PART A

- **Each question has TWO correct answers. Circle the correct answers.**
- **2 marks and 1 mark will be awarded for the first and second correct answers respectively. However NO mark will be awarded if one or more of the answers to the question is wrong, or the question is not answered.**
- **To get the best possible mark you should only circle answers you are sure are correct. Never circle an answer if you are not sure because it might be an incorrect answer. Do not attempt to guess answers as you might lose all marks for the question.**
- **Total marks for this part is 18**

Q1: Which of the following statements are true of Process Control Blocks (PCBs)?

A: A PCB is needed as it stores state information used by the operating system to manage the process

B: A PCB is not needed as state information is not used by the *operating* system to manage the process

C: A PCB is created when a process is created. It does not need to be updated after the process has been created.

D: A PCB is created when a process is created. It must be kept updated during the lifetime of the process.

E: A typical PCB must contain a process identifier (PID). However, the PCB would never contain the program counter value or information specifying memory use.

Q2: Which of the following statements are true?

A: A process is in a 'blocked' state if it is waiting for the completion of another service, such as an I/O operation or a system call that cannot return immediately.

B: A process is in a 'runnable' state if it is waiting for the completion of another service, such as an I/O operation or a system call that cannot be completed immediately.

C: A process is in a 'blocked' state if it has been prevented from accessing protected or invalid memory.

D: A process in a 'blocked' state is always terminated by the operating system since it cannot continue to run.

E: A process in a 'blocked' state may be suspended (swapped out of memory) by the operating system to free up memory for a ready process

Q3: Which of the following statements are true about switching between processes? During a process switch...

A: ...only the program counter register must be saved.

B: ...the process control block must be updated.

C: ...the process control block must be saved to secondary storage.

D: ...the process control block must be unblocked via the DMA module.

E: ...the process control block (or a pointer to it) must be placed in an appropriate queue, such as ready, blocked, suspended, etc.

Q4: The boot sector...

A: Is stored in a non-volatile part of main memory.

B: Is the first addressable block on the hard drive.

C: Contains instructions to locate and begin loading the operating system from disk.

D: Holds the system BIOS.

E: Holds a copy of the file allocation table for the file system.

Q5: A disk access request requires track 16 and sector 140 on a hard disk surface. The head is currently over track 24 and sector 210. What two things need to happen for the head to be in the correct location?

A: The hard drive must stop spinning and spin backwards to the required track.

B: The disk arm must move to align the head with the required track.

C: The drive needs time to rotate to the correct track.

D: The drive needs time to rotate to the correct sector.

E: The disk arm must align the head with the required sector.

Q6: An operating system schedules processes round-robin. Which of the following describes the operation of this scheduling algorithm?

A: Processes are scheduled in order of track locations on the hard drive.

B: Each running process will be interrupted at the end of its allocated timeslice.

C: A process must be blocked for I/O or a system call before another process can enter the 'running' state.

D: When a process which is not waiting for I/O leaves the running state at the end of its timeslice, it enters the Suspended state.

E: When a process which is not waiting for I/O leaves the running state at the end of its timeslice, it is usually pushed to the back of the Ready queue.

PART B

- **Answer all the questions in the space provided below the question**
- **The marks for each question as indicated at the end of the question**
- **The maximum number of marks for this part is 12.**
- **Rough work can be completed at the back of the paper, and will not be marked.**

Q1: The following section of C code uses the fork() system call. (Assume that this code is part of a larger program, which compiles with no errors and no warnings.)

The purpose of the code is to have exactly 10 child processes created by the original process. The parent process for each child should be the originally-running process. Any child processes should not fork further processes.

```
pid_t fork_ret;
int counter = 0;

do {
    fork_ret = fork();
    counter++;
} while(fork_ret == 0 && counter < 10);

printf("Count: %d\n", counter);
exit(0);
```

(a) Does the code meet the purpose described? Why or why not? Explain your reasoning. **(4 marks)**

No. The child process generated will create the same instructions as the parent ~~parent~~ process. So the child processes will fork further process causing it to have more than exactly 10 child processes by the original process. Each time the child process runs, it continues the next instruction of the parent process and fork further (in the loop).

2

(b) Write down the first 4 lines of a possible output from this code. **(4 marks)**

Count: 10
Count: 10
Count: 10
Count: 10

1

Q2: Requests are made for data on a hard drive, located at the following tracks. (The track numbers are listed here in the order that the requests were made, oldest first.)

63, 76, 163, 180, 32, 84, 190, 33.

Assume that the disk head is currently positioned over track 40 and is currently moving in the direction of decreasing track number.

In the space below, write the order in which the track requests should be served if **C-SCAN** disk scheduling is used. **(4 marks)**

33, 32, 190, 180, 163, 84, 76, 63.

4

END OF TEST