import java.util.Arrays;

public class FCFS {

    // Class to represent a Process

    static class Process {

        int pid;       // Process ID

        int arrival;   // Arrival Time

        int burst;     // Burst Time

        int finish;    // Finish Time

        int turnaround; // Turnaround Time

        int waiting;   // Waiting Time

        Process(int pid, int arrival, int burst) {

            this.pid = pid;

            this.arrival = arrival;

            this.burst = burst;

        }

    }

    public static void main(String[] args) {

        // Hardcoded process data: {Process ID, Arrival Time, Burst Time}

        Process[] processes = {

            new Process(1, 0, 5),

            new Process(2, 1, 3),

            new Process(3, 2, 8),

            new Process(4, 3, 6)

        };

        // Calculate Finish Time, Turnaround Time, and Waiting Time

        calculateTimes(processes);

        // Print the results

        System.out.printf("%-10s %-15s %-15s %-15s %-15s %-15s\n",

            "Process", "Arrival Time", "Burst Time", "Finish Time", "Turnaround", "Waiting Time");

        for (Process p : processes) {

            System.out.printf("%-10d %-15d %-15d %-15d %-15d %-15d\n",

                p.pid, p.arrival, p.burst, p.finish, p.turnaround, p.waiting);

        }

        // Calculate and print average TAT and WT

        double avgTurnaround = calculateAverage(processes, "turnaround");

        double avgWaiting = calculateAverage(processes, "waiting");

        System.out.printf("\nAverage Turnaround Time: %.2f\n", avgTurnaround);

        System.out.printf("Average Waiting Time: %.2f\n", avgWaiting);

    }

    // Method to calculate Finish Time, Turnaround Time, and Waiting Time

    public static void calculateTimes(Process[] processes) {

        int currentTime = 0;

        for (Process p : processes) {

            // If the CPU is idle, move the time to the arrival of the next process

            if (currentTime < p.arrival) {

                currentTime = p.arrival;

            }

            // Calculate Finish Time

            p.finish = currentTime + p.burst;

            // Calculate Turnaround Time (Finish Time - Arrival Time)

            p.turnaround = p.finish - p.arrival;

            // Calculate Waiting Time (Turnaround Time - Burst Time)

            p.waiting = p.turnaround - p.burst;

            // Update current time

            currentTime = p.finish;

        }

    }

    // Method to calculate average Turnaround Time or Waiting Time

    public static double calculateAverage(Process[] processes, String type) {

        int total = 0;

        for (Process p : processes) {

            if (type.equals("turnaround")) {

                total += p.turnaround;

            } else if (type.equals("waiting")) {

                total += p.waiting;

            }

        }

        return (double) total / processes.length;

    }

}