

By: Jcook03266

<https://github.com/jcook03266>

Round Robin CPU Scheduling Algorithm Implemented in Java

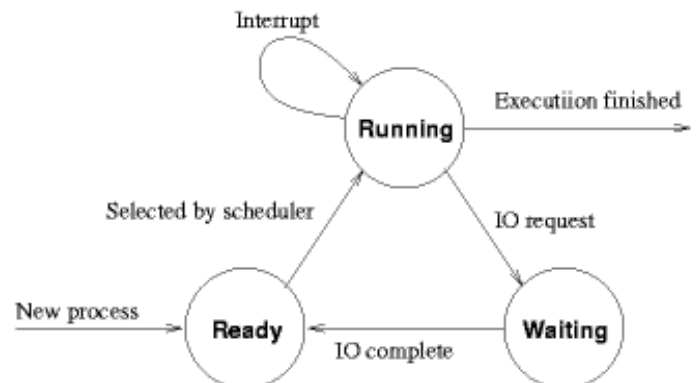
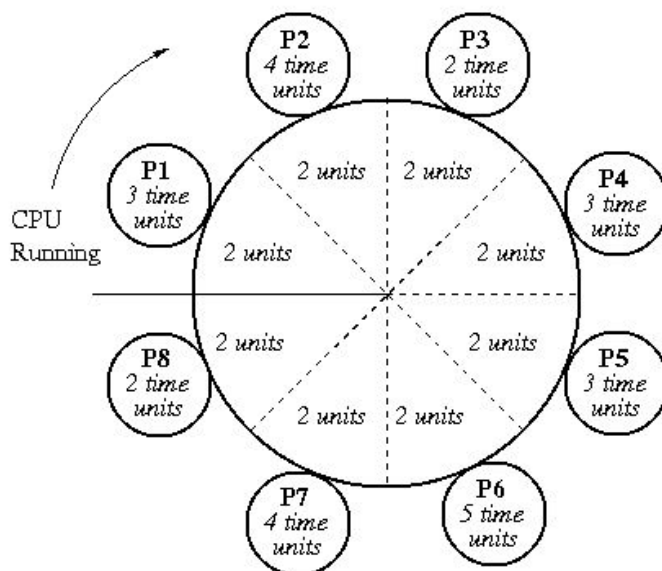
FYI: Page 5 is where the instructions are on how to run the jar file, and the last page is analysis

/////////////////What is CPU Scheduling?/////////////////

CPU scheduling at the fundamental level is a method of assigning work to specific resources that will then complete said work, i.e cores on a cpu. This work can be entirely virtual implementations of computational instances such as processes and threads and data flows of some sort. These virtual implementations are then scheduled onto hardware resources where they're then handled via the appropriate execution methods, these hardware resources can be comprised of processor cores much like I mentioned earlier, expansion cards i.e graphics cards and even network links.

