1.13)

a) When it comes to mainframe computers cpu time and network usage need to be utilized extremely efficiently since it needs to manage an extreme amount of tasks at once that generally result in some sort of network transfer.

b) Workstations connected to servers really only need to worry about managing its own cpu usage, and its memory.

c) Mobile computers like laptops and phones need to also manage its cpu time and memory accessing to help reduce the strain on their batteries.

1.19)

Traps aren’t within the kernel and can be raised by user code. They are helpful for catching errors in your program like trying to access memory that you shouldn’t be. Interrupts can only be called by the kernel and let the system know it has a new task to do.

1.23)

If each of the processors each access the same address in main memory and perform different actions on that data then they would all have different values in their cache. Like lets say the all get int x from memory which is equal to 1 initially. Lets say that processor 1 multiplies it by 1, processor 2 multiplies it by 2, and processor 3 multiplies it by 3 then they would and have different values stored in their cache.

1.25)

You’d basically just keep a record of the allocated memory for each process, if a process tries to access memory that wasn’t allocated to that specific process you could throw an exception.

2.13)

According to the book there are three different ways that parameters are passed from the program to the OS. The first and easiest way is to store the parameters in the registers, but this method only works when there are less parameters than there are registers. The second method is similar except it stores the parameters as a block of memory and the memory address is then passed to the registers. The final method involves the program pushing the parameter onto the stack and then the operating system will pop the parameter off of the stack.

2.14)

To keep track of time you would need to count how many cpu cycles a specific section took to convert that to real time you would need to divide that by the processors cycles per second. In the end you could just add up all the different times you got for each section and then divide by the time taken by a specific section to see what percentage of the cpu usage was by a certain section. This type of analysis is important because it allows you to compare different algorithms and find the best one for your situation.