**Student name:** Anchal Kumari

**Registration no.:**  11808628

**Email address:** anchalkumari9122@gmail.com

**GitHub link:**

**Description:**

* **Shortest job first (SJF)** is a scheduling algorithm in which the process having the smallest execution time is chosen for the next execution.
* This algorithm can be pre-emptive or non-pre-emptive.
* It can improve process throughput by making sure that shorter jobs are executed first, hence possibly has a short turnaround time.
* In non-pre-emptive scheduling, once a CPU cycle is allocated to the process, the process holds it till it reaches a waiting state or terminated. It also leads to starvation if a process with long burst time is running after that the process with less burst time will come.
* But in pre-emptive one jobs are put in the ready queue as they came and process with short burst time begins but the current running process is pre-empted if any new job came with short burst time.
* SJF is frequently used for long term scheduling. E.g.: Batch system.
* Complexity: O (nlogn).

**Algorithm:**

* First sort all the processes in the increasing order of arrival time. By using swapping process.
* Then find the :
* Completion time: time at which process completes its execution.
* Turnaround time: time difference between completion time and arrival time. I.e. turnaround time=completion time- arrival time.
* Waiting time: time difference between turnaround time and burst time.

**Code:**

waiting\_time[0] = 0;

for(i=1;i<n;i++)

{

sum1 += burst\_time[i-1];

waiting\_time[i] = sum1 - arrival\_time[i];

final\_wait += waiting\_time[i];

}

for(i=0;i<n;i++)

{

sum2 += burst\_time[i];

turnaround\_time[i] = sum2 - arrival\_time[i];

final\_turnaround += turnaround\_time[i];

}

* After this finally we prioritize our processes in shortest job next scheduling algorithm means shortest job first non-pre-emptive algorithm by applying :

**Formula:** 1+waiting time/completion time

**Code:**

completion\_time[0] = burst\_time[0];

for(i=1;i<n;i++)

{

completion\_time[i] = completion\_time[i-1] + burst\_time[i];

}

for(i=0;i<n;i++)

{

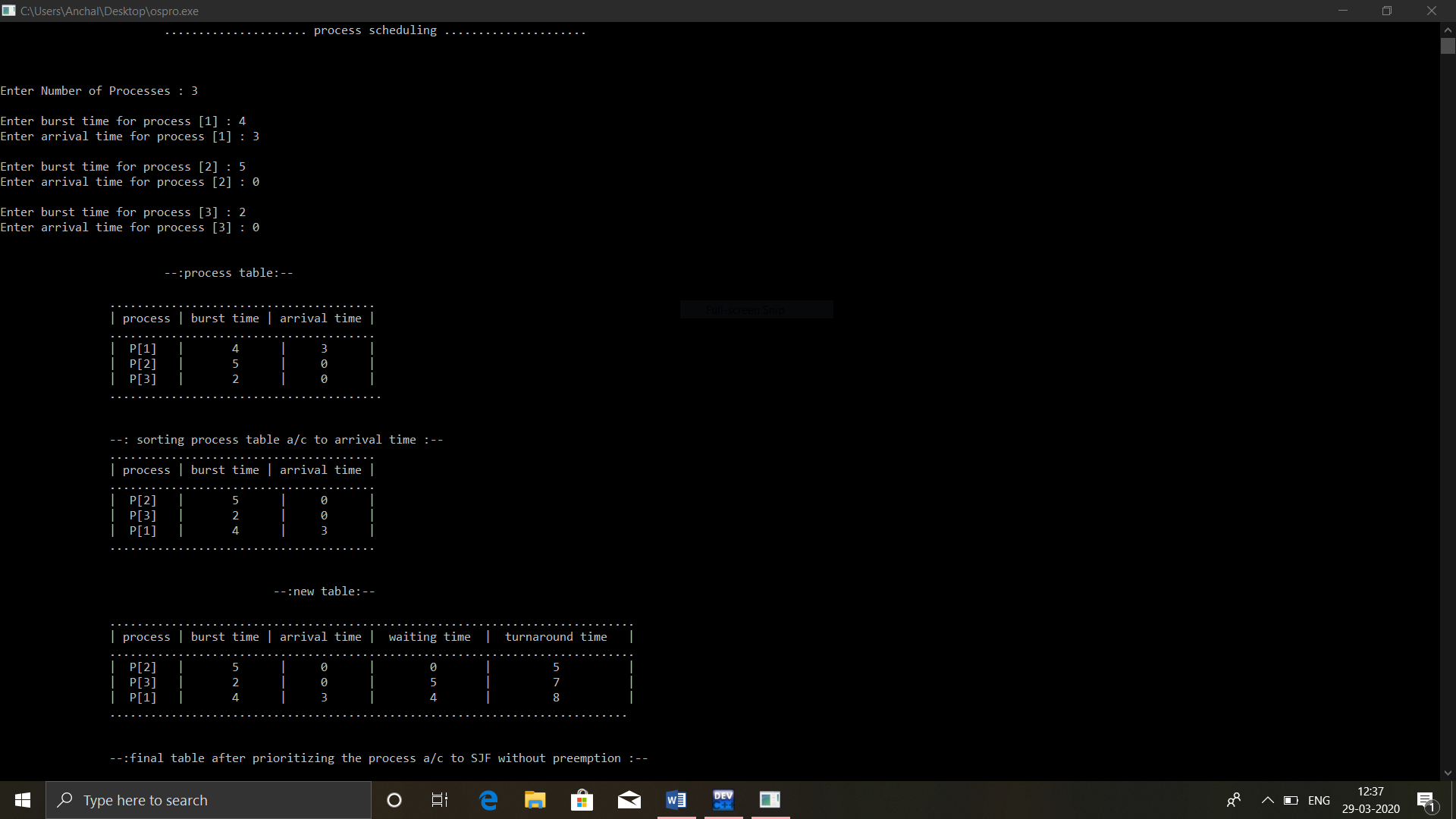
priority[i] = 1+waiting\_time[i]/completion\_time[i];

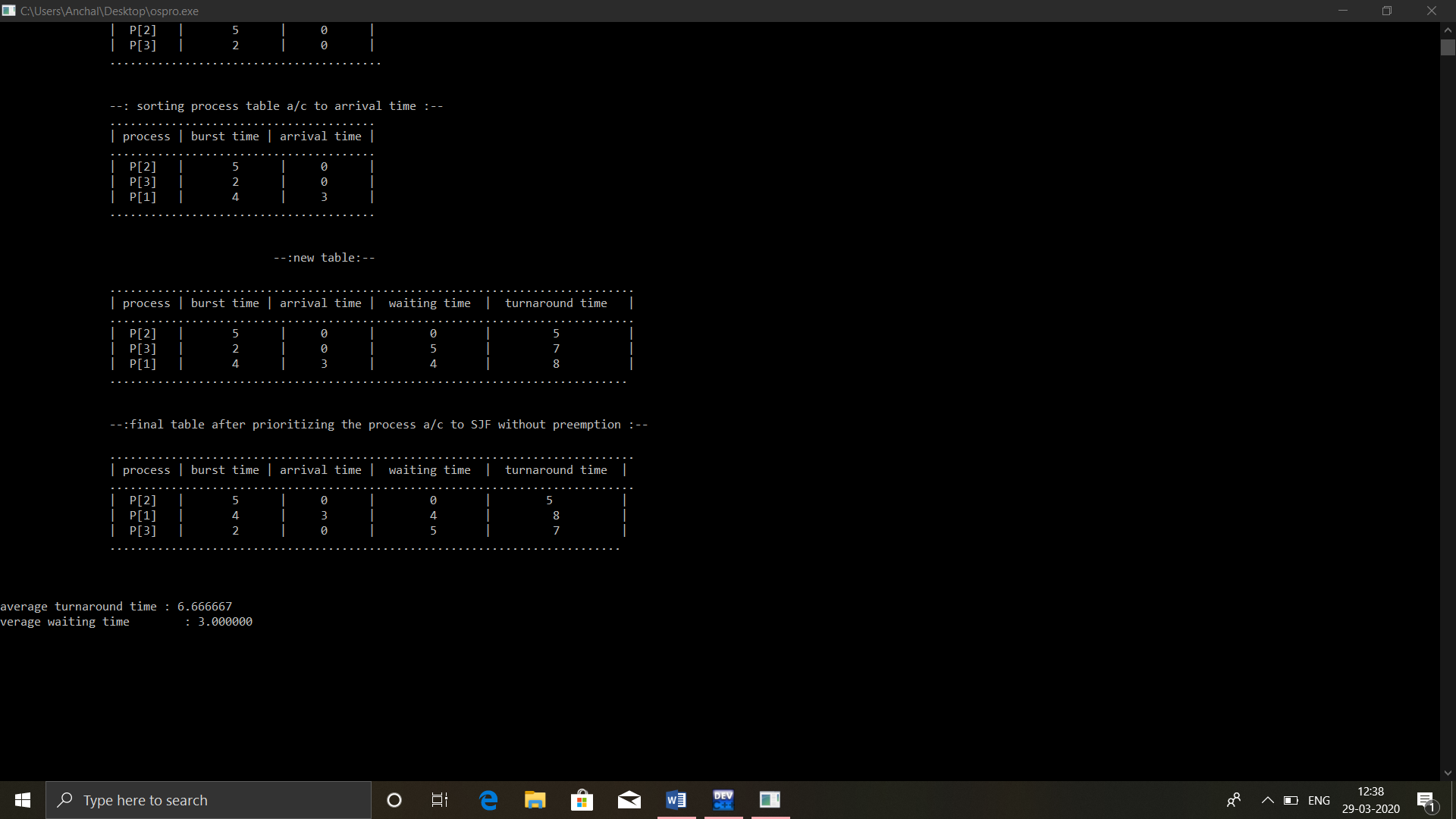
}

* After this we will also find average waiting time and average turnaround time for the scheduling processes.

**Boundary conditions:**

* Burst time of program cannot be zero or less than zero, otherwise it will throw an error because the process do not able to run.
* The minimum number of processes required to schedule them is two. It do not work in less than two processes. Because we need at least two processes for comparison.

**Output:** 



**Thank you…**