FIFO Quantum to SJF

Priya Jeyaprakash, Xinyi Li, Zion Jones, Nick Garcia

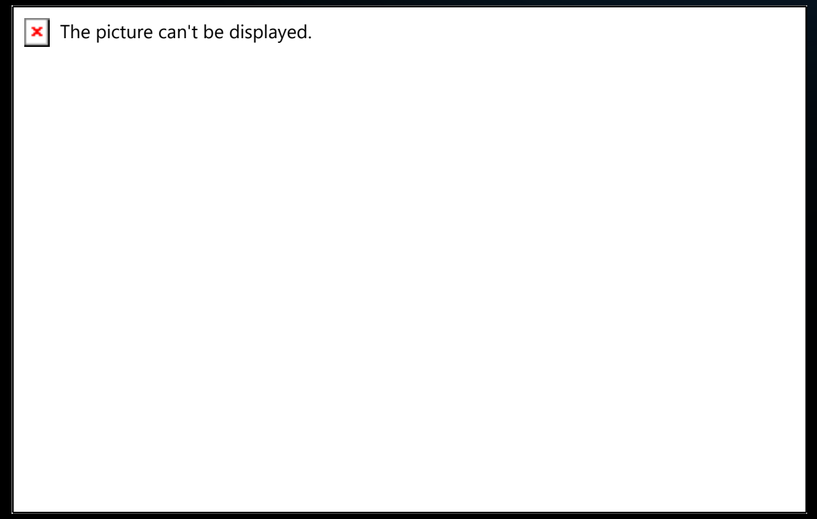
CPU Schedulers

- CPU Schedulers Manage Processes that need to run on the CPU
- Good practices for a scheduler are :
  - Flexibility in CPU Occupation
  - Fairness
  - Increased Performance

```
TEST CASE 1: Basic 3 Jobs
    SJF Scheduler:
    FIFO Scheduler:
    Hybrid Quantum (Q=2):
Scheduler    | Avg Wait | Avg Turnaround | Avg Response
=============================================================
FIFO         |     0.33 |           2.00 |         1.67
SJF          |     0.89 |           3.11 |         0.00
Hybrid       |     1.33 |           3.56 |         0.22
=============================================================


TEST CASE 2: Same Arrival Time
    SJF Scheduler:
    FIFO Scheduler:
    Hybrid Quantum (Q=2):
Scheduler    | Avg Wait | Avg Turnaround | Avg Response
=============================================================
FIFO         |     0.25 |           1.75 |         1.50
SJF          |     1.88 |           3.50 |         0.00
Hybrid       |     2.12 |           3.75 |         1.56
=============================================================


TEST CASE 3: Short Job Arrives Later
    SJF Scheduler:
    FIFO Scheduler:
    Hybrid Quantum (Q=2):
Scheduler    | Avg Wait | Avg Turnaround | Avg Response
=============================================================
FIFO         |     0.33 |           1.67 |         1.33
SJF          |     0.78 |           2.89 |         0.00
Hybrid       |     1.22 |           3.33 |         1.11
=============================================================
```



FIFO

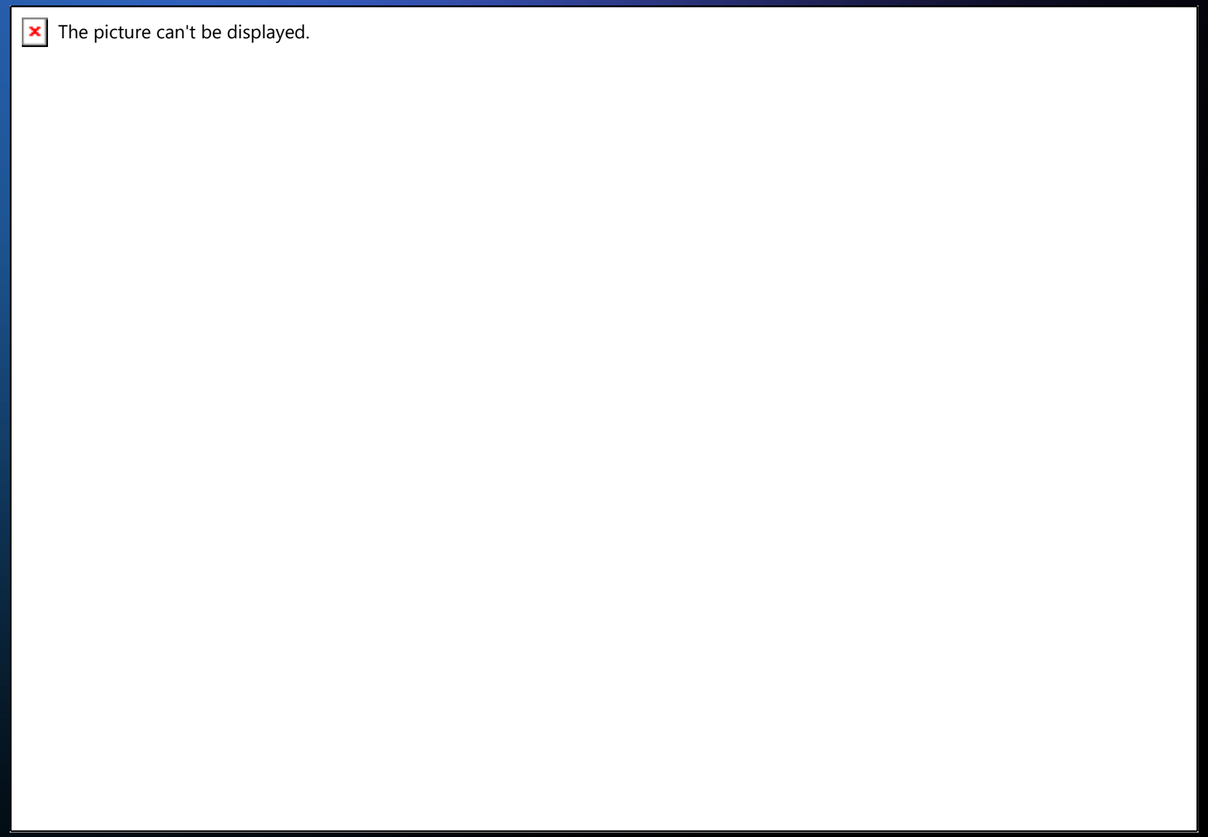
- First in First Out
  - Earliest arriving job will execute first, with the subsequent jobs running in the order in which they arrived.
  - Once a job is started it will complete before CPU is used by another scheduler

findNextJob()

Function to be used in FIFO to find what job is the next in the arrival queue

FIFO()

FIFO function to be applied onto existing job queue to save CPU processing data



The picture can't be displayed.

The picture can't be displayed.

SJF

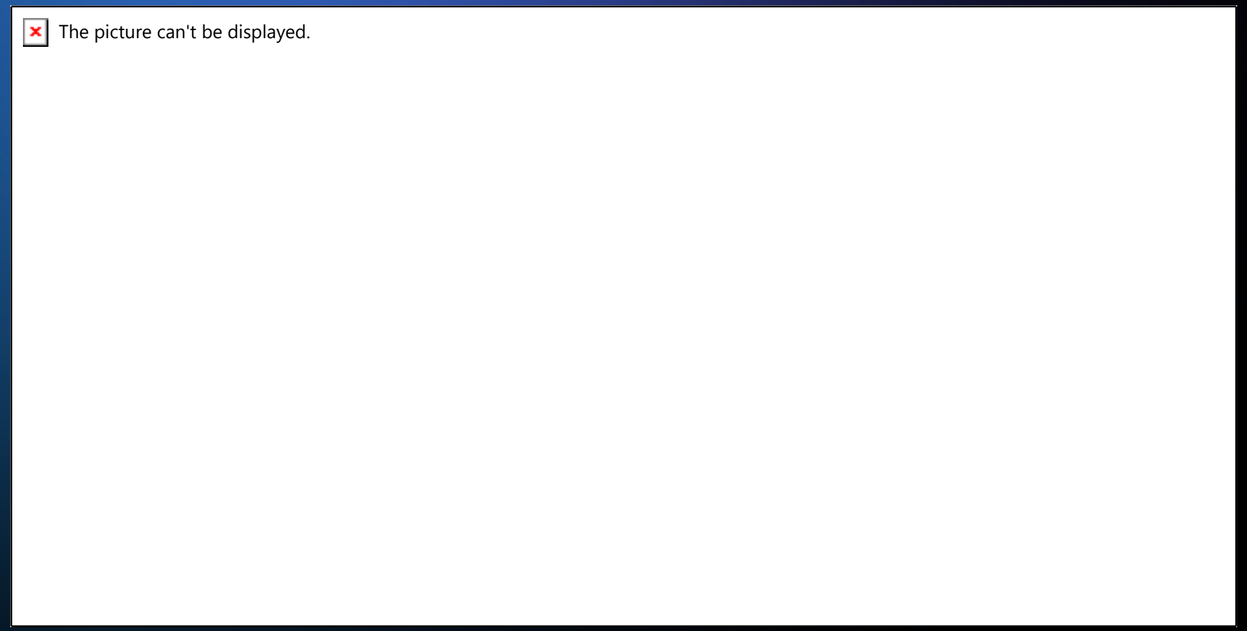
- Shortest Job First
  - Will pick the shortest arrived job at time of scheduling and execute to completion

bubble()

Function used by SJF to sort all incoming jobs so it goes in order of shortest to longest using bubblesort algorithm.

swap()

Helper function used in bubble() to implement the bubblesort sorting algorithm.



sjf_scheduler()

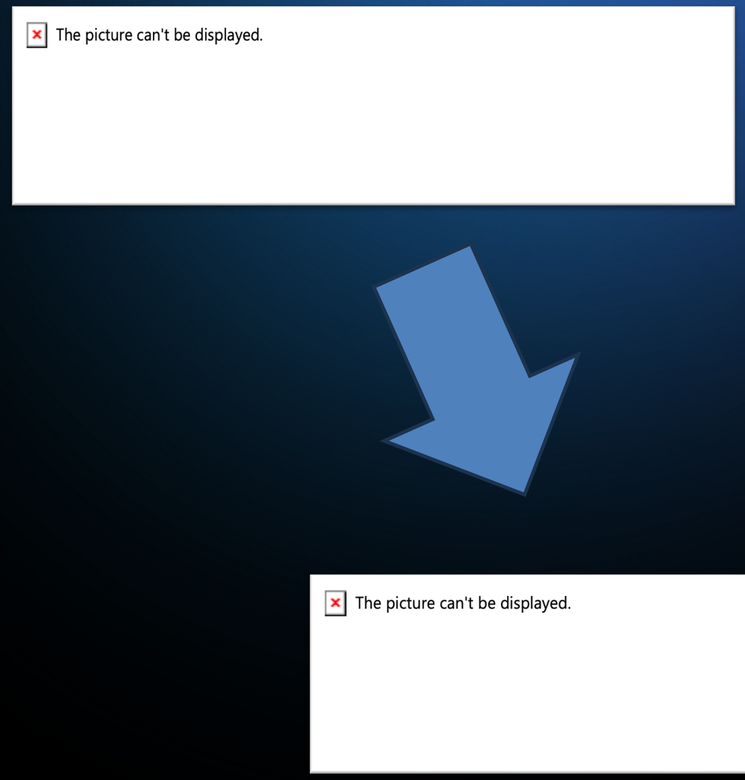
SJF function to be applied onto existing job queue to save CPU processing data
(e.g. turnaround and wait time)

Pros vs. Cons

- FIFO
  - Pros
    - Easy to understand
    - Simple to implement
    - No starvation
    - No context switching time
  - Cons
    - Can have long wait time for big job first
- SJF
  - Pros
    - Low average turn around time
  - Cons
    - Big jobs take a very long time to turn around
    - Big jobs can be starved
    - Low predictability

Quantum FIFO to SJF

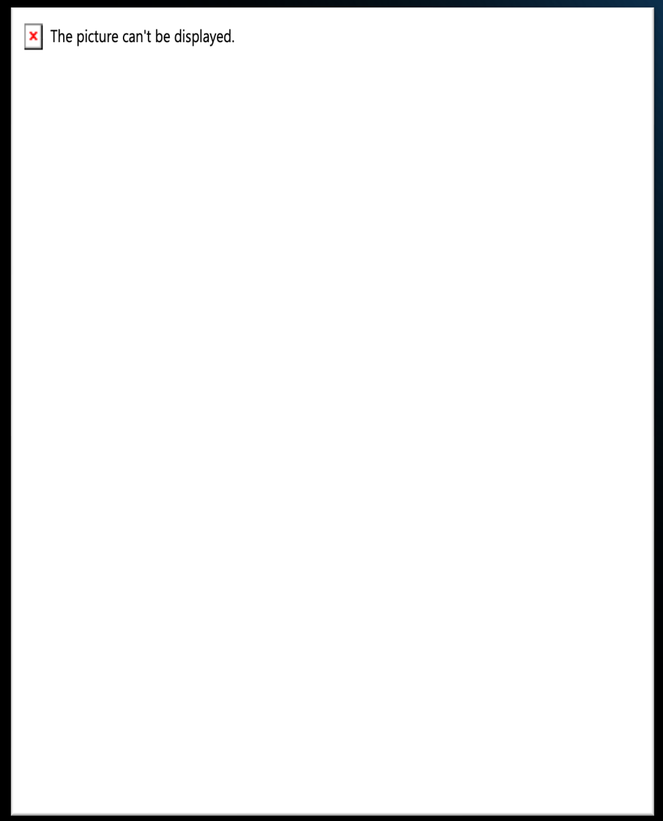
- A **quantum** is a fixed time slice given to a CPU scheduler. In all, it is the maximum continuous time a process or job will be allowed to run before the scheduler pauses it and switches to another process. In our case, the quantum represents the time FIFO is allowed to run before switching to SJF



Quantum.c
A deep dive

This is our ***hybrid_quantum()*** function, which is responsible for implementing and maintaining our hybrid scheduler where we...

1. Start with FIFO in FIFO.c
2. Run until a fixed quantum specified expires
3. Then switch to SJF in SJF.c for the remaining jobs



This system handles the task of ensuring early jobs are handled predictably in an arrival-order fashion, then when the time window closes, the system will switch to shortest burst time to ensure optimization

Quantum.c
A deep dive

For our **data structures** within quantum.c we have the following...

- **work[]**: a local copy of all jobs so the original input isn't modified or corrupted during the process
- **remaining[]**: keeps track of unfinished burst time of each job

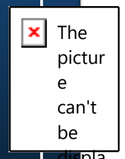
The picture can't be displayed.

The picture can't be displayed.

Quantum.c
A deep dive

This helper function ***find_next_fifo()*** is used to find the earliest arrival job that has arrived before the current time and is unfinished. This essentially allows the limited FIFO phase to behave like a ready queue processed in arrival order.

earliest_future_arrival() is used for when no job has arrived yet and we need to jump the clock forward to the next arrival. It also prevents CPU idling and correctly simulates scheduler behavior.

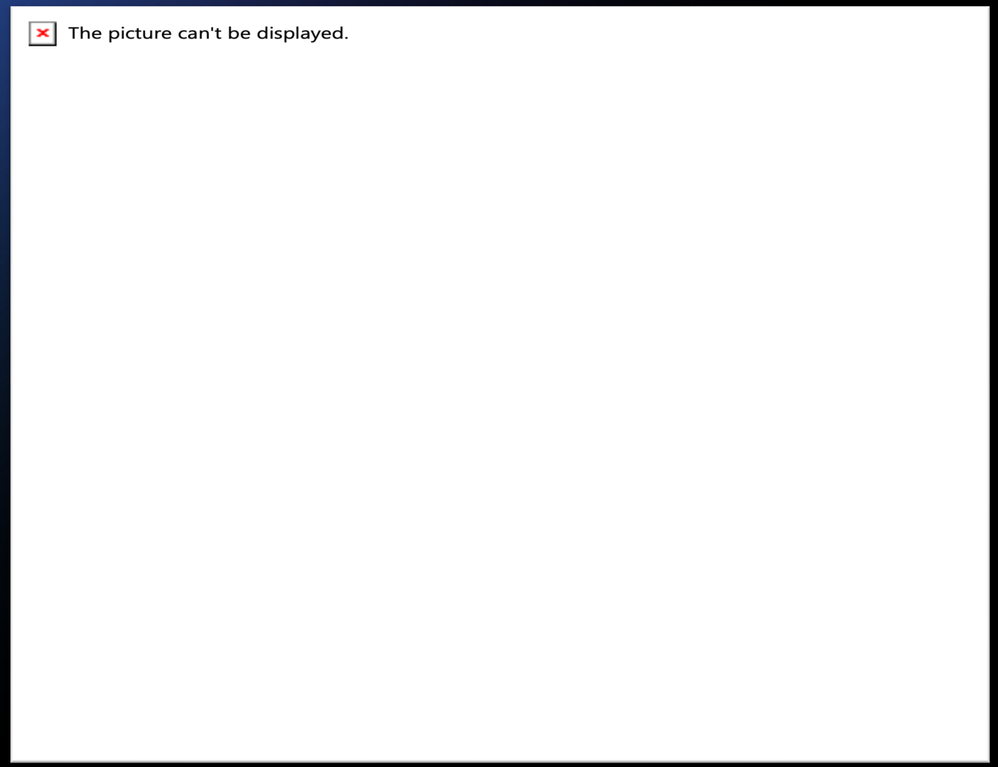


Quantum.c
A deep dive

In **the FIFO phase**, behavior consists of running only until time < quantum

and in each loop we..

1. pick the next FIFO job with the helper function *find_next_fifo()*
2. then jump to next arrival if none arrived yet
3. compute response if this is the first time running
4. then compute the amount of time left in the quantum
5. lastly, either the job finishes before the quantum or the quantum ends and we must break and switch to SJF

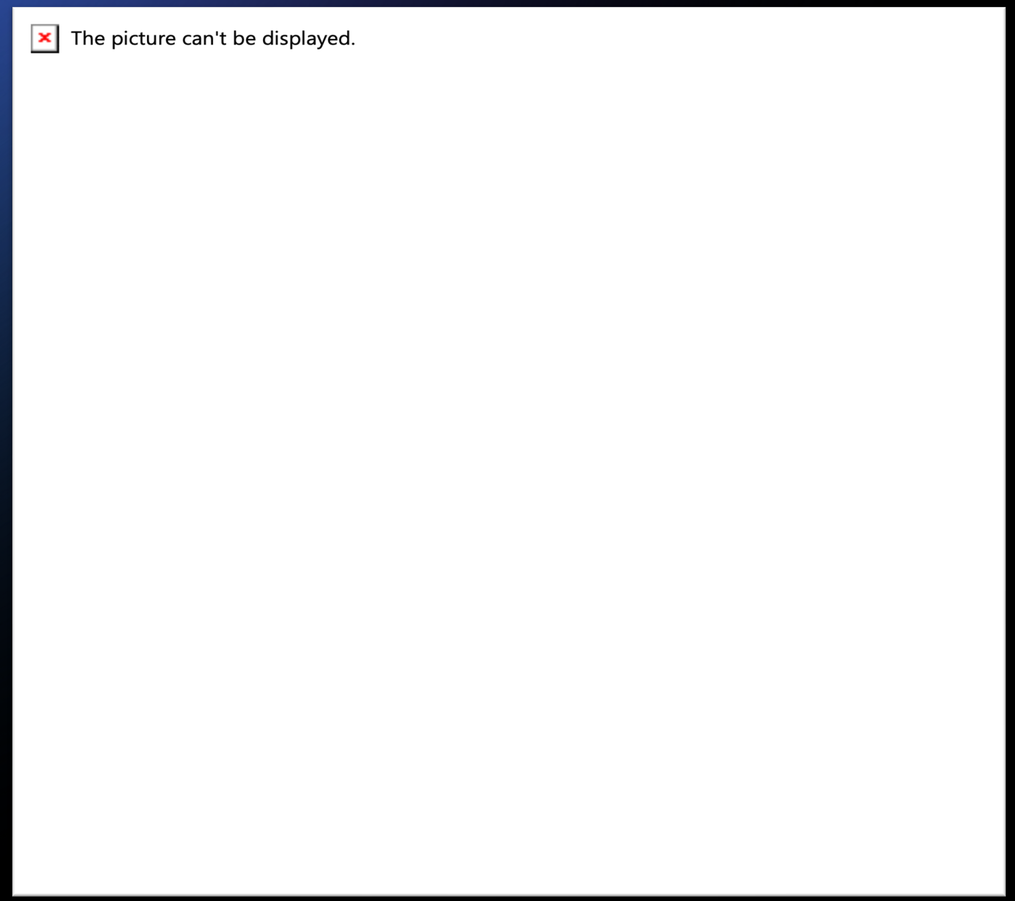


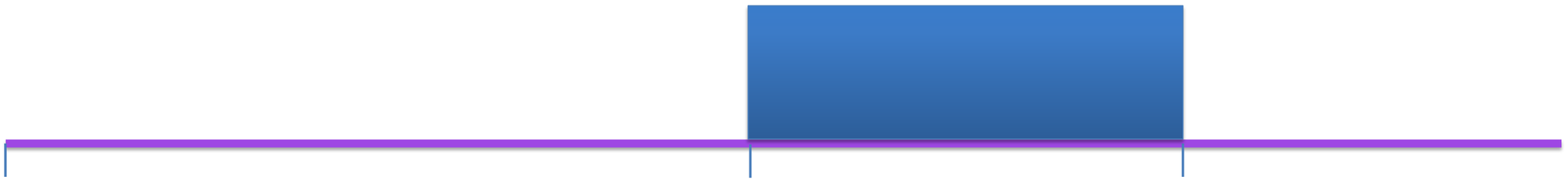
Quantum.c
A deep dive

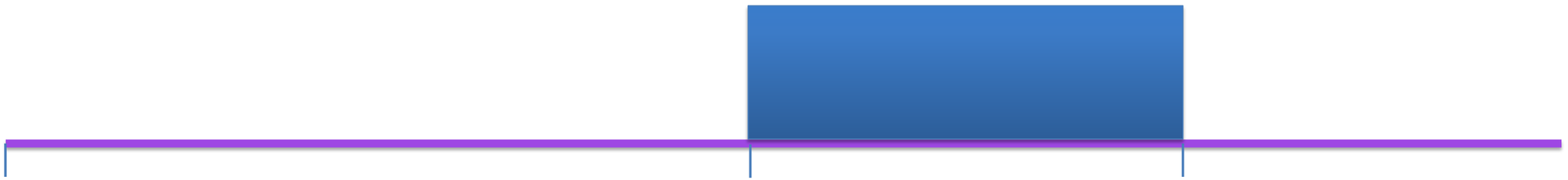
In **the SJF phase**, behavior consists of running only for the work that remains after the quantum ends

and in each loop we..

1. choose the job with the smallest *remaining[]* time
2. jump to next arrival if nothing is available
3. compute response if this is the first run
4. run job to completion
5. then update metrics







Comparisons

This code gives us the average of wait time, turnaround time, and response time that we see used from the structs that are being used for the schedulers we've implemented. We used this function after calling test cases for our schedulers.

Key Takeaw

As we can see, different schedulers prioritiz
- FIFO focuses on simplicity, SJF focuses o
  time, and our Quantum FIFO to SJF tries

The hybrid model shows better balance than
alone.
- It avoids the inefficiency of FIFO and the sta
  SJF has, which creates a more fair and respo

```
TEST CASE 1: Basic 3 Jobs
    SJF Scheduler:
    FIFO Scheduler:
    Hybrid Quantum (Q=2):
Scheduler   | Avg Wait | Avg Turnaround | Avg Response
============================================================
FIFO        |     0.33 |           2.00 |         1.67
SJF         |     0.89 |           3.11 |         0.00
Hybrid      |     1.33 |           3.56 |         0.22
============================================================


TEST CASE 2: Same Arrival Time
    SJF Scheduler:
    FIFO Scheduler:
    Hybrid Quantum (Q=2):
Scheduler   | Avg Wait | Avg Turnaround | Avg Response
============================================================
FIFO        |     0.25 |           1.75 |         1.50
SJF         |     1.88 |           3.50 |         0.00
Hybrid      |     2.12 |           3.75 |         1.56
============================================================


TEST CASE 3: Short Job Arrives Later
    SJF Scheduler:
    FIFO Scheduler:
    Hybrid Quantum (Q=2):
Scheduler   | Avg Wait | Avg Turnaround | Avg Response
============================================================
FIFO        |     0.33 |           1.67 |         1.33
SJF         |     0.78 |           2.89 |         0.00
Hybrid      |     1.22 |           3.33 |         1.11
============================================================
```

Future Ide

- Compare More Schedulers
  - Round Robin
  - Shortest Job Remaining
- Create a Round Robin sche
  FIFO in one Quantum and
  next, continuing until jobs c