



Discipline

Activity

CS5003

Activity 1

Sheikh Muhammed Tadeeb (AU19B1014)

❖ Problem Statement:

Do the parsing on the file and generate inference.

❖ Solution:

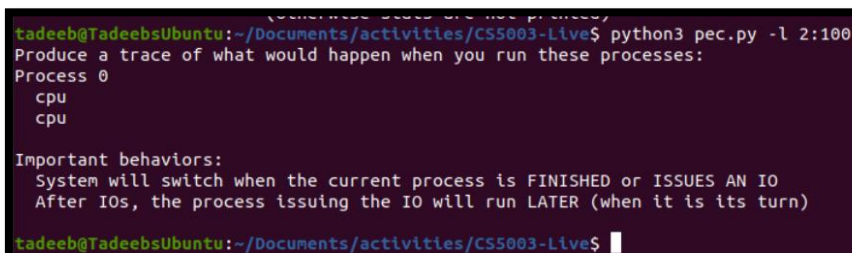
- Process List:

It is a comma-separated list of processes to run, in the form of X1:Y1, X2:Y2 where X is the number of instructions that process should run and Y are the chances (from 0 to 100 %) that an instruction will use the CPU or issue an IO.

– 1 represents the process list, which specifies exactly what each running program (or process) will do. A process consists of instructions, and each instruction can just do one of two things:

1. Use the CPU
2. Issue an IO (and wait for it to complete)

Below, 2 instructions with 100% chances is given. So, the allocation is done for 2 CPU.



```
tadeeb@TadeebUbuntu:~/Documents/activities/CS5003-Live$ python3 pec.py -l 2:100
Produce a trace of what would happen when you run these processes:
Process 0
  cpu
  cpu

Important behaviors:
  System will switch when the current process is FINISHED or ISSUES AN IO
  After IOs, the process issuing the IO will run LATER (when it is its turn)

tadeeb@TadeebUbuntu:~/Documents/activities/CS5003-Live$
```

In this, 6 instructions are given, where 50% chances are there for the 3 different process states, which are cpu, io (input output) and io_done (input output_done).

That's why each process is running 3 times (as 50% of 6 is 3).

- c abbreviation computes answers for the instructions, displaying the process ID (PID), which is initialized after the instruction is given.
- With the PID, based on the instructions and chances of instructions CPU will take, the numbers will be given accordingly to the column CPU and IOs.
- For 6 instructions, with 100% chances, 5 CPUs are added on the PID=0, with null for IOs.

```
tadeeb@TadeebsUbuntu:~/Documents/activities/CS5003-Live$ python3 pec.py -l 6:100 -c
Time      PID: 0      CPU      I/Os
1         RUN:cpu      1
2         RUN:cpu      1
3         RUN:cpu      1
4         RUN:cpu      1
5         RUN:cpu      1
6         RUN:cpu      1
tadeeb@TadeebsUbuntu:~/Documents/activities/CS5003-Live$
```

- With 4 instructions given and 0% chances, the program issues 4 I/Os. When each I/O is issued, the process moves to a SUSPEND state, and while the device is busy servicing the I/O, the CPU is idle.

```
tadeeb@TadeebsUbuntu:~/Documents/activities/CS5003-Live$ python3 pec.py -l 4:0 -c
Time      PID: 0      CPU      I/Os
1         RUN:to      1
2         SUSPEND      1
3         SUSPEND      1
4         SUSPEND      1
5         SUSPEND      1
6         SUSPEND      1
7*        RUN:to_done  1
8         RUN:to      1
9         SUSPEND      1
10        SUSPEND      1
11        SUSPEND      1
12        SUSPEND      1
13        SUSPEND      1
14*       RUN:to_done  1
15        RUN:to      1
16        SUSPEND      1
17        SUSPEND      1
18        SUSPEND      1
19        SUSPEND      1
20        SUSPEND      1
21*       RUN:to_done  1
22        RUN:to      1
23        SUSPEND      1
24        SUSPEND      1
25        SUSPEND      1
26        SUSPEND      1
27        SUSPEND      1
28*       RUN:to_done  1
tadeeb@TadeebsUbuntu:~/Documents/activities/CS5003-Live$
```

- `-p` print statistics at end, which is only useful with `-c` flag (otherwise stats are not printed).
- In the below program, the statistics are not seen as the option `-p` is not used with the option `-c`.

```
tadeeb@TadeebsUbuntu:~/Documents/activities/CS5003-Live$ python3 pec.py -l 4:0 -p
Produce a trace of what would happen when you run these processes:
Process 0
to
to_done
to
to_done
to
to_done
to
to
to_done

Important behaviors:
System will switch when the current process is FINISHED or ISSUES AN IO
After I/Os, the process issuing the IO will run LATER (when it is its turn)
tadeeb@TadeebsUbuntu:~/Documents/activities/CS5003-Live$
```

- After using the option `-c` with `-p`, the stats are shown, including the total time, which is 28 clock ticks to run, but the CPU was only busy for approximately 29% of the time. The IO device, on the other hand, was quite busy i.e., 71.43%.

```
tadeeb@TadeebUbuntu:~/Documents/activities/CS5003-Live$ python3 pec.py -l 4:0 -c -p
Time      PID: 0      CPU      IOs
1         RUN:to      1
2         SUSPEND
3         SUSPEND
4         SUSPEND
5         SUSPEND
6         SUSPEND
7*        RUN:to_done  1
8         RUN:to      1
9         SUSPEND
10        SUSPEND
11        SUSPEND
12        SUSPEND
13        SUSPEND
14*        RUN:to_done  1
15        RUN:to      1
16        SUSPEND
17        SUSPEND
18        SUSPEND
19        SUSPEND
20        SUSPEND
21*        RUN:to_done  1
22        RUN:to      1
23        SUSPEND
24        SUSPEND
25        SUSPEND
26        SUSPEND
27        SUSPEND
28*        RUN:to_done  1

Stats: Total Time 28
Stats: CPU Busy 8 (28.57%)
Stats: IO Busy 20 (71.43%)
tadeeb@TadeebUbuntu:~/Documents/activities/CS5003-Live$
```

- In this case, two different processes run, each just using the CPU

```
tadeeb@TadeebUbuntu:~/Documents/activities/CS5003-Live$ python3 pec.py -l 4:100,5:100
Produce a trace of what would happen when you run these processes:
Process 0
cpu
cpu
cpu
cpu

Process 1
cpu
cpu
cpu
cpu
cpu

Important behaviors:
System will switch when the current process is FINISHED or ISSUES AN IO
After IOs, the process issuing the IO will run LATER (when it is its turn)
tadeeb@TadeebUbuntu:~/Documents/activities/CS5003-Live$
```

- Below it is seen that the first process with process ID or PID 0 runs, while process 1 is in READY state to run but just waits until 0 is done. When 0 is finished, it moves to the DONE state, while 1 run. When 1 finish, the trace is done.

```
tadeeb@TadeebUbuntu:~/Documents/activities/CS5003-Live$ python3 pec.py -l 4:100,5:100 -c
Time      PID: 0      PID: 1      CPU      IOs
1         RUN:cpu      READY      1
2         RUN:cpu      READY      1
3         RUN:cpu      READY      1
4         RUN:cpu      READY      1
5         DONE        RUN:cpu      1
6         DONE        RUN:cpu      1
7         DONE        RUN:cpu      1
8         DONE        RUN:cpu      1
9         DONE        RUN:cpu      1
tadeeb@TadeebUbuntu:~/Documents/activities/CS5003-Live$
```

- The below snippet has the first process with 4 instructions and 100% chances and the second with 1 instruction and 0% chances. So, first PID 0 is in the running state and PID 1 is in Ready state waiting for PID 0 to be DONE. As the PID 0 is DONE PID 1 is ready to Run with 1 IOs and is done.

```
tadeeb@TadeebUbuntu:~/Documents/activities/CS5003-Live$ python3 pec.py -l 5:100,1:0 -c
Time  PID: 0      PID: 1      CPU      IOs
1     RUN:cpu    READY      1
2     RUN:cpu    READY      1
3     RUN:cpu    READY      1
4     RUN:cpu    READY      1
5     RUN:cpu    READY      1
6     DONE      RUN:io     1
7     DONE      SUSPEND    1
8     DONE      SUSPEND    1
9     DONE      SUSPEND    1
10    DONE      SUSPEND    1
11    DONE      SUSPEND    1
12*   DONE      RUN:io_done 1
tadeeb@TadeebUbuntu:~/Documents/activities/CS5003-Live$
```

- -S: PROCESS_SWITCH_BEHAVIOR is when it switches between processes. There are two methods:

1. SWITCH_ON_IO
2. SWITCH_ON_END

- First, switching between the IO, so there are two processes given, one with 1 instruction and 0% chances and the other with 5 instructions and 100% chances.
- Then switching on the END, with the same two processes given, one with 1 instruction and 0% chances and the other with 5 instructions and 100% chances.

```
tadeeb@TadeebUbuntu:~/Documents/activities/CS5003-Live$ python3 pec.py -l 1:0,5:100 -c -S SWITCH_ON_IO
Time  PID: 0      PID: 1      CPU      IOs
1     RUN:io    READY      1
2     SUSPEND  RUN:cpu    1
3     SUSPEND  RUN:cpu    1
4     SUSPEND  RUN:cpu    1
5     SUSPEND  RUN:cpu    1
6     SUSPEND  RUN:cpu    1
7*    RUN:io_done  DONE      1
tadeeb@TadeebUbuntu:~/Documents/activities/CS5003-Live$ python3 pec.py -l 1:0,5:100 -c -S SWITCH_ON_END
Time  PID: 0      PID: 1      CPU      IOs
1     RUN:io    READY      1
2     SUSPEND  READY      1
3     SUSPEND  READY      1
4     SUSPEND  READY      1
5     SUSPEND  READY      1
6     SUSPEND  READY      1
7*    RUN:io_done  READY      1
8     DONE      RUN:cpu    1
9     DONE      RUN:cpu    1
10    DONE      RUN:cpu    1
11    DONE      RUN:cpu    1
12    DONE      RUN:cpu    1
tadeeb@TadeebUbuntu:~/Documents/activities/CS5003-Live$
```

- From the above snippet we can see that in the switch IO mode, as soon as the IO issued to PID 0, the PID 1 is in READY state and then PID 0 gets suspended, making the PID 1 in the RUN state and then finally DONE.

- Whereas, in the switch END mode, the CPU process will not use the CPU when the IO process is in SUSPEND state. CPU will switch to CPU process when the IO process is done.
- When two processes just use the CPU, there will be no such effect by the switch modes.

```
tadeeb@TadeebUbuntu:~/Documents/activities/C55003-Live$ python3 pec.py -l 5:100,4:100 -c -S SWITCH_ON_END
Time  PID: 0      PID: 1      CPU      I/Os
1     RUN:cpu    READY      1
2     RUN:cpu    READY      1
3     RUN:cpu    READY      1
4     RUN:cpu    READY      1
5     RUN:cpu    READY      1
6     DONE      RUN:cpu    1
7     DONE      RUN:cpu    1
8     DONE      RUN:cpu    1
9     DONE      RUN:cpu    1
tadeeb@TadeebUbuntu:~/Documents/activities/C55003-Live$ python3 pec.py -l 5:100,4:100 -c -S SWITCH_ON_IO
Time  PID: 0      PID: 1      CPU      I/Os
1     RUN:cpu    READY      1
2     RUN:cpu    READY      1
3     RUN:cpu    READY      1
4     RUN:cpu    READY      1
5     RUN:cpu    READY      1
6     DONE      RUN:cpu    1
7     DONE      RUN:cpu    1
8     DONE      RUN:cpu    1
9     DONE      RUN:cpu    1
tadeeb@TadeebUbuntu:~/Documents/activities/C55003-Live$
```

- I: *IO_DONE_BEHAVIOR* is a type of behavior when IO ends using the *IO_RUN_LATER* or *IO_RUN_IMMEDIATE* mode.
- In the below case, with two processes run, each just using the CPU, there is no difference as such.

```
tadeeb@TadeebUbuntu:~/Documents/activities/C55003-Live$ python3 pec.py -l 3:100,2:100 -c -I IO_RUN_LATER
Time  PID: 0      PID: 1      CPU      I/Os
1     RUN:cpu    READY      1
2     RUN:cpu    READY      1
3     RUN:cpu    READY      1
4     DONE      RUN:cpu    1
5     DONE      RUN:cpu    1
tadeeb@TadeebUbuntu:~/Documents/activities/C55003-Live$ python3 pec.py -l 3:100,2:100 -c -I IO_RUN_IMMEDIATE
Time  PID: 0      PID: 1      CPU      I/Os
1     RUN:cpu    READY      1
2     RUN:cpu    READY      1
3     RUN:cpu    READY      1
4     DONE      RUN:cpu    1
5     DONE      RUN:cpu    1
tadeeb@TadeebUbuntu:~/Documents/activities/C55003-Live$
```

- With *IO_RUN_LATER*, when an I/O completes, the process that issued it does not necessarily run right away, rather, whatever was running at the time keeps running.
- *IO_RUN_LATER* means that the IO process has to wait until all the CPU processes have done. During the execution, there are two part which need to promote, one is process 0 is having a Ready State when process 2 and process 3 are Running, the other is the last three of IO waiting when the CPU is idle.

```
tadeeb@TadeebUbuntu:~/Documents/activities/CS5003-Live$ python3 pec.py -l 1:0,4:100 -c -I IO_RUN_LATER
Time PID: 0 PID: 1 CPU Ios
1 RUN:io READY 1
2 SUSPEND RUN:cpu 1 1
3 SUSPEND RUN:cpu 1 1
4 SUSPEND RUN:cpu 1 1
5 SUSPEND RUN:cpu 1 1
6 SUSPEND DONE 1
7* RUN:io_done DONE 1
tadeeb@TadeebUbuntu:~/Documents/activities/CS5003-Live$ python3 pec.py -l 1:0,4:100 -c -I IO_RUN_IMMEDIATE
Time PID: 0 PID: 1 CPU Ios
1 RUN:io READY 1
2 SUSPEND RUN:cpu 1 1
3 SUSPEND RUN:cpu 1 1
4 SUSPEND RUN:cpu 1 1
5 SUSPEND RUN:cpu 1 1
6 SUSPEND DONE 1
7* RUN:io_done DONE 1
tadeeb@TadeebUbuntu:~/Documents/activities/CS5003-Live$
```

✓ Three Processes.

```
tadeeb@TadeebUbuntu:~/Documents/activities/CS5003-Live$ python3 pec.py -l 1:0,4:100,4:100 -S SWITCH_ON_IO -I IO_RUN_LATER -c
Time PID: 0 PID: 1 PID: 2 CPU Ios
1 RUN:io READY READY 1
2 SUSPEND RUN:cpu READY 1 1
3 SUSPEND RUN:cpu READY 1 1
4 SUSPEND RUN:cpu READY 1 1
5 SUSPEND RUN:cpu READY 1 1
6 SUSPEND DONE RUN:cpu 1 1
7* READY DONE RUN:cpu 1
8 READY DONE RUN:cpu 1
9 READY DONE RUN:cpu 1
10 RUN:io_done DONE 1
tadeeb@TadeebUbuntu:~/Documents/activities/CS5003-Live$ python3 pec.py -l 1:0,4:100,4:100 -S SWITCH_ON_IO -I IO_RUN_IMMEDIATE -c
Time PID: 0 PID: 1 PID: 2 CPU Ios
1 RUN:io READY READY 1
2 SUSPEND RUN:cpu READY 1 1
3 SUSPEND RUN:cpu READY 1 1
4 SUSPEND RUN:cpu READY 1 1
5 SUSPEND RUN:cpu READY 1 1
6 SUSPEND DONE RUN:cpu 1 1
7* RUN:io_done DONE READY 1
8 DONE DONE RUN:cpu 1
9 DONE DONE RUN:cpu 1
10 DONE DONE RUN:cpu 1
tadeeb@TadeebUbuntu:~/Documents/activities/CS5003-Live$
```

- -P: PROGRAM specific controls over programs, the no of CPU to be issued and the IO been used. It tells about the state of execution.

```
tadeeb@TadeebUbuntu:~/Documents/activities/CS5003-Live$ python3 pec.py -l 5:80 -c -P c2
Time PID: 0 CPU Ios
1 RUN:cpu 1
2 RUN:cpu 1
tadeeb@TadeebUbuntu:~/Documents/activities/CS5003-Live$ python3 pec.py -l 5:80 -c -P c6
Time PID: 0 CPU Ios
1 RUN:cpu 1
2 RUN:cpu 1
3 RUN:cpu 1
4 RUN:cpu 1
5 RUN:cpu 1
6 RUN:cpu 1
tadeeb@TadeebUbuntu:~/Documents/activities/CS5003-Live$ python3 pec.py -l 5:80 -c -P i
Time PID: 0 CPU Ios
1 RUN:io 1
2 SUSPEND 1
3 SUSPEND 1
4 SUSPEND 1
5 SUSPEND 1
6 SUSPEND 1
7* RUN:io_done 1
tadeeb@TadeebUbuntu:~/Documents/activities/CS5003-Live$ python3 pec.py -l 1:100 -c -P i
Time PID: 0 CPU Ios
1 RUN:io 1
2 SUSPEND 1
3 SUSPEND 1
4 SUSPEND 1
5 SUSPEND 1
6 SUSPEND 1
7* RUN:io_done 1
tadeeb@TadeebUbuntu:~/Documents/activities/CS5003-Live$
```

```
tadeeb@TadeebUbuntu:~/Documents/activities/CS5003-Live$ python3 pec.py -l 1:0 -c -P i
Time PID: 0 CPU Ios
1 RUN:io 1
2 SUSPEND 1
3 SUSPEND 1
4 SUSPEND 1
5 SUSPEND 1
6 SUSPEND 1
7* RUN:io_done 1
tadeeb@TadeebUbuntu:~/Documents/activities/CS5003-Live$
```

- -L: IO_LENGTH tells how long an IO will take to be done.
- If the length given is 1 the 1 IO is issued, 1 suspend state is seen.
- If the length given is 5, then 5 IOs are issued, with first in the io state, then 5 suspended states then the io_done state, which is done executing.
- If there are 100% chances of CPU to be issued then there is no time taken for the instruction given.

```
tadeeb@TadeebUbuntu:~/Documents/activities/CS5003-Live$ python3 pec.py -l 1:0 -c -L 1
Time   PID: 0      CPU   IOs
1      RUN:io    1
2      SUSPEND   1
3*     RUN:io_done 1
tadeeb@TadeebUbuntu:~/Documents/activities/CS5003-Live$ python3 pec.py -l 1:0 -c -L 5
Time   PID: 0      CPU   IOs
1      RUN:io    1
2      SUSPEND   1
3      SUSPEND   1
4      SUSPEND   1
5      SUSPEND   1
6      SUSPEND   1
7*     RUN:io_done 1
tadeeb@TadeebUbuntu:~/Documents/activities/CS5003-Live$ python3 pec.py -l 1:100 -c -L 5
Time   PID: 0      CPU   IOs
1      RUN:cpu    1
tadeeb@TadeebUbuntu:~/Documents/activities/CS5003-Live$
```