3

3. Write one use that you can think of the `/proc` filesystem?

.....

.....

.....

Score

/ 3

4. What is *preemption* and why is it necessary to properly support multitasking?

.....

.....

.....

.....

Score

/ 1

5. If the modular architecture and micro-kernel approaches both split the operating system in smaller modules, what is the benefit of the micro-kernel architecture over the modular structure?

.....

.....

.....

.....

.....

Score

/ 1

6. Is it possible for a context switch to occur without a mode switch? Support your answer with a logical explanation.

.....

.....

.....

.....

Score

/ 2

7. List the three *types* of traps that can interrupt the normal execution of a program?

.....

.....

.....

Score

/ 2

8. Take a look at the following code and write the maximum number of processes that will will be present at any time.

```
void main(int argc, char **argv) {
    int pid;
    for (int i = 0; i < 4; i++) {
        pid = fork();
        if(pid != 0) {
            sleep(2000);
            break;
        }
    }
    wait(NULL)
}
```

Score / 2

.....

9. What is the need of the *created* state in the process life cycle? Mention at least one specific task that is performed in this state.

.....

.....

.....

.....

.....

Score / 2

10. When we say that the kernel must 'save the state before switching context', what do we mean?

.....

.....

Score / 1

11. Explain the operations carried out by the kernel when the fork system call is issued by a process. Provide as much detail as you can in the allocated space.

.....

.....

.....

.....

Score / 2

12. What is the *most important* limitation of using user-space threads. How is it remedied by modern operating systems.

.....

.....

.....

.....

Score

/ 4

13. Give an example from the programs you've coded where a race condition can lead to data inconsistency. (The example must *not* be of cash withdrawl and deposit.) Explain fully, the sequence of operations, which can lead to the data inconsistency.

.....

.....

.....

.....

.....

.....

.....

.....

Score

/ 3

14. In the example given above, suggest changes to ensure that the race condition is avoided.

.....

.....

Score

/ 1

15. If you initialize a semaphore with the value `5` and issue the `down` call on it from `7` different threads ($t_1, \ldots, t_7$), *list* the state of each thread after the calls. (States can be 'ready', 'running', 'blocked', 'zombie').

.....

.....

.....

.....

Score

/ 3

16. Explain with the help of a figure how the *init* process creates all the other processes in the linux system. Describe the two critical system calls required for this procedure.

.....

.....

.....

.....

.....

Score

/ 2

17. Assume there's a thread-aware operating system. Thread t_1 from within process p_1 makes an IO request. Consider the time when the operating system has executed the corresponding syscall handler, but I/O has not yet been completed. Can the process p_1 still be selected by the scheduler to run at this time? Give a reason for your answer.

.....

.....

.....

.....

Score

/ 2

18. *Bonus question:* What is the name of the structure in the Linux kernel source that corresponds to the PCB?

.....

Score

/ 1