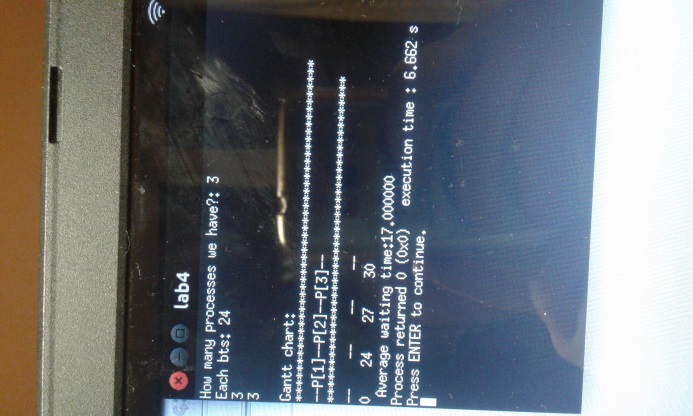
At all, scheduling algorithms are used to decide which process in queue is to be allocated to CPU.

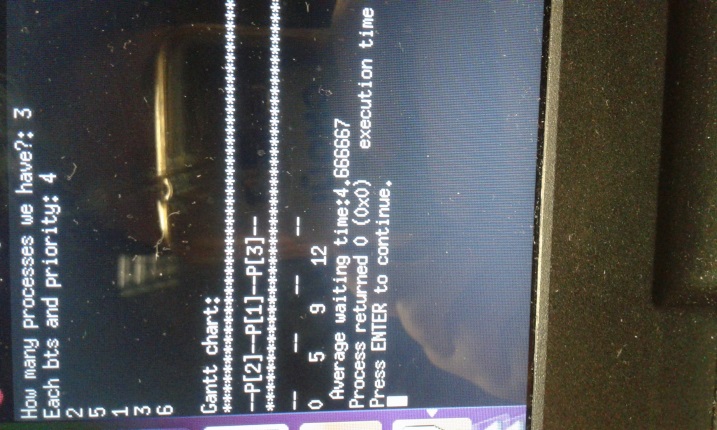
1. First algorithm is FCFS, which works similar to principle of LIFO. As CPU will become free it is allocated to the process which is in the head of queue. In the program we have array of burst time and waiting time of each process. Waiting time for each process is calculated within loop by adding burst time of process executed before it. In order to calculate average waiting time we just take sum of waiting times and divide by number of processes. Also we have turnaround time which defined by sum of burst and waiting time of each process. And to illustrate this schedule we use Gantt chart, which includes the start and finish times of each of the participating processes. Result is shown below:



2.Shortest job first algorithm, as its name says executes first those processes which burst time is the smallest. So here in the program we simply write some sorting algorithm in order to sort our burst times in ascending order. In my case I used selection sorting. Then similar to fcfs illustrated it in Gantt chart. This case is without arrival time. So it can be considered as non-preemptive, which will allow the currently running process to finish its CPU burst. Preemptive case is called as shortest remaining time first algorithm. Result as follows:



3. Also we have priority scheduling, which is general case of SJF . In this algorithm the CPU is allocated to the process with the highest priority. Its implementation similar to previous one, but here we also sort priorities and indexes of each process to be executed. So we have array of priorities and array of indexes of each process. Sorting implemented using selection sorting, which repeatedly find the minimum element from unsorted part and puts it to the beginning.



4. In the case of time sharing systems we will need Round Robin algorithm, as it switches between processes under the given time quantum, which shows that it is preemptive. The CPU scheduler sets a timer for first process from queue to interrupt after 1 time quantum, and dispatches the process. Here we check whether CPU burst is less than 1 time quantum. If so, process itself will release the CPU voluntarily. Otherwise, timer will cause an interrupt to the OS and our process will go the tail of queue. Implementation of program also takes into consideration arrival time of each process by creating array ar[i]. Result as follows:

