CPU Scheduler and Round Robin CPU Scheduling Project Report

Instructions:

1. Using the Java IDE of your choice, copy and open the three Java classes, under the path "CSCI509\CPUScheduler\src\CSCI509_CPU_Scheduler", and the TESTCSV.csv.
2. Run "TheRunnable.java"
3. The user will be requested to enter the filename of the CSV (which is "TESTCSV.csv") and then the desired Time Quantum (the program will fail if the number is too high).
4. After the user inputs the requested information, the program will run, finish, and display the "CPU Utilization", "Throughput", "Average Waiting Time", and "Average Turnaround Time".

Project and Program Explained:

According to "What is CPU Scheduling?", CPU scheduling is a process which allows one process to use the CPU while the execution of another process is on hold (in waiting state) due to unavailability of any resource like I/O etc, thereby making full use of CPU (What is CPU Scheduling?). The aim of CPU scheduling is to make the system efficient, fast and fair (What is CPU Scheduling?).

According to "Program for Round Robin scheduling", Round Robin is a CPU scheduling algorithm where each process is assigned a fixed time slot in a cyclic way. It is simple, easy to implement, and starvation-free as all processes get a fair share of CPU (Program for Round Robin scheduling).

According to "Program for Round Robin scheduling", One of the most commonly used technique in CPU scheduling as a core (Program for Round Robin scheduling). It is preemptive as processes are assigned CPU only for a fixed slice of time at most (Program for Round Robin scheduling). The disadvantage of it is more overhead of context switching (Program for Round Robin scheduling).

For this project, there are three classes that are called "TheRunnable", "TheProcess", and "TheRRScheduler". The objective of this project was to create a program that simulates Round Robin CPU scheduling. The program created generates a schedule of processes. The processes are stored in the CSV file. The program accepts two parameters. The first parameter is the name of processes CSV file. The second parameter is the desired time quantum of the user.

The "TheProcess" class was created to read processes and all other information that comes from the "CSVTest.csv" file. The three variables that are in this specific class that is related to the CSV table are called "PID", "ArrivalTime", and "BurstTime".

The "TheRRScheduler" class is where the simulation of Round Robin CPU Scheduling takes place. Below are some important features that will help one to understand both the program and the importance of Round Robin to the CPU Scheduler.

In this class, there is a "ProcessesList", "ReadyQueue", "Timer" (serves as the purpose of a clock for the running program), "TimeQuantum" (requires user input), CPU (only holds one process at a time), "ContextSwitch" (tracks the switches of each process), and a "Counter".

Inside the RoundRobin() method, a while loop exists to check if our "ReadyQueue", "ProcessesList", or the "CPU" is not empty. Unless we are finished with the processes list, the loop will continue running. Under the conditionals of the while loop, there is a for loop that goes through the "ProcessesList" and verifies which processes' "ArrivalTime" is equal to the "Timer" (clock). Once a process successfully accomplishes verification through the "if" statement, it will be loaded onto the "ReadyQueue".

The "CPU" undergoes verification on whether it is empty or not. If it is empty, then the "ReadyQueue" loads process onto the "CPU". Next, the "Counter" records the current process of the "CPU" and that process' "ServiceTime". Afterward, we have an "if" statement that confirms whether or not current process' "BurstTime" and "ServiceTime" are equal. If they are confirmed to be equal, then the process is done and can be removed from the "CPU".

The program records the processes' completion time before it is removed from the "CPU". Then, that process will be added to the "EndProcesses" list. This enables the program to track the processes that have been completed. Then, the "CPU" becomes equals null which would enable it to accept the subsequent waiting process. The "ContextSwitch" is corrected to match the info within the program. In addition, the "Counter" becomes 0 which then triggers a reset.

In the method, the "else if" statement determines whether the process is greater than the value of the "TimeQuantum". If this happens to be true, the process returns to the "ReadyQueue" to be loaded again, the "CPU" would be updated to null again, the "ContextSwitch" would also be corrected too. In the end, the "Counter" rests to 0.

This process is the way the "CPU" is effectively occupied, and the Round Robin ensures this to be the case. Afterward, the program successfully completes by computing and then printing the pieces of information known as "CPU Utilization", "Throughput", "Average Waiting Time", and "Average Turnaround Time". The program keeps track of the calculations with the following equations:

CPU Utilization = (Burst Time - (ContextTime * 0.01)) / Processes

Throughput = Processes / Timer

Avg. Waiting Time = The total of all Waiting Times / Processes

Avg. Turnaround Time = The total of all Turnaround Times / Processes

the CPU Scheduling after it prints the "CPU Utilization", "Throughput", "Average Waiting Time", and "Average Turnaround Time".

In short, the "TheRunnable" class requests the user to input the filename and desired Time Quantum. Afterward, the list of processes that comes from the CSV file is inputted into the ArrayList. Then, the Round Robin method within the "TheRRScheduler" class is utilized. Finally, the calculations are finished and printed to the console for the user to read after all the processes have been completed.

References:

Program for Round Robin scheduling | Set 1. (2018, September 06). Retrieved from https://www.geeksforgeeks.org/program-round-robin-scheduling-set-1/

What is CPU Scheduling? (n.d.). Retrieved from https://www.studytonight.com/operating-system/cpu-scheduling

TheRunnable.Java

```java
/**
 * Description: The “TheRunnable” class is started to prompt the user for the
 * filename and desired Time Quantum. From there, the csv table transfers the
 * list of processes to the ArrayList.)
 *
 * @author eomed
 */

package CSCI509_CPU_Scheduler;

import java.io.File;
import java.io.IOException;
import java.util.ArrayList;
import java.util.Scanner;

public class TheRunnable {

       /**
        * Parameterized constructor
        *
        * @param name
        *            main(String[] arg) throws IOException
        * @param arg
        *            Parameter
        * @param override
        *            None
        * @param comment
        *            Prompts the user for the filename of CSVFile
        *            and desired Time Quantum
        * @param visiblity
        *            public
        */

       public static void main(String[] arg) throws IOException {
              Scanner kb = new Scanner(System.in);
              System.out.print("Please Enter Filename: ");
              String FileName = kb.nextLine();
              System.out.print("Please Enter Time Quantum: ");
              int tq = kb.nextInt();
              Scanner inputStream = new Scanner(new File(FileName));
              inputStream.useDelimiter("(,|\\v+)");
              inputStream.nextLine();
              ArrayList<CSCI509_CPU_Scheduler.TheProcess> CSVFile = new ArrayList<>();
              while (inputStream.hasNext()) {
                     CSCI509_CPU_Scheduler.TheProcess Process = new
CSCI509_CPU_Scheduler.TheProcess(inputStream.nextInt(),
                                   inputStream.nextInt(), inputStream.nextInt());
                     CSVFile.add(Process);
              }

              CSCI509_CPU_Scheduler.TheRRScheduler MyScheduler = new
CSCI509_CPU_Scheduler.TheRRScheduler(CSVFile, tq);
              MyScheduler.RoundRobin();
              kb.close();
              inputStream.close();
       }
}
```

TheProcess.Java

```
/**
 * Description: This class was created to read processes and all other information
 * that come from the “CSVTest.csv” file. The three variables that are in this
 * specific class that are related from the csv table are called “PID”,
 * “ArrivalTime”, and “BurstTime”.
 *
 * @author eomed
 *
 */

package CSCI509_CPU_Scheduler;




public class TheProcess
{
    public int PID, ArrivalTime, ServiceTime, BurstTime, CompletionTime;

    /**
        * Parameterized constructor
        *
        * @param name
        *           TheProcess(int PID, int ArrivalTime, int BurstTime)
        * @param PID
        *           Parameter
        * @param ArrivalTime
        *           Parameter
        * @param BurstTime
        *           Parameter
        * @param override
        *           None
        * @param comment
        *           Reads the information that come from the “CSVTest.csv” file
        * @param visiblity
        *           public
        */
    public TheProcess(int PID, int ArrivalTime, int BurstTime)
    {
        this.PID = PID; // Unique
        this.ArrivalTime = ArrivalTime;
        this.BurstTime = BurstTime;
        this. ServiceTime = 0;
    }
}
```

TheRRScheduler.Java

```
package CSCI509_CPU_Scheduler;

/**
 * Description: The “TheRRScheduler” class is where the simulation of Round Robin CPU
Scheduling
 * takes place. Below are some important features that will help one to understand
both the
 * program and importance of Round Robin to the CPU Scheduler.
 *
 * @author eomed
 *
 */


import java.util.ArrayList;


//Source Code
public class TheRRScheduler
{
    int Timer;
    ArrayList<CSCI509_CPU_Scheduler.TheProcess> ProcessesList;
    ArrayList<CSCI509_CPU_Scheduler.TheProcess> ReadyQueue;
    ArrayList<CSCI509_CPU_Scheduler.TheProcess> EndProcesses;

    int TimeQuantum;
    int ContextSwitch; // small number that should be less than half the TimeQuantum

    CSCI509_CPU_Scheduler.TheProcess CPU;
    int Counter;

    /**
        * Parameterized constructor
        *
        * @param name
        *            TheRRScheduler(ArrayList<CSCI509_CPU_Scheduler.TheProcess>
ProcessesList,
        *            int TimeQuantum)
        * @param ProcessesList
        *            Parameter
        * @param TimeQuantum
        *            Parameter
        * @param override
        *            None
        * @param comment
        *            Stores the information in the ArrayList for which would be used
in the
        *            RoundRobin() method
        * @param visiblity
        *            public
        *
        */
     public TheRRScheduler(ArrayList<CSCI509_CPU_Scheduler.TheProcess> ProcessesList, 
 int TimeQuantum)
     {
         this.TimeQuantum = TimeQuantum;
         this.ProcessesList = ProcessesList;
     }

     /**
        * Parameterized constructor
        *
        * @param name
        *            RoundRobin()
        * @param override
        *            None
        * @param comment
        *            Inside the RoundRobin() method, a while loop exists to check
        *            if our “ReadyQueue”, “ProcessesList”, or the “CPU” is not empty.
        * @param visiblity
        *            public
        *
        */
     public void RoundRobin()
     {
         Timer = 0;
         ContextSwitch = 0;
         CPU = null;
         ReadyQueue = new ArrayList<>();
         EndProcesses = new ArrayList<>();

         while(!ReadyQueue.isEmpty() || !ProcessesList.isEmpty() || CPU != null)
         {
             //add to ReadyQueue
             for(int i = 0; i < ProcessesList.size(); i++)
             {
                 if(ProcessesList.get(i).ArrivalTime == Timer)
                 {
                     ReadyQueue.add(ProcessesList.remove(i));
                 }
             }

             // Add to CPU
             if (CPU == null)
             {
                 CPU = ReadyQueue.remove(0);
             }

             Counter++;
             CPU.ServiceTime++;

             if(CPU.BurstTime == CPU.ServiceTime)
             {
                 // done
                 CPU.CompletionTime = Timer; //Completion Time is set
                 EndProcesses.add(CPU);
                 CPU = null;
                 ContextSwitch++;
                 Counter = 0;
             }
             else if(Counter == TimeQuantum)
             {
                 //Exceeds TimeQuantum
                 ReadyQueue.add(CPU);
                 CPU = null;
                 ContextSwitch++;
                 Counter = 0;
             }
             Timer++; // Real Time

             double SumTurnAroundTime = 0.0;
             double SumWaitingTime = 0.0;
             double SumUtil = 0.0;
             for(int j = 0; j < EndProcesses.size(); j++)
             {
                 SumTurnAroundTime += EndProcesses.get(j).CompletionTime - 
 EndProcesses.get(j).ArrivalTime;
                 SumWaitingTime += (EndProcesses.get(j).CompletionTime - 
 EndProcesses.get(j).ArrivalTime)
                         - EndProcesses.get(j).BurstTime;
                 SumUtil += EndProcesses.get(j).BurstTime;
             }

             double AvgTurnAroundTime = SumTurnAroundTime / EndProcesses.size();
             double AvgWaitingTime = SumWaitingTime / EndProcesses.size();
             double CPUUtilization = (SumUtil - (ContextSwitch * 0.01)) / Timer;
             double Throughput = (double) EndProcesses.size() / Timer;

             System.out.println("");
             System.out.println("CPU Utilization: " + CPUUtilization);
             System.out.println("Throughput: " + Throughput);
             System.out.println("Average Waiting Time: " + AvgWaitingTime);
             System.out.println("Average Turnaround Time: " + AvgTurnAroundTime);

         }


     }

 }
```

Output Results:
Please Enter Filename: TestCSV.csv
Please Enter Time Quantum: 1

```
CPU Utilization: -0.01
Throughput: 0.0
Average Waiting Time: NaN
Average Turnaround Time: NaN

CPU Utilization: -0.01
Throughput: 0.0
Average Waiting Time: NaN
Average Turnaround Time: NaN

CPU Utilization: -0.01
Throughput: 0.0
Average Waiting Time: NaN
Average Turnaround Time: NaN

CPU Utilization: -0.01
Throughput: 0.0
Average Waiting Time: NaN
Average Turnaround Time: NaN

CPU Utilization: 0.39
Throughput: 0.2
Average Waiting Time: 2.0
Average Turnaround Time: 4.0

CPU Utilization: 0.3233333333333333
Throughput: 0.16666666666666666
Average Waiting Time: 2.0
Average Turnaround Time: 4.0

CPU Utilization: 0.2757142857142857
Throughput: 0.14285714285714285
Average Waiting Time: 2.0
Average Turnaround Time: 4.0

CPU Utilization: 0.24
Throughput: 0.125
Average Waiting Time: 2.0
Average Turnaround Time: 4.0
```

```
CPU Utilization: 0.2122222222222222
Throughput: 0.1111111111111111
Average Waiting Time: 2.0
Average Turnaround Time: 4.0

CPU Utilization: 0.19
Throughput: 0.1
Average Waiting Time: 2.0
Average Turnaround Time: 4.0

CPU Utilization: 0.17181818181818181
Throughput: 0.09090909090909091
Average Waiting Time: 2.0
Average Turnaround Time: 4.0

CPU Utilization: 0.15666666666666665
Throughput: 0.08333333333333333
Average Waiting Time: 2.0
Average Turnaround Time: 4.0

CPU Utilization: 0.5284615384615384
Throughput: 0.15384615384615385
Average Waiting Time: 4.5
Average Turnaround Time: 8.0

CPU Utilization: 0.49000000000000005
Throughput: 0.14285714285714285
Average Waiting Time: 4.5
Average Turnaround Time: 8.0

CPU Utilization: 0.45666666666666667
Throughput: 0.13333333333333333
Average Waiting Time: 4.5
Average Turnaround Time: 8.0

CPU Utilization: 0.4275
Throughput: 0.125
Average Waiting Time: 4.5
Average Turnaround Time: 8.0

CPU Utilization: 0.40176470588235297
Throughput: 0.11764705882352941
Average Waiting Time: 4.5
Average Turnaround Time: 8.0
```

```
CPU Utilization: 0.3788888888888889
Throughput: 0.1111111111111111
Average Waiting Time: 4.5
Average Turnaround Time: 8.0

CPU Utilization: 0.6742105263157895
Throughput: 0.15789473684210525
Average Waiting Time: 6.333333333333333
Average Turnaround Time: 10.666666666666666

CPU Utilization: 0.99
Throughput: 0.2
Average Waiting Time: 7.5
Average Turnaround Time: 12.5
```

