Homework #4a: Scheduling (100 points)

Submit a compressed (.tgz) file with **source code** and **Makefile** to Canvas

For this assignment, you must implement a *scheduling simulator* in **C++**. Your program *must* meet the following requirements:

- Takes **two** or **three** *valid* command line arguments:
    - `UNIX> ./hw4 sim_time algorithm [time_slice]`
        - `sim_time` – total simulation time (arbitrary units)
        - `algorithm` – scheduling algorithm to simulate
            - **`FCFS`** — First Come First Serve
            - **`SJF`** — Shortest Job First (nonpreemptive)
            - **`RR`** — Round Robin
        - `[time_slice]` — optional argument specifying RR time quantum

- Reads a list of “processes” to schedule from **stdin**:
    - One “process” per line
        - Arbitrary number of lines / processes
    - Each line contains *three* integers (separated by white space), e.g.,
        - “`1    0    24`”
            - First number is **process id** (e.g., PID 1)
            - Second number is **arrival time** (e.g., arrives at time 0)
            - Third number is **CPU burst time** (e.g., 24 units)
    - Processes can be listed in **any order**

- Scheduler writes progress to **stdout**, including *when*:
    - Processes have been *scheduled*
    - Processes have *terminated*
    - Processes have been *suspended* (RR only)

- Scheduler maintains performance statistics and writes results to **stderr** (after simulation completes).
    - Total **throughput**
    - Average **wait** time
    - Average **turnaround** time
    - Number of **remaining processes**

- Your code must compile / run on the Linux image provided.

HINTS:

- C++ standard template library multimap, list, queue
- <time.h> NOT needed!
- int tmp; while(cin >> tmp){ … }
- Work through algorithms with pencil / paper first!!

EXAMPLES:

```
//command line args
UNIX> ./hw4
usage: ./a.out sim_time algorithm [time_slice]


//FCFS example 1 (pg 274-275)
UNIX> cat book_input1.txt
=====================================================
         0: scheduling PID     1, CPU =       24
        24:            PID        1  terminated
        24: scheduling PID     2, CPU =        3
        27:            PID        2  terminated
        27: scheduling PID     3, CPU =        3
        30:            PID        3  terminated
=====================================================
Throughput          =       3
Avg wait time       = 17.00
Avg turnaround time = 27.00
Remaining tasks     =       0

//FCFS example 2 (pg 275)
UNIX> cat book_input2.txt
1 0 3
2 0 3
3 0 24

UNIX> ./hw4 50 FCFS < book_input2.txt
=====================================================
         0: scheduling PID     1, CPU =        3
         3:            PID        1  terminated
         3: scheduling PID     2, CPU =        3
         6:            PID        2  terminated
         6: scheduling PID     3, CPU =       24
        30:            PID        3  terminated
=====================================================
Throughput          =       3
Avg wait time       =  3.00
Avg turnaround time = 13.00
Remaining tasks     =       0
```

```
//SJF example (pg 276)
UNIX> cat book_input3.txt
1 0 6
2 0 8
3 0 7
4 0 3

UNIX> ./hw4 50 SJF < book_input3.txt
=====================================================
         0: scheduling PID        4, CPU =        3
         3:            PID        4  terminated
         3: scheduling PID        1, CPU =        6
         9:            PID        1  terminated
         9: scheduling PID        3, CPU =        7
        16:            PID        3  terminated
        16: scheduling PID        2, CPU =        8
        24:            PID        2  terminated
=====================================================
Throughput          =       4
Avg wait time       =  7.00
Avg turnaround time = 13.00
Remaining tasks     =       0


//example with differing arrival times
UNIX> cat sjf_input.txt
1 0 10
2 2 7
3 0 15

UNIX> ./hw4 50 SJF < sjf_input.txt
=====================================================
         0: scheduling PID        1, CPU =       10
        10:            PID        1  terminated
        10: scheduling PID        2, CPU =        7
        17:            PID        2  terminated
        17: scheduling PID        3, CPU =       15
        32:            PID        3  terminated
=====================================================
Throughput          =       3
Avg wait time       =  8.33
Avg turnaround time = 19.00
Remaining tasks     =       0
```

```
//RR example (pg 280)
UNIX> cat book_input1.txt
=====================================================
         0: scheduling PID        1, CPU =       24
         4: suspending PID        1, CPU =       20
         4: scheduling PID        2, CPU =        3
         7:            PID        2 terminated
         7: scheduling PID        3, CPU =        3
        10:            PID        3 terminated
        10: scheduling PID        1, CPU =       20
        14: suspending PID        1, CPU =       16
        14: scheduling PID        1, CPU =       16
        18: suspending PID        1, CPU =       12
        18: scheduling PID        1, CPU =       12
        22: suspending PID        1, CPU =        8
        22: scheduling PID        1, CPU =        8
        26: suspending PID        1, CPU =        4
        26: scheduling PID        1, CPU =        4
        30:            PID        1 terminated
=====================================================
Throughput          =       3
Avg wait time       =  5.67
Avg turnaround time = 15.67
Remaining tasks     =       0


//your solution must work with arbitrarily large inputs!
// e.g.,

UNIX> wc -l large_input.txt
1000 large_input.txt


UNIX> head large_input.txt
17760 5285 582
19182 9627 826
16939 2013 858
9648 5979 926
6694 120 872
29479 967 770
17792 8313 137
25227 364 213
23931 6752 966
16418 6095 193


UNIX> ./hw4 10000 FCFS < large_input.txt > /dev/null
=====================================================
Throughput          =      22
Avg wait time       = 3783.73
Avg turnaround time = 4439.09
Remaining tasks     =     978
```

```
UNIX> ./hw4 10000 SJF < large_input.txt > /dev/null
=====================================================
Throughput          =     150
Avg wait time       = 243.99
Avg turnaround time = 310.59
Remaining tasks     =     850
```

```
UNIX> ./hw4 10000 RR 10 < large_input.txt > /dev/null
=====================================================
Throughput          =      11
Avg wait time       = 3415.27
Avg turnaround time = 3437.36
Remaining tasks     =     989
```

```
UNIX> ./hw4 10000 RR 100 < large_input.txt > /dev/null
=====================================================
Throughput          =      17
Avg wait time       = 4417.94
Avg turnaround time = 4553.29
Remaining tasks     =     983
```