```
import java.util.*;

public class OSSimulator {

    static class Process { // creates class to store process info
        int processId;
        int arrivalTime;
        int burstDuration;
        int memoryNeeded;
        int waitingTime;
        int turnaroundTime;

        public Process(int processId, int arrivalTime, int burstDuration, int memoryNeeded) { // these are the variables that will be calculated during emulation
            this.processId = processId;
            this.arrivalTime = arrivalTime;
            this.burstDuration = burstDuration;
            this.memoryNeeded = memoryNeeded;
        }
    }

    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        // first we get number of processes
        System.out.print("Enter number of processes: ");
```

```
int numberOfProcesses = scanner.nextInt();

List<Process> processes = new ArrayList<>();

// then we input process details
for (int i = 0; i < numberOfProcesses; i++) {
        System.out.println("Enter info for process " + (i + 1));
        System.out.print("Arrival Time: ");
        int arrival = scanner.nextInt();
        System.out.print("Burst Time: ");
        int burst = scanner.nextInt();
        System.out.print("Memory Required: ");
        int memory = scanner.nextInt();
        processes.add(new Process(i + 1, arrival, burst, memory));
}

// after we sort by arrival time
processes.sort(Comparator.comparingInt(p -> p.arrivalTime));

int currentTime = 0;
double totalWaitingTime = 0;
double totalTurnaroundTime = 0;

// calculating waiting and turnaround times
for (Process p : processes) {
        if (currentTime < p.arrivalTime) {
```

```
                currentTime = p.arrivalTime;
            }
            p.waitingTime = currentTime - p.arrivalTime;
            p.turnaroundTime = p.waitingTime + p.burstDuration;
            currentTime += p.burstDuration;

            totalWaitingTime += p.waitingTime;
            totalTurnaroundTime += p.turnaroundTime;
        }

        // Displaying results
        System.out.println("\nResults:");
        System.out.println("PID\tArrival\tBurst\tWaiting\tTurnaround");
        for (Process p : processes) {
            System.out.printf("%d\t%d\t%d\t%d\t%d\n",
                    p.processId, p.arrivalTime, p.burstDuration, p.waitingTime, p.turnaroundTime);
        }

        // Shows averages
        System.out.printf("\nAverage Waiting Time: %.2f\n", totalWaitingTime / numberOfProcesses);
        System.out.printf("Average Turnaround Time: %.2f\n", totalTurnaroundTime /
numberOfProcesses);
    }
}
```