/////////////////Why do we need CPU Scheduling?/////////////////

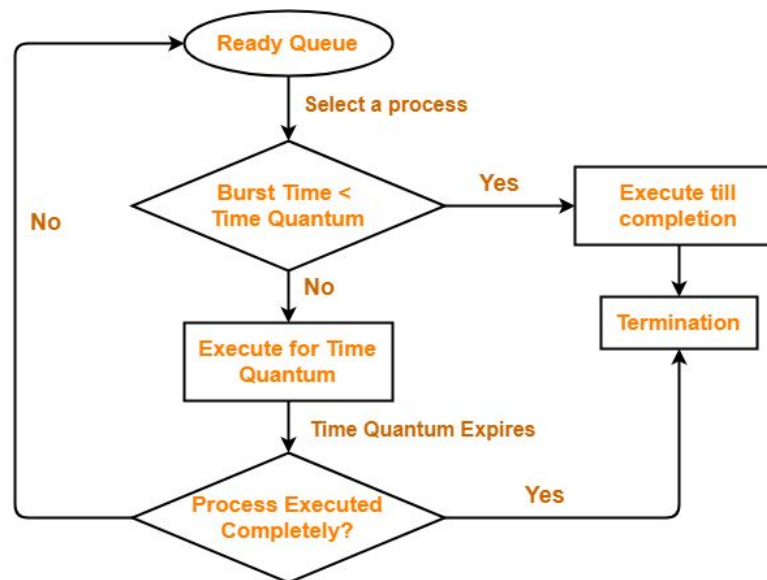
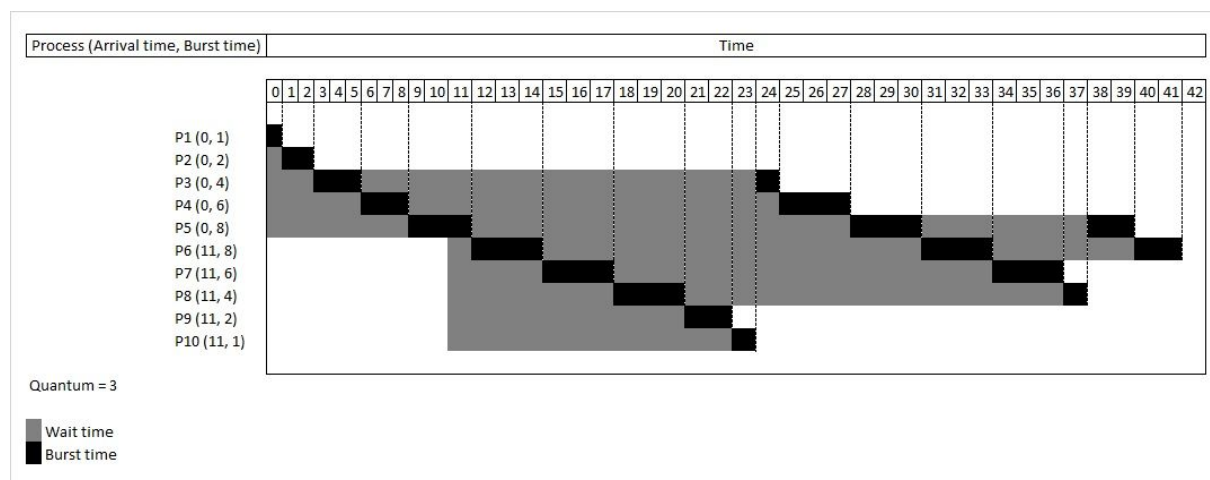
We use cpu scheduling for mainly efficiency purposes, we schedule tasks to be handled later on down the line in order to prevent cpu from idling and wasting precious cycles it could be using for doing essential work. This efficiency is then used to make the system faster and more responsive as it can handle more by doing less, hence scheduling, much like a form of procrastination but necessary procrastination. The operating system has to select one of the process scheduled in the ready queue to be executed. The scheduler selects from an assortment of processes in memory that are ready to be executed and allocated the cpu to one of them. Now the allocation process is up to the scheduling algorithm, this is where Round Robin and many other algorithms come into play.



By: Jcook03266
<https://github.com/jcook03266>
 Round Robin CPU Scheduling Algorithm Implemented in Java

//////////What is the Round Robin Scheduling algorithm and how does it work?//////////

The Round Robin Scheduling algorithm is a scheduling algorithm utilized by process and network schedulers in computing. In this algorithm there are time slices aka time quantum that are equally cut up and assigned to each process in a circular order. These processes are handled in a circular order meaning one after the other and one before another, without priority designation to certain processes unlike other algorithms which we won't discuss here, this process is known as cyclic executive, meaning there is only one unified task, not many others with different priorities over one another. Round robin is a rather simple scheduling algorithm and is easy to implement with it being void of any risk of starvation, with starvation referring to resource starvation, starvation is a problem encountered in concurrent computing where a process is perpetually (endlessly) denied the necessary resources to process its work. Think of resource starvation like your printer refusing to print out the test page you need for your final exam, it can be pretty disastrous.



By: Jcook03266
<https://github.com/jcook03266>
Round Robin CPU Scheduling Algorithm Implemented in Java

//////////////////How was this algorithm simulated in Java?//////////////////

In Java, you implement Round Robin easily with the many libraries available. In terms of hard coding the functionality you break the scheduling algorithm into 5 parts:

<Look at source code for the appropriate methods and variables and java libraries used in each class>

1.) CPU Class-

In this class we execute the processes and see if their remaining burst times are less than, greater than or equal to the time quantum value, if any, we update the remaining burst time, mark the processes as completed with the counter and return the time quantum value or the remaining burst time depending on the condition.

2.) Clock Class-

Clock Class, In this class we update the time elapsed since the start of the simulation, we update the arrival time of a process waiting to get into the ready queue, as well as the wait time of a process already in the ready queue by using an iterator and then setting the wait time of the current cpu.

3.) Process Class-

Process class enables the creation of our process objects which use like trading cards with the other classes, with clock and cpu enabling the tracking of these processes and the statistics attributed to them via later analysis performed in the execution class. All of the process objects start off here.

4.) Execution Class-

This is where all of our process objects go to get executed and analyzed while being executed by the round robin scheduling algorithm. We basically input a process into the scheduling algorithm where we track whether or not it has been completed or not, its position in the ready queue, whether it's ready to go into the ready queue or come out of it to be executed, and if it has to go back into the queue and popped out from the head of the queue once more. This class is pretty self explanatory, it executes the processes using RR.

5.) Main Class-

By: Jcook03266
<https://github.com/jcook03266>
Round Robin CPU Scheduling Algorithm Implemented in Java

This is where the processes are parsed from the input file and placed into arrays with their proper information in order to be turned into process objects that retain all of this necessary information for later.

tracking application and analysis. This class accepts the two fundamental inputs which are the time quantum and the input file that is parsed.

These are the fundamental parts of the basic algorithm implementation.

To further simplify simulating and debugging this scheduling algorithm we can design a basic UI to control our inputs, launch the simulation, and even save our session's data and quit at any time with the ability to restart the simulation after one has already been completed.

This is the 6th Class that bridges all of the others together

6.) GUI Class-

Basic GUI that allows the user to interact with the round robin algorithm with simplicity. The user simply provides the path directory to the csv format file that's required by this implementation of the scheduler via typing it directly, or by importing it, and gives the time quantum value by sliding the slider from 1 - 15 at an interval of 1. The results of the algorithm are printed out to the custom 'Event log' console located on the right half of the JFrame, with these results being able to be saved to a text file named 'RRSimEventLog.txt' located on your desktop by default. You can restart the simulation as many times as you'd like and provide multiple different time quantum values for each run. Note: For the csv file input, you can simply remove the placeholder text around Resources/data.txt and use only that no quotations or any other characters, as your input path as the default csv file is located via that path.

<Look at source code for the appropriate methods and variables and java libraries used in each class>

-Note: In this implementation of the algorithm we use CSV formatted files to parse the process information and feed it to our backend, comma separated value formatted .txt files are simple and efficient ways of going about this process without overcomplicating things and enabling more room for error. We can even use headers in this format to further organize our data even more.

Now that we've covered the basics of how to implement Round Robin in Java, we can now address how to run this implementation:

By: Jcook03266
<https://github.com/jcook03266>
Round Robin CPU Scheduling Algorithm Implemented in Java

- 1.) Click on the Round Robin.jar file to execute the program.
- 2.) Input the path the csv file you wish to use, by default this program comes with a default CSV file already linked,
just copy the Resources/data.txt from the placeholder text without quotations, press the clear button directly below the input bar and paste it back into the input bar. Or you can just copy the path of the file you want to use, or just click import and then browse your system for the file which is the recommended method of going about this.
- 3.) Press the validate button below the input bar to validate that the file is indeed linked correctly and that is in the CSV format.
- 4.) After linking the file and validating it we now go to the time quantum value input section and simply drag the slider to whichever value we'd like, by default the value is set to 15 which is the middle value of the slider. It's recommended you use lower values below '10' for better results, anything above that results in uncertain analysis depending on the data input.
- 5.) After you set the time quantum value you simply press start simulation and sit back while everything is displayed in the event log on the right
- 6.) Once the simulation is finished you have the options of: saving the contents of the event log to a file named 'RRSimEventLog.txt' that will be automatically saved to your desktop, exiting the program, clearing the event log, and or restarting the simulation using different criteria or the same criteria depending on what you want, it's up to you.

And this concludes the tutorial on how to run and simulate the Round Robin CPU scheduling algorithm implemented in Java

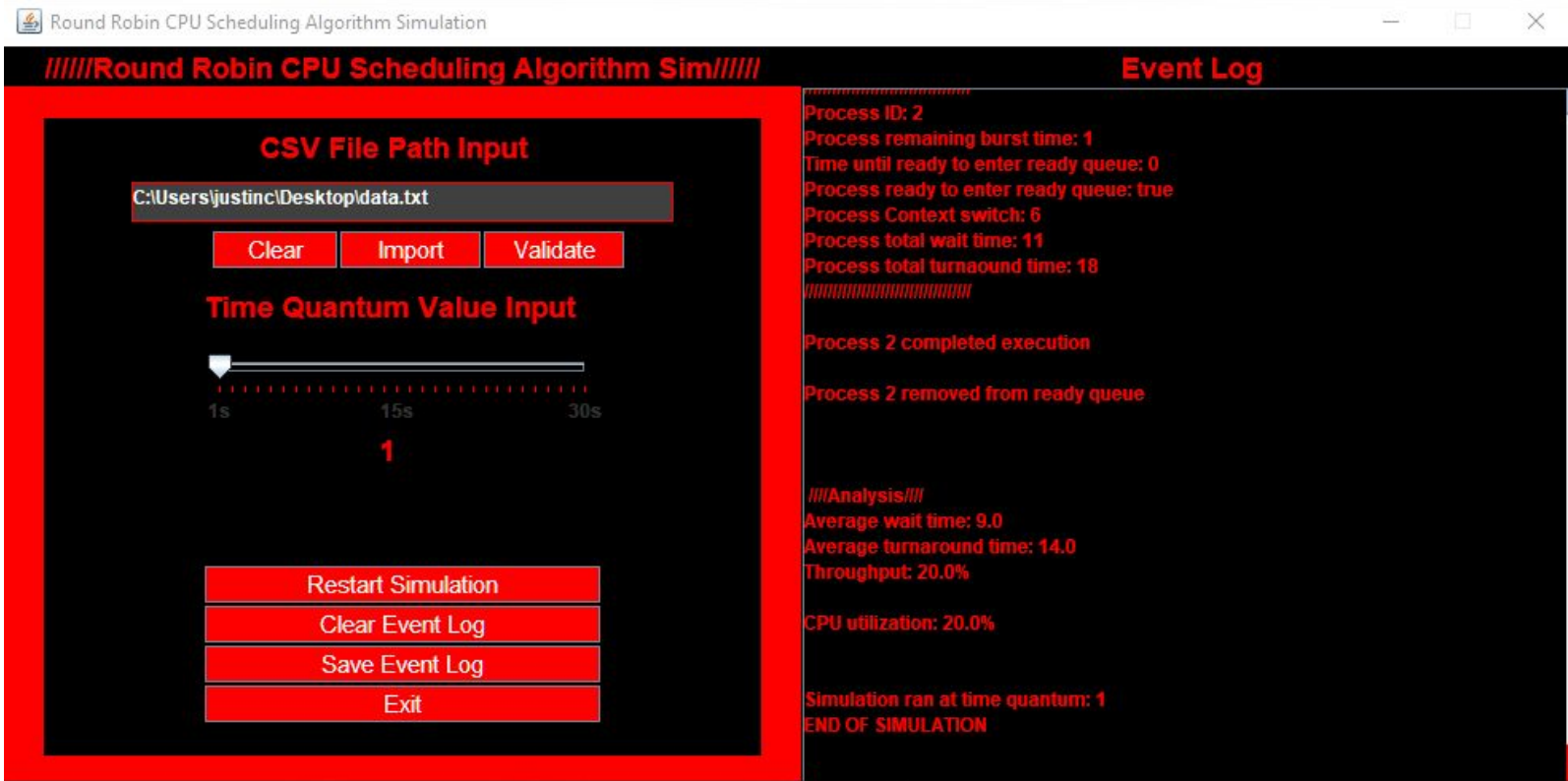
And this also concludes the minor walkthrough on cpu scheduling and Round Robin, if you're interested in this topic even more I recommend exploring more operating systems based ideas and looking at other scheduling algorithms

The next page contains visual examples of this program in action, enjoy.

Examples of this program running on 5 different time quanta:

By: Jcook03266
<https://github.com/jcook03266>
Round Robin CPU Scheduling Algorithm Implemented in Java

Time Quantum $t = 1$



Round Robin CPU Scheduling Algorithm Simulation

CSV File Path Input

C:\Users\justinc\Desktop\data.txt

Clear Import Validate

Time Quantum Value Input

1s 15s 30s

1

Restart Simulation

Clear Event Log

Save Event Log

Exit

Event Log

Process ID: 2

Process remaining burst time: 1

Time until ready to enter ready queue: 0

Process ready to enter ready queue: true

Process Context switch: 6

Process total wait time: 11

Process total turnaround time: 18

Process 2 completed execution

Process 2 removed from ready queue

///Analysis///

Average wait time: 9.0

Average turnaround time: 14.0

Throughput: 20.0%

CPU utilization: 20.0%

Simulation ran at time quantum: 1

END OF SIMULATION

Readout from the Event Log:

//////////Validating CSV File//////////
//////////Parsing data from CSV File//////////

Parsed Group 1:

Process ID: 1
Arrival Time: 0
Burst time: 5

Parsed Group 2:

Process ID: 2
Arrival Time: 1
Burst time: 7

Parsed Group 3:

Process ID: 3
Arrival Time: 0
Burst time: 2

By: Jcook03266
<https://github.com/jcook03266>
Round Robin CPU Scheduling Algorithm Implemented in Java

Parsed Group 4:

Process ID: 4

Arrival Time: 2

Burst time: 6

/////////////////Validating CSV File////////////////

/////////////////Parsing data from CSV File////////////////

Parsed Group 1:

Process ID: 1

Arrival Time: 0

Burst time: 5

Parsed Group 2:

Process ID: 2

Arrival Time: 1

Burst time: 7

Parsed Group 3:

Process ID: 3

Arrival Time: 0

Burst time: 2

Parsed Group 4:

Process ID: 4

Arrival Time: 2

Burst time: 6

New Process: 1 Added to Ready Queue

New Process: 3 Added to Ready Queue

New Process: 2 Added to Ready Queue

New Process: 4 Added to Ready Queue

////////////////////////////////////

Process ID: 3

Process remaining burst time: 2

Time until ready to enter ready queue: 0

Process ready to enter ready queue: true

Process Context switch: 0

Process total wait time: 0

Process total turnaround time: 2

By: Jcook03266
<https://github.com/jcook03266>
Round Robin CPU Scheduling Algorithm Implemented in Java

////////////////////////////////////

////////////////////////////////////

Process ID: 3
Process remaining burst time: 1
Time until ready to enter ready queue: 0
Process ready to enter ready queue: true
Process Context switch: 1
Process total wait time: 1
Process total turnaound time: 3

////////////////////////////////////

Process 3 completed execution

Process 3 removed from ready queue

////////////////////////////////////

Process ID: 1
Process remaining burst time: 5
Time until ready to enter ready queue: 0
Process ready to enter ready queue: true
Process Context switch: 0
Process total wait time: 0
Process total turnaound time: 5

////////////////////////////////////

////////////////////////////////////

Process ID: 1
Process remaining burst time: 4
Time until ready to enter ready queue: 0
Process ready to enter ready queue: true
Process Context switch: 1
Process total wait time: 0
Process total turnaound time: 5

////////////////////////////////////

////////////////////////////////////

Process ID: 1
Process remaining burst time: 3
Time until ready to enter ready queue: 0
Process ready to enter ready queue: true
Process Context switch: 2
Process total wait time: 2
Process total turnaound time: 7

////////////////////////////////////

////////////////////////////////////

Process ID: 1
Process remaining burst time: 2
Time until ready to enter ready queue: 0
Process ready to enter ready queue: true

By: Jcook03266
<https://github.com/jcook03266>
Round Robin CPU Scheduling Algorithm Implemented in Java

Process Context switch: 3
Process total wait time: 5
Process total turnaound time: 10
////////////////////////////////////

////////////////////////////////////
Process ID: 1
Process remaining burst time: 1
Time until ready to enter ready queue: 0
Process ready to enter ready queue: true
Process Context switch: 4
Process total wait time: 7
Process total turnaound time: 12
////////////////////////////////////

Process 1 completed execution

Process 1 removed from ready queue

////////////////////////////////////
Process ID: 4
Process remaining burst time: 6
Time until ready to enter ready queue: 0
Process ready to enter ready queue: true
Process Context switch: 0
Process total wait time: 0
Process total turnaound time: 6
////////////////////////////////////

////////////////////////////////////
Process ID: 4
Process remaining burst time: 5
Time until ready to enter ready queue: 0
Process ready to enter ready queue: true
Process Context switch: 1
Process total wait time: 2
Process total turnaound time: 8
////////////////////////////////////

////////////////////////////////////
Process ID: 4
Process remaining burst time: 4
Time until ready to enter ready queue: 0
Process ready to enter ready queue: true
Process Context switch: 2
Process total wait time: 5
Process total turnaound time: 11
////////////////////////////////////

////////////////////////////////////
Process ID: 4

By: Jcook03266
<https://github.com/jcook03266>
Round Robin CPU Scheduling Algorithm Implemented in Java

Process remaining burst time: 3
Time until ready to enter ready queue: 0
Process ready to enter ready queue: true
Process Context switch: 3
Process total wait time: 7
Process total turnaround time: 13
////////////////////////////////

////////////////////////////////
Process ID: 4
Process remaining burst time: 2
Time until ready to enter ready queue: 0
Process ready to enter ready queue: true
Process Context switch: 4
Process total wait time: 9
Process total turnaround time: 15
////////////////////////////////

////////////////////////////////
Process ID: 4
Process remaining burst time: 1
Time until ready to enter ready queue: 0
Process ready to enter ready queue: true
Process Context switch: 5
Process total wait time: 10
Process total turnaround time: 16
////////////////////////////////

Process 4 completed execution

Process 4 removed from ready queue

////////////////////////////////
Process ID: 2
Process remaining burst time: 7
Time until ready to enter ready queue: 0
Process ready to enter ready queue: true
Process Context switch: 0
Process total wait time: 0
Process total turnaround time: 7
////////////////////////////////

////////////////////////////////
Process ID: 2
Process remaining burst time: 6
Time until ready to enter ready queue: 0
Process ready to enter ready queue: true
Process Context switch: 1
Process total wait time: 1
Process total turnaround time: 8
////////////////////////////////

By: Jcook03266
<https://github.com/jcook03266>
Round Robin CPU Scheduling Algorithm Implemented in Java

////////////////////////////////////

Process ID: 2
Process remaining burst time: 5
Time until ready to enter ready queue: 0
Process ready to enter ready queue: true
Process Context switch: 2
Process total wait time: 4
Process total turnaound time: 11

////////////////////////////////////

////////////////////////////////////

Process ID: 2
Process remaining burst time: 4
Time until ready to enter ready queue: 0
Process ready to enter ready queue: true
Process Context switch: 3
Process total wait time: 6
Process total turnaound time: 13

////////////////////////////////////

////////////////////////////////////

Process ID: 2
Process remaining burst time: 3
Time until ready to enter ready queue: 0
Process ready to enter ready queue: true
Process Context switch: 4
Process total wait time: 8
Process total turnaound time: 15

////////////////////////////////////

////////////////////////////////////

Process ID: 2
Process remaining burst time: 2
Time until ready to enter ready queue: 0
Process ready to enter ready queue: true
Process Context switch: 5
Process total wait time: 10
Process total turnaound time: 17

////////////////////////////////////

////////////////////////////////////

Process ID: 2
Process remaining burst time: 1
Time until ready to enter ready queue: 0
Process ready to enter ready queue: true
Process Context switch: 6
Process total wait time: 11
Process total turnaound time: 18

////////////////////////////////////

By: Jcook03266
<https://github.com/jcook03266>
Round Robin CPU Scheduling Algorithm Implemented in Java

Process 2 completed execution

Process 2 removed from ready queue

////Analysis////

Average wait time: 9.0

Average turnaround time: 14.0

Throughput: 20.0%

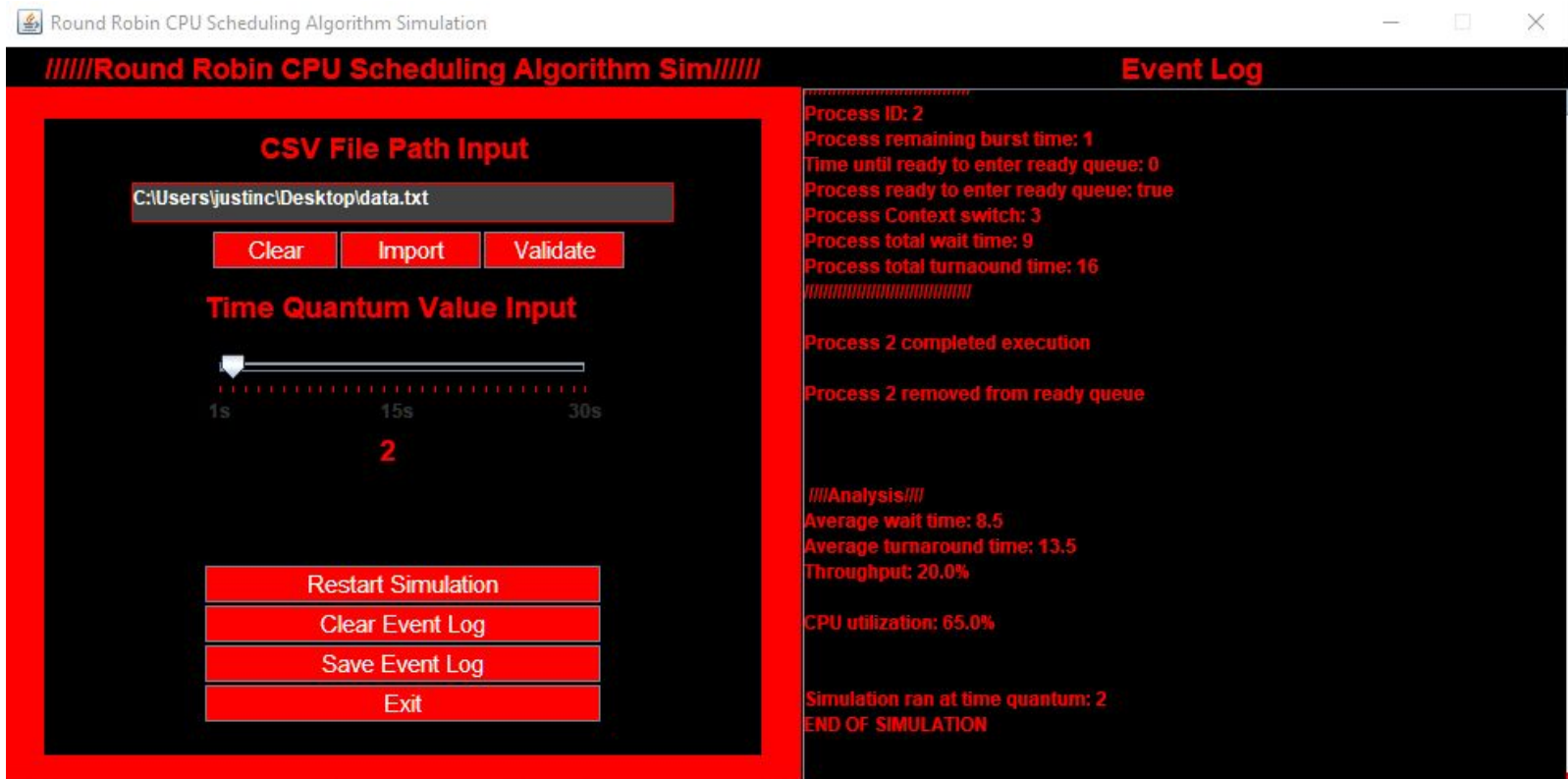
CPU utilization: 20.0%

Simulation ran at time quantum: 1

END OF SIMULATION

By: Jcook03266
<https://github.com/jcook03266>
Round Robin CPU Scheduling Algorithm Implemented in Java

Time Quantum $t = 2$



The screenshot shows a Java application window titled "Round Robin CPU Scheduling Algorithm Simulation". The interface is divided into two main panels. The left panel, titled "//////Round Robin CPU Scheduling Algorithm Sim//////", contains a "CSV File Path Input" section with a text field showing "C:\Users\justinc\Desktop\data.txt" and buttons for "Clear", "Import", and "Validate". Below this is a "Time Quantum Value Input" section with a slider ranging from 1s to 30s, currently set at 2, and buttons for "Restart Simulation", "Clear Event Log", "Save Event Log", and "Exit". The right panel, titled "Event Log", displays the following text: "Process ID: 2", "Process remaining burst time: 1", "Time until ready to enter ready queue: 0", "Process ready to enter ready queue: true", "Process Context switch: 3", "Process total wait time: 9", "Process total turnaround time: 16", followed by a separator line, "Process 2 completed execution", "Process 2 removed from ready queue", another separator line, "////Analysis////", "Average wait time: 8.5", "Average turnaround time: 13.5", "Throughput: 20.0%", "CPU utilization: 65.0%", and finally "Simulation ran at time quantum: 2" and "END OF SIMULATION".

Readout from the Event Log:

//////////Validating CSV File//////////
//////////Parsing data from CSV File//////////

Parsed Group 1:
Process ID: 1
Arrival Time: 0
Burst time: 5

Parsed Group 2:
Process ID: 2
Arrival Time: 1
Burst time: 7

Parsed Group 3:
Process ID: 3
Arrival Time: 0
Burst time: 2

By: Jcook03266
<https://github.com/jcook03266>
Round Robin CPU Scheduling Algorithm Implemented in Java

Parsed Group 4:

Process ID: 4

Arrival Time: 2

Burst time: 6

//////////Validating CSV File//////////

//////////Parsing data from CSV File//////////

Parsed Group 1:

Process ID: 1

Arrival Time: 0

Burst time: 5

Parsed Group 2:

Process ID: 2

Arrival Time: 1

Burst time: 7

Parsed Group 3:

Process ID: 3

Arrival Time: 0

Burst time: 2

Parsed Group 4:

Process ID: 4

Arrival Time: 2

Burst time: 6

New Process: 1 Added to Ready Queue

New Process: 3 Added to Ready Queue

New Process: 2 Added to Ready Queue

New Process: 4 Added to Ready Queue

////////////////////////////////////

Process ID: 3

Process remaining burst time: 2

Time until ready to enter ready queue: 0

Process ready to enter ready queue: true

Process Context switch: 0

Process total wait time: 0

Process total turnaround time: 2

////////////////////////////////////

By: Jcook03266
<https://github.com/jcook03266>
Round Robin CPU Scheduling Algorithm Implemented in Java

Process 3 completed execution

Process 3 removed from ready queue

////////////////////////////////////

Process ID: 1
Process remaining burst time: 5
Time until ready to enter ready queue: 0
Process ready to enter ready queue: true
Process Context switch: 0
Process total wait time: 0
Process total turnaround time: 5

////////////////////////////////////

////////////////////////////////////

Process ID: 1
Process remaining burst time: 3
Time until ready to enter ready queue: 0
Process ready to enter ready queue: true
Process Context switch: 1
Process total wait time: 0
Process total turnaround time: 5

////////////////////////////////////

////////////////////////////////////

Process ID: 1
Process remaining burst time: 1
Time until ready to enter ready queue: 0
Process ready to enter ready queue: true
Process Context switch: 2
Process total wait time: 6
Process total turnaround time: 11

////////////////////////////////////

Process 1 completed execution

Process 1 removed from ready queue

////////////////////////////////////

Process ID: 4
Process remaining burst time: 6
Time until ready to enter ready queue: 0
Process ready to enter ready queue: true
Process Context switch: 0
Process total wait time: 0
Process total turnaround time: 6

////////////////////////////////////

////////////////////////////////////

Process ID: 4

By: Jcook03266
<https://github.com/jcook03266>
Round Robin CPU Scheduling Algorithm Implemented in Java

Process remaining burst time: 4
Time until ready to enter ready queue: 0
Process ready to enter ready queue: true
Process Context switch: 1
Process total wait time: 4
Process total turnaround time: 10
////////////////////////////////

////////////////////////////////
Process ID: 4
Process remaining burst time: 2
Time until ready to enter ready queue: 0
Process ready to enter ready queue: true
Process Context switch: 2
Process total wait time: 8
Process total turnaround time: 14
////////////////////////////////

Process 4 completed execution

Process 4 removed from ready queue

////////////////////////////////
Process ID: 2
Process remaining burst time: 7
Time until ready to enter ready queue: 0
Process ready to enter ready queue: true
Process Context switch: 0
Process total wait time: 0
Process total turnaround time: 7
////////////////////////////////

////////////////////////////////
Process ID: 2
Process remaining burst time: 5
Time until ready to enter ready queue: 0
Process ready to enter ready queue: true
Process Context switch: 1
Process total wait time: 2
Process total turnaround time: 9
////////////////////////////////

////////////////////////////////
Process ID: 2
Process remaining burst time: 3
Time until ready to enter ready queue: 0
Process ready to enter ready queue: true
Process Context switch: 2
Process total wait time: 6
Process total turnaround time: 13
////////////////////////////////

By: Jcook03266
<https://github.com/jcook03266>
Round Robin CