1)

(a) Processor is a small chip which resides inside the computer whose main function is to take an input and provide an appropriate output. The words CPU and processor are used interchangeably. In general processor or CPU corresponds to virtual CPU or LCPU. There'll be only one physical CPU present. So words Processor, CPU, virtual CPU, LCPU and threads are used interchangeably. Name of my processor is Intel(R) Core(TM) i7-8550U. It has x86 processor architecture and its a 64 bit processor. A core is a small CPU or processor built into a big CPU. According to my machine, it has 2 threads per core i.e, 2 processors or LCPUs per one core.

(b) My machine has 4 cores per socket and 1 socket. So there are 4 cores in my machine. Command used to find this is lscpu.

(c) My machine has 8 processors or CPUs. Command used to find this is lscpu.

(d) Frequency of each processor is 800MHz. It can have a maximum value of 4000MHz and minimum value of 400MHz. Command used to find this is lscpu.

(e) My machine has 8058708 kB of physical memory i.e., ~8GB. Command used to find this is grep MemTotal /proc/meminfo

(f) My machine has 3585152 kB of free memory i.e., ~3.5GB. Command used to find this is free.

(g) 11738 forks were done since the boot of this system. Command used to find this is vmstat -f.

(h) Number of Context Switches made sice bootup is 18743696. Command used to find this is cat /proc/stat | grep ctxt.

2)

(a) The PID of process running the cpu command is 15865.

(b) It consumes 100% CPU and 0% memory.

(c) The current state is R i.e., running. This can be found in S column of top command.

3)

(a) The PID of process running the cpu-print is 17532. Command used to find this is ps -A | grep cpu-print.

(b) The PID of parent process is 15843. Command used to find this is ps -f 17532.
The PID of second generation is 15834. Command used to find this is ps -f 15843.
The PID of third generation is 3017. Command used to find this is ps -f 15834.
The PID of fourth generation is 1. Command used to find this is ps -f 3017.
As the PID is 1 it's the first init process and doesnot have any parent.

(c) File descriptor 0 points to dev/pts/1 which corresponds to input. File descriptor 1 points to /tmp/tmp.txt which coresponds to output. File descriptor 2 points to /dev/pts/1 which corresponds to error output. Command used to find this is ls -l /proc/6492/fd where 6492 is pid of the process. So the command takes input from the terminal i.e., dev/pts/1 and outputs in the tmp/tmp.txt file. So the I/O direction will be from dev/pts/1 to tmp/tmp.txt.

(d) File descriptor 0 points to pipe:[83551] which corresponds to input. File descriptor 1 points to /dev/pts/1 which coresponds to output. File descriptor 2 points to /dev/pts/1 which corresponds to error output. Command used to find this is ls -l /proc/7354/fd where 7354 is pid of the process. So the command takes input from the terminal using pipe and outputs that again in terminal. So the I/O direction will be from terminal to terminal via pipe.

(e) history and cd already exist as in-built executables where as ls and ps have executables in /bin. Command used to find this is type <x>. Where <x> means history or cd or ls or ps.

4)

Vitrual Memory used by memory1.c is 8296KB where as virtual memory used by memory2.c is 8292KB. The physical Memory used by memory1.c is 756KB where as physical memory used by memory2.c is 3204KB. This is because of lazy allocation of the OS i.e., OS allocates the memory when it is actually needed. In the first scenario the array is just initialized and not used so it is not allocated with memory while in the second case the array is actually called so it got allocated in the RAM so it has more physical memory. Command used to find this is ps -eo pid,tid,class,rtprio,stat,vsz,rss,comm.

5)

Disk is used by the first file for every loop but disk is used only once by the second file. In the first scenario the file calls random file every time so it needs to read the file every time from the disk. But in the second case the file calls the same file 0 everytime so it reads from the disk only once and stores in the cache of the RAM for faster access. So it neednot read everytime from the disk. Command used to find this iostat -d 1. Command used to clear cache is sudo sh -c 'echo 3 >/proc/sys/vm/drop_caches'. For the both cases I measured the KB read from the disk every second. In the first case for the first time it was around ~4.9GB and it is almost the same when we run the command solely without running the files. From next time it was showing ~30MB everytime indicating that everytime its reading a file from the disk. In the second case for the first time it had shown same value i.e., around ~4.9GB and from next time onward it was showing ~0 indicating that it’s reading from the cache and not from the disk.