Section A

Question 1.

1. CPU clock frequency is faster than main memory access time. Caches can be used to hold subset of the main memory, sufficient to to cover the program’s working set. Lowering the effective memory access time closer to the CPU clock time.
2. Temporal locality: means that memory locations accessed recently are likely to be accessed again.   
   Temporal locality is related to cache size, associativity and replacement policy, because those governs how long a cache line is likely to stay in the cache before being replaced.  
     
   Spatial locality: memory location located close to each other are likely to be accessed as well. It is dependent on cache line/block size, which determines the amount of data close to a requested memory address will be present in the cache line.
3. Least recently used (LRU): Expensive to implement. Replace the item that hasn’t been used the longest.   
   Round robin: cycle around locations, least recently fetched from memory. Replace the item that has been first fetched from memory.  
   Random: easy to implement. Choose position in random, not really bad.
4. No idea.
5. Something about how to have 100% hit rate is to have the same size of memory in cache, and bigger size means more complex, expensive and probably slower.   
     
   We can’t simply increase the size because we will always reach a size that will miss.
6. For 100 access:  
     
   L1 cache: 100 \* 98% \* 1 = 98 cycle   
   L2 cache: 2 \* 90% \* (4 + 1) = 9 cycles  
   L3 cache: 0.2 \* 70% \* (4 + 1 + 10) = 2.1 cycles  
   Memory: 0.06 \* (100 + 4 + 1 + 10) = 6.9 cycles  
     
   Total cycles = 98+ 9 + 2.1 + 6.9 = 116 cycles  
   On average that is 116/100 = 1.16 cycle in average

Question 2.

1. If there are instructions that aren’t dependent on each other we can run these instructions at the same time without side effects. E.G.  
   ADD R0, R1, R2  
   SUB R3, R4, R5   
   We can execute both ADD and SUB same time without any problem
3. Because they might appear in an order that blocks other instructions from being executed due to dependency.
4. By reordering instruction and putting instructions that don’t depend on previous instruction calculation in between instructions that do, will reduce the number of stalls in the program leading to a better utilization of processor resources.
5. ADD R6, R4, R5  
   ADD R3, R2, R1  
   SUB R4, R4, R5  
   SUB R2, R2, R1  
   MUL R4, R4, R6  
   MUL R3, R2, R3  
   MUL R3, R3, R4 (There will be a stall here)

Section B.

Question 3

1. Virtualization can “translate” between technologies (different instruction sets, system calls), change the level of abstraction (garbage collector, debuggers…) and multiplex the system resources to look different.
2. Something something
3. What
4. WAWT

Question 4.

1. Seek time: usually in the nanoseconds, really small value, the time it takes for the head of read/write to reach the position over the correct cylinder.   
   Rotation speed: determines the time it takes on average for a given disk sector to arrive under the head. Usually is 5400 RPM, 7200RPM for consumers.   
   Transfer rate: determines the number of data per unit of time that can be accessed sequentially once the head is in position.
2. Screw this
3. Performance and reliability.  
   By using RAID0 (file striping) we can strip files into multiple disks which increase transfer rate.  
   Reliability can be enhanced by mirroring the files (RAID1)   
   Or by storing parity data that allow reconstructing of files in case of disk failure (RAID2-6)
4. Probably about capacity, and probably that disks are unreliable and slow, so we need to use RAID0 and RAID1 and RAID2-6 in some sort of combination like RAID010