Output Results:

```
Please Enter Filename: TestCSV.csv
Please Enter Time Quantum: 10

CPU Utilization: 0.0
Throughput: 0.0
Average Waiting Time: NaN
Average Turnaround Time: NaN

CPU Utilization: 0.0
Throughput: 0.0
Average Waiting Time: NaN
Average Turnaround Time: NaN

CPU Utilization: 0.0
Throughput: 0.0
Average Waiting Time: NaN
Average Turnaround Time: NaN

CPU Utilization: 0.0
Throughput: 0.0
Average Waiting Time: NaN
Average Turnaround Time: NaN

CPU Utilization: 0.998
Throughput: 0.2
Average Waiting Time: -1.0
Average Turnaround Time: 4.0

CPU Utilization: 0.8316666666666667
Throughput: 0.16666666666666666
Average Waiting Time: -1.0
Average Turnaround Time: 4.0

CPU Utilization: 0.9971428571428572
Throughput: 0.2857142857142857
Average Waiting Time: 1.5
Average Turnaround Time: 5.0

CPU Utilization: 0.8725
Throughput: 0.25
Average Waiting Time: 1.5
Average Turnaround Time: 5.0
```

```
CPU Utilization: 0.7755555555555556
Throughput: 0.2222222222222222
Average Waiting Time: 1.5
Average Turnaround Time: 5.0

CPU Utilization: 0.6980000000000001
Throughput: 0.2
Average Waiting Time: 1.5
Average Turnaround Time: 5.0


CPU Utilization: 0.6345454545454546
Throughput: 0.18181818181818182
Average Waiting Time: 1.5
Average Turnaround Time: 5.0

CPU Utilization: 0.5816666666666667
Throughput: 0.16666666666666666
Average Waiting Time: 1.5
Average Turnaround Time: 5.0

CPU Utilization: 0.536923076923077
Throughput: 0.15384615384615385
Average Waiting Time: 1.5
Average Turnaround Time: 5.0

CPU Utilization: 0.9978571428571429
Throughput: 0.21428571428571427
Average Waiting Time: 2.6666666666666665
Average Turnaround Time: 7.333333333333333

CPU Utilization: 0.9313333333333333
Throughput: 0.2
Average Waiting Time: 2.6666666666666665
Average Turnaround Time: 7.333333333333333

CPU Utilization: 0.873125
Throughput: 0.1875
Average Waiting Time: 2.6666666666666665
Average Turnaround Time: 7.333333333333333

CPU Utilization: 0.821764705882353
Throughput: 0.17647058823529413
Average Waiting Time: 2.6666666666666665
Average Turnaround Time: 7.333333333333333
```

```
CPU Utilization: 0.7761111111111112
Throughput: 0.16666666666666666
Average Waiting Time: 2.6666666666666665
Average Turnaround Time: 7.333333333333333

CPU Utilization: 0.7352631578947368
Throughput: 0.15789473684210525
Average Waiting Time: 2.6666666666666665
Average Turnaround Time: 7.333333333333333

CPU Utilization: 0.998
Throughput: 0.2
Average Waiting Time: 4.75
Average Turnaround Time: 9.75
```

Output Results:

Please Enter Filename: TestCSV.csv
Please Enter Time Quantum: 29

```
CPU Utilization: 0.0
Throughput: 0.0
Average Waiting Time: NaN
Average Turnaround Time: NaN

CPU Utilization: 0.0
Throughput: 0.0
Average Waiting Time: NaN
Average Turnaround Time: NaN

CPU Utilization: 0.0
Throughput: 0.0
Average Waiting Time: NaN
Average Turnaround Time: NaN

CPU Utilization: 0.0
Throughput: 0.0
Average Waiting Time: NaN
Average Turnaround Time: NaN

CPU Utilization: 0.998
Throughput: 0.2
Average Waiting Time: -1.0
Average Turnaround Time: 4.0

CPU Utilization: 0.8316666666666667
Throughput: 0.16666666666666666
Average Waiting Time: -1.0
Average Turnaround Time: 4.0

CPU Utilization: 0.9971428571428572
Throughput: 0.2857142857142857
Average Waiting Time: 1.5
Average Turnaround Time: 5.0

CPU Utilization: 0.8725
Throughput: 0.25
Average Waiting Time: 1.5
Average Turnaround Time: 5.0
```

```
CPU Utilization: 0.7755555555555556
Throughput: 0.2222222222222222
Average Waiting Time: 1.5
Average Turnaround Time: 5.0

CPU Utilization: 0.6980000000000001
Throughput: 0.2
Average Waiting Time: 1.5
Average Turnaround Time: 5.0

CPU Utilization: 0.6345454545454546
Throughput: 0.18181818181818182
Average Waiting Time: 1.5
Average Turnaround Time: 5.0

CPU Utilization: 0.5816666666666667
Throughput: 0.16666666666666666
Average Waiting Time: 1.5
Average Turnaround Time: 5.0

CPU Utilization: 0.536923076923077
Throughput: 0.15384615384615385
Average Waiting Time: 1.5
Average Turnaround Time: 5.0

CPU Utilization: 0.9978571428571429
Throughput: 0.21428571428571427
Average Waiting Time: 2.6666666666666665
Average Turnaround Time: 7.333333333333333

CPU Utilization: 0.9313333333333333
Throughput: 0.2
Average Waiting Time: 2.6666666666666665
Average Turnaround Time: 7.333333333333333

CPU Utilization: 0.873125
Throughput: 0.1875
Average Waiting Time: 2.6666666666666665
Average Turnaround Time: 7.333333333333333

CPU Utilization: 0.821764705882353
Throughput: 0.17647058823529413
Average Waiting Time: 2.6666666666666665
Average Turnaround Time: 7.333333333333333
```

```
CPU Utilization: 0.7761111111111112
Throughput: 0.16666666666666666
Average Waiting Time: 2.6666666666666665
Average Turnaround Time: 7.333333333333333

CPU Utilization: 0.7352631578947368
Throughput: 0.15789473684210525
Average Waiting Time: 2.6666666666666665
Average Turnaround Time: 7.333333333333333

CPU Utilization: 0.998
Throughput: 0.2
Average Waiting Time: 4.75
Average Turnaround Time: 9.75
```

Output Results:

```
Please Enter Filename: TestCSV.csv
Please Enter Time Quantum: 77
```

```
CPU Utilization: 0.0
Throughput: 0.0
Average Waiting Time: NaN
Average Turnaround Time: NaN

CPU Utilization: 0.0
Throughput: 0.0
Average Waiting Time: NaN
Average Turnaround Time: NaN

CPU Utilization: 0.0
Throughput: 0.0
Average Waiting Time: NaN
Average Turnaround Time: NaN

CPU Utilization: 0.0
Throughput: 0.0
Average Waiting Time: NaN
Average Turnaround Time: NaN

CPU Utilization: 0.998
Throughput: 0.2
Average Waiting Time: -1.0
Average Turnaround Time: 4.0

CPU Utilization: 0.8316666666666667
Throughput: 0.16666666666666666
Average Waiting Time: -1.0
Average Turnaround Time: 4.0

CPU Utilization: 0.9971428571428572
Throughput: 0.2857142857142857
Average Waiting Time: 1.5
Average Turnaround Time: 5.0

CPU Utilization: 0.8725
Throughput: 0.25
Average Waiting Time: 1.5
Average Turnaround Time: 5.0
```

```
CPU Utilization: 0.7755555555555556
Throughput: 0.2222222222222222
Average Waiting Time: 1.5
Average Turnaround Time: 5.0

CPU Utilization: 0.6980000000000001
Throughput: 0.2
Average Waiting Time: 1.5
Average Turnaround Time: 5.0

CPU Utilization: 0.6345454545454546
Throughput: 0.18181818181818182
Average Waiting Time: 1.5
Average Turnaround Time: 5.0

CPU Utilization: 0.5816666666666667
Throughput: 0.16666666666666666
Average Waiting Time: 1.5
Average Turnaround Time: 5.0

CPU Utilization: 0.536923076923077
Throughput: 0.15384615384615385
Average Waiting Time: 1.5
Average Turnaround Time: 5.0

CPU Utilization: 0.9978571428571429
Throughput: 0.21428571428571427
Average Waiting Time: 2.6666666666666665
Average Turnaround Time: 7.333333333333333

CPU Utilization: 0.9313333333333333
Throughput: 0.2
Average Waiting Time: 2.6666666666666665
Average Turnaround Time: 7.333333333333333

CPU Utilization: 0.873125
Throughput: 0.1875
Average Waiting Time: 2.6666666666666665
Average Turnaround Time: 7.333333333333333

CPU Utilization: 0.821764705882353
Throughput: 0.17647058823529413
Average Waiting Time: 2.6666666666666665
Average Turnaround Time: 7.333333333333333
```

```
CPU Utilization: 0.7761111111111112
Throughput: 0.16666666666666666
Average Waiting Time: 2.6666666666666665
Average Turnaround Time: 7.333333333333333

CPU Utilization: 0.7352631578947368
Throughput: 0.15789473684210525
Average Waiting Time: 2.6666666666666665
Average Turnaround Time: 7.333333333333333

CPU Utilization: 0.998
Throughput: 0.2
Average Waiting Time: 4.75
Average Turnaround Time: 9.75
```

