1a)i) Shortest job first

ii)10-200ms

iii) When running times are similar

iv) When each job takes less than or equal to one second to run

b) The instructions are:

a=i;

b=i;

**If there are n threads and m instructions in each thread,**

**Number of interleavings = (mn)!/(m!)^n**

**Or if we have two pieces of code with a and b instructions (can be extended) then:**

**(a+b)!/(a!b!)**

**Or more generally for threads with instructions:**

So the number of interleavings with 4 threads is (4\*2)!/(2!)^4 = 8!/16 = 2520 total interleavings.

One way to think about this kind of problem is to consider it as a combination problem. Specifically, in our case, we have 4 threads and 2 instructions per thread, that means we have 4\*2 = 8 instructions in total to be executed. The first thread will need to choose 2 out of these 8 instructions slots to execute its 2 instructions which means 8C2, similarly for the second thread, it needs to choose 2 out of the 6 remaining slots(2 slot have already been chosen by thread 1) to execute its 2 instruction which gives us 6C2. And we repeat the process and obtain 8C2 \* 6C2 \* 4C2 \* 2C2 = 2520.

And this approach can generalized, which leads to the above formula. Proof:

ii) In thread 1:

down(s);

down(s);

down(s);

a=i;

b=i;

In all 3 other threads:

a=i;

b=i;

up(s);

c) Not examinable that is what they want you to believe

FOR Q2 SEE 2020 LIVE SESSION [HERE](https://imperial.cloud.panopto.eu/Panopto/Pages/Viewer.aspx?id=b2a2312b-f69e-4f28-ae72-ac8000ca5fa7) AT AROUND 35:00

2.a) A file system fs is mounted if there is a link in a directory of another file system to the root of fs.

b)

/\* device \*/ dev\_t st\_dev;

/\* inode \*/ ino\_t st\_ino;

/\* protection \*/ mode\_t st\_mode;

/\* number of hard links \*/ 0

/\* user ID of owner \*/ root

/\* group ID of owner \*/ gid\_t st\_gid;

/\* device type (if inode device) \*/ dev\_t st\_rdev;

/\* total size, in bytes \*/ off\_t st\_size;

/\* blocksize for filesystem I/O \*/ unsigned long st\_blksize;

/\* number of blocks allocated \*/ unsigned long st\_blocks;

/\* time of last access \*/ time\_t st\_atime;

/\* time of last modification \*/ time\_t st\_mtime;

/\* time of last change \*/ time\_t st\_ctime;

c) OS opens inode table and creates inode entry in memory (with disk device number, inode number, number of processes with opened file, minor/major device number).

Three points the dude who taught us wanted:

1. The filesystem will be traversed from the root directory to find the file.
2. The permissions will be checked on the file to see if the user has sufficient privileges to read the file.
3. The file inode is brought into main memory and a file descriptor is created by the kernel when making the open system call. The file descriptor is then returned to the user process.

d) i)No, a hard link cannot be made across different file systems since it directly reference the address of the file.

ii) Yes, a soft link is simply an alias to a different path, so it can point to a different file system.

e) Translation table to tell, which disk does the hard link refer to. This would need a hashmap from memory address to disk.