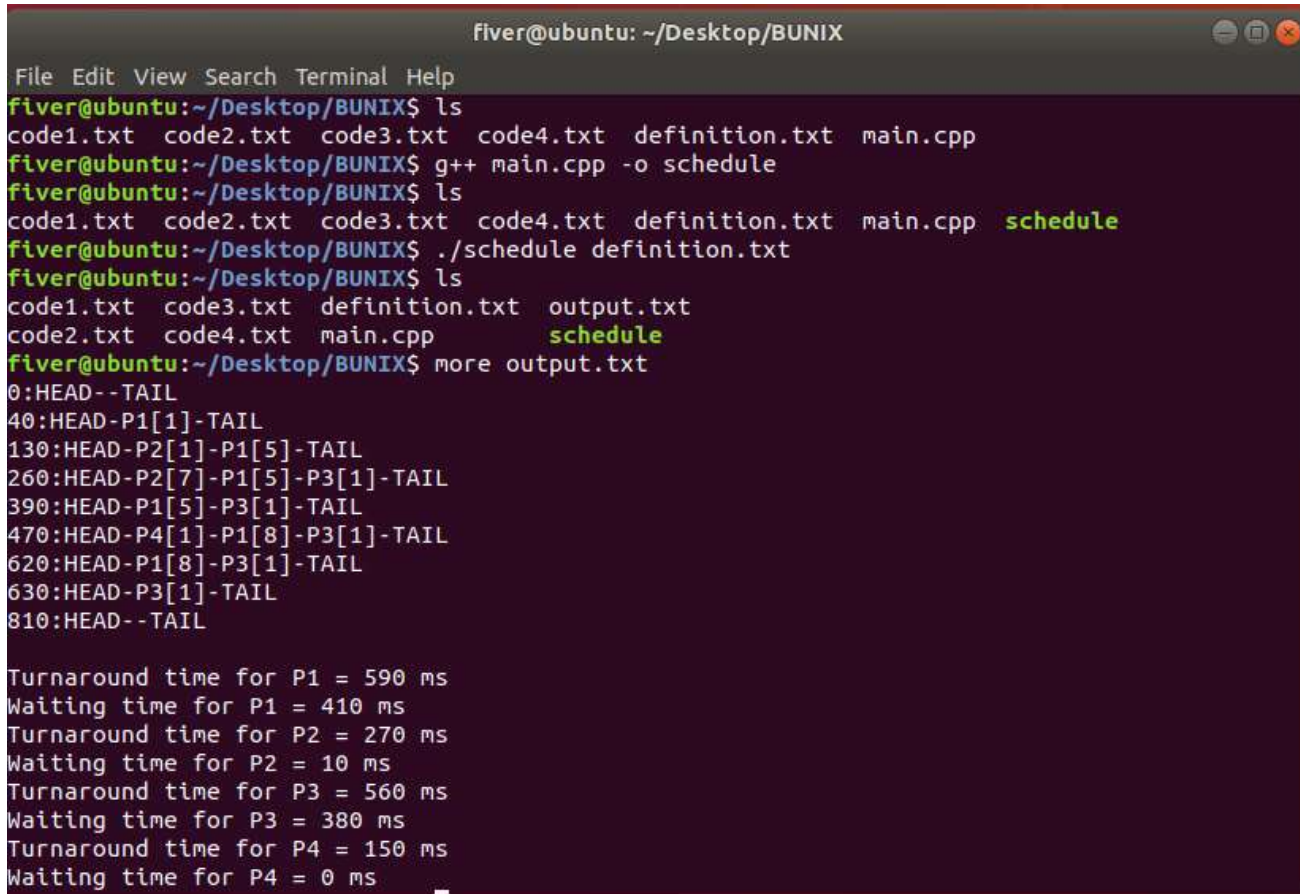


## CPU SCHEDULER FOR BUNIX

The screenshot below shows how to compile and run the cpu scheduler.



```
fiver@ubuntu: ~/Desktop/BUNIX
File Edit View Search Terminal Help
fiver@ubuntu:~/Desktop/BUNIX$ ls
code1.txt code2.txt code3.txt code4.txt definition.txt main.cpp
fiver@ubuntu:~/Desktop/BUNIX$ g++ main.cpp -o schedule
fiver@ubuntu:~/Desktop/BUNIX$ ls
code1.txt code2.txt code3.txt code4.txt definition.txt main.cpp schedule
fiver@ubuntu:~/Desktop/BUNIX$ ./schedule definition.txt
fiver@ubuntu:~/Desktop/BUNIX$ ls
code1.txt code3.txt definition.txt output.txt
code2.txt code4.txt main.cpp schedule
fiver@ubuntu:~/Desktop/BUNIX$ more output.txt
0:HEAD--TAIL
40:HEAD-P1[1]-TAIL
130:HEAD-P2[1]-P1[5]-TAIL
260:HEAD-P2[7]-P1[5]-P3[1]-TAIL
390:HEAD-P1[5]-P3[1]-TAIL
470:HEAD-P4[1]-P1[8]-P3[1]-TAIL
620:HEAD-P1[8]-P3[1]-TAIL
630:HEAD-P3[1]-TAIL
810:HEAD--TAIL

Turnaround time for P1 = 590 ms
Waiting time for P1 = 410 ms
Turnaround time for P2 = 270 ms
Waiting time for P2 = 10 ms
Turnaround time for P3 = 560 ms
Waiting time for P3 = 380 ms
Turnaround time for P4 = 150 ms
Waiting time for P4 = 0 ms
```

### DEVELOPMENT ENVIRONMENT:

**Operating System:** Windows 10 Home Single Language 64 Bit - Version 1803

**Integrated Development Environment:** Codeblocks-17.12 with MinGW

**\*Tested on Ubuntu 18.04.1 LTS. G++ is used for compilation.**

How does it work?

- Check for arrival process or processes
- If there is any, add them to the ready queue
- If ready queue is not empty, execute the first process's instruction
- Increment instruction counter of the process, update the system time.
- Check whether CPU executed the "exit" instruction of the process
- If then, remove the process from the ready queue
- If there are processes to arrive and ready queue is empty, update the system time.
- If there is no arrival and ready is empty, no more things to do.