Michael Laucella
HW1

1. MISD: This type of architecture is unusual because all of the instruction streams perform the same instruction on the same pieces of data, this is used for sensitive applications such as government or air traffic control systems which need to certain of the values calculated. This is not useful for user level systems as most users don't need to run such critical tasks, this type of architecture uses a lot of computation power on a focused task and wouldn't be used to run trivial programs.

2. 5 types of parallelism
    a. ILP [HARDWARE]: simultaneous execution of instructions, executes multiple instances of the same instruction

    b. Pipelining [HARDWARE]: splitting instructions into an assembly line fashion and running different stages of the pipeline overlapped

    c. Hyperthreading [HARDWARE]: a core acts as 2 virtual cores which can share resources

    d. Task level [SOFTWARE]: distributed memory - running multiple process each with its own private memory that interact through message passing

    e. Thread level [SOFTWARE]: shared memory - running a single process with multiple threads that can access both shared and private memory

3. Calculated using the formula 1/(F+((1-F)/P)), with the given parameters F=30% or 3/10 while 1-F=70% or 7/10 which gives us 1/(.3+(.7)/P) and then plug in P
    a. T(1) : 1
    b. T(2) : 1.538
    c. T(4) : 2.105
    d. T(8) : 2.581
    e. T(16) : 2.909
    f. T(INF) : 3.333

4. Analysis of DAG
    a. work=20
    b. T(1)=20
    c. T(5)=T(1)/5=20/5=4
    d. T(6)=T(1)/6=20/6=3.333
    e. Critical path=9

5. Concurrency v Parallelism

    a. Concurrency 1(core): the tasks would be switching in and out running for a period of time based on OS scheduling
    b. Concurrency 2(core): each core should run the process thereby making it more parallel
    c. Parallelism 1(core): would run concurrently by switching between the threads through OS scheduling
    d. Parallelism 2(core): each core should run a thread

6. Cache: the purposes of caching is to improve memory access time by keeping smaller memory units called caches near the cores, they allow for faster memory access when the information has been stored in the cache. The idea is that an element that has been used is likely to be used again (temporal locality) and that elements near it might also be accessed (spatial locality). Its purpose does not change between multi and single core processors but it’s implementation does. Cache coherence is needed to make sure values in each core's cache are consistent, this synchronizing of data between the caches and main memory doesn’t need to happen if there's only a single core.