Output Results:

```
Please Enter Filename: TestCSV.csv
Please Enter Time Quantum: 1223

CPU Utilization: 0.0
Throughput: 0.0
Average Waiting Time: NaN
Average Turnaround Time: NaN

CPU Utilization: 0.0
Throughput: 0.0
Average Waiting Time: NaN
Average Turnaround Time: NaN

CPU Utilization: 0.0
Throughput: 0.0
Average Waiting Time: NaN
Average Turnaround Time: NaN

CPU Utilization: 0.0
Throughput: 0.0
Average Waiting Time: NaN
Average Turnaround Time: NaN

CPU Utilization: 0.998
Throughput: 0.2
Average Waiting Time: -1.0
Average Turnaround Time: 4.0

CPU Utilization: 0.8316666666666667
Throughput: 0.16666666666666666
Average Waiting Time: -1.0
Average Turnaround Time: 4.0

CPU Utilization: 0.9971428571428572
Throughput: 0.2857142857142857
Average Waiting Time: 1.5
Average Turnaround Time: 5.0

CPU Utilization: 0.8725
Throughput: 0.25
Average Waiting Time: 1.5
Average Turnaround Time: 5.0
```

```
CPU Utilization: 0.7755555555555556
Throughput: 0.2222222222222222
Average Waiting Time: 1.5
Average Turnaround Time: 5.0

CPU Utilization: 0.6980000000000001
Throughput: 0.2
Average Waiting Time: 1.5
Average Turnaround Time: 5.0

CPU Utilization: 0.6345454545454546
Throughput: 0.18181818181818182
Average Waiting Time: 1.5
Average Turnaround Time: 5.0

CPU Utilization: 0.5816666666666667
Throughput: 0.16666666666666666
Average Waiting Time: 1.5
Average Turnaround Time: 5.0

CPU Utilization: 0.536923076923077
Throughput: 0.15384615384615385
Average Waiting Time: 1.5
Average Turnaround Time: 5.0

CPU Utilization: 0.9978571428571429
Throughput: 0.21428571428571427
Average Waiting Time: 2.6666666666666665
Average Turnaround Time: 7.333333333333333

CPU Utilization: 0.9313333333333333
Throughput: 0.2
Average Waiting Time: 2.6666666666666665
Average Turnaround Time: 7.333333333333333

CPU Utilization: 0.873125
Throughput: 0.1875
Average Waiting Time: 2.6666666666666665
Average Turnaround Time: 7.333333333333333

CPU Utilization: 0.821764705882353
Throughput: 0.17647058823529413
Average Waiting Time: 2.6666666666666665
Average Turnaround Time: 7.333333333333333
```

```
CPU Utilization: 0.7761111111111112
Throughput: 0.16666666666666666
Average Waiting Time: 2.6666666666666665
Average Turnaround Time: 7.333333333333333

CPU Utilization: 0.7352631578947368
Throughput: 0.15789473684210525
Average Waiting Time: 2.6666666666666665
Average Turnaround Time: 7.333333333333333

CPU Utilization: 0.998
Throughput: 0.2
Average Waiting Time: 4.75
Average Turnaround Time: 9.75
```