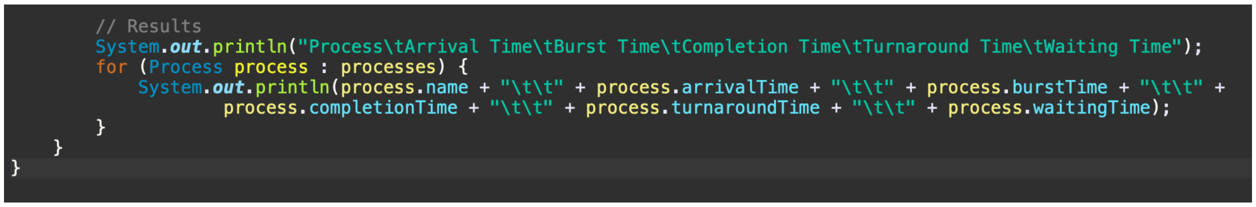
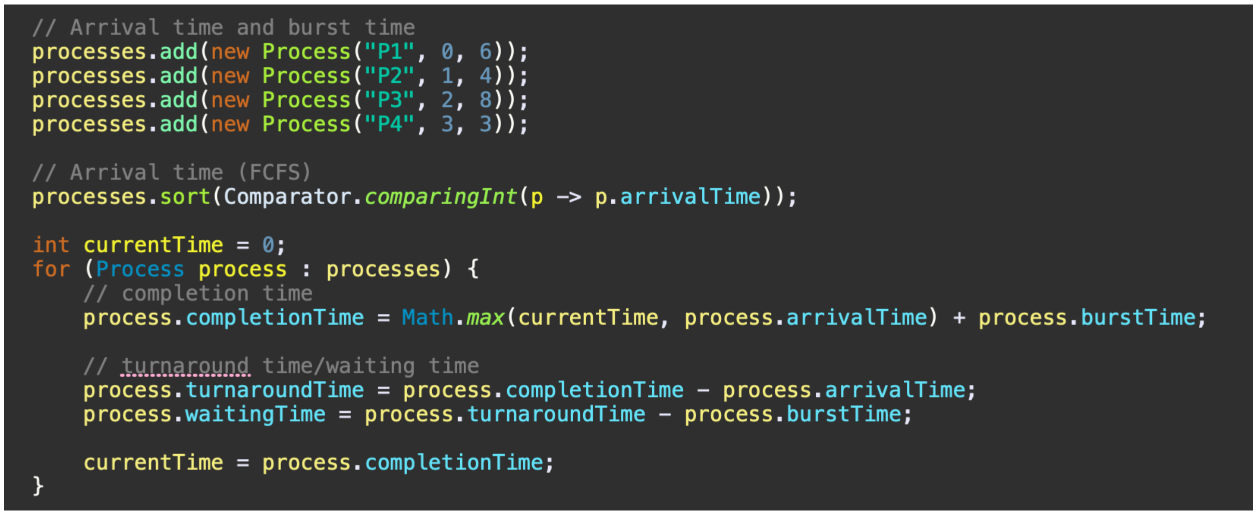
Project 1 second iteration



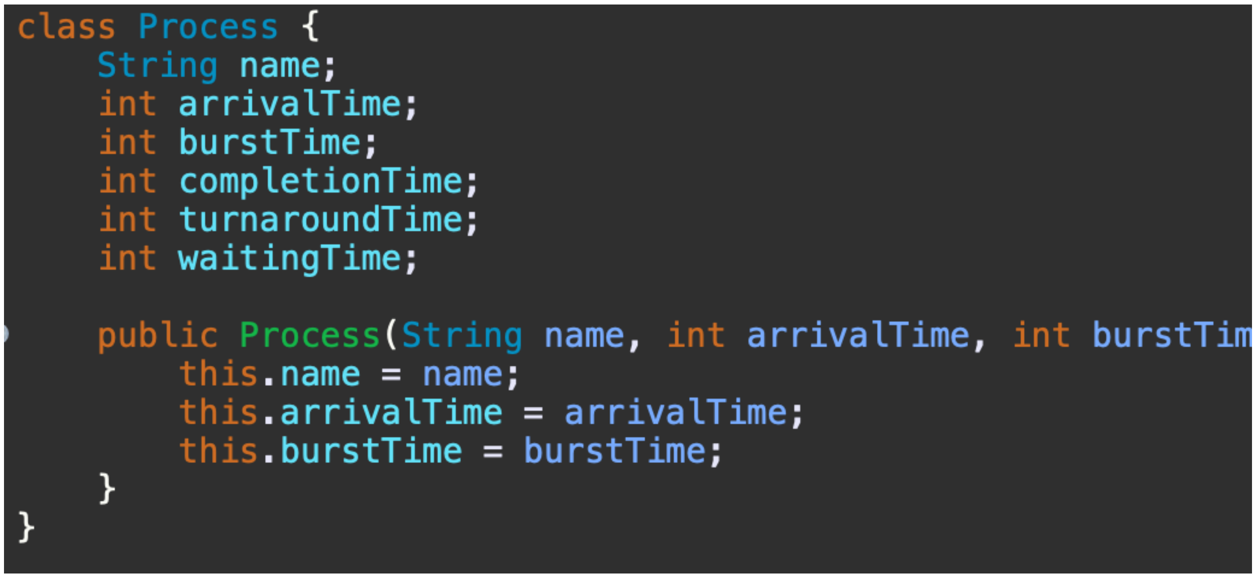
```java
        // Results
        System.out.println("Process\tArrival Time\tBurst Time\tCompletion Time\tTurnaround Time\tWaiting Time");
        for (Process process : processes) {
            System.out.println(process.name + "\t\t" + process.arrivalTime + "\t\t" + process.burstTime + "\t\t" +
                    process.completionTime + "\t\t" + process.turnaroundTime + "\t\t" + process.waitingTime);
        }
    }
}
```

New File for this results class



```java
// Arrival time and burst time
processes.add(new Process("P1", 0, 6));
processes.add(new Process("P2", 1, 4));
processes.add(new Process("P3", 2, 8));
processes.add(new Process("P4", 3, 3));

// Arrival time (FCFS)
processes.sort(Comparator.comparingInt(p -> p.arrivalTime));

int currentTime = 0;
for (Process process : processes) {
    // completion time
    process.completionTime = Math.max(currentTime, process.arrivalTime) + process.burstTime;

    // turnaround time/waiting time
    process.turnaroundTime = process.completionTime - process.arrivalTime;
    process.waitingTime = process.turnaroundTime - process.burstTime;

    currentTime = process.completionTime;
}
```

New file for arrival time/burst time



```java
class Process {
    String name;
    int arrivalTime;
    int burstTime;
    int completionTime;
    int turnaroundTime;
    int waitingTime;

    public Process(String name, int arrivalTime, int burstTim
        this.name = name;
        this.arrivalTime = arrivalTime;
        this.burstTime = burstTime;
    }
}
```

New file for Main Method Class

Important notes: Improve overall structure of code and reorganzie to streamline the processes. We're going to do some more research into the java language in order to run the software better.

Things to improve in iteration 2:

- **<u>Modularization</u>:** Breaking down the code into functions or methods.
- **<u>Documentation</u>:** Adding comments and documentation to describe the purpose of the code.
- **<u>Dynamic Process Input</u>:** Instead of hardcoding process data, allow dynamic input. Prompt the user to input the number of processes and their details.
- **<u>User-Friendly Output:</u>** Improve the format and readability of the output for the user.
- **<u>Testing</u>:** Test the code more often when changing files in order to make sure nothing is broken as we're working on the code.
- **<u>Scalability:</u>** Work with the code to ensure it will run better with more methods/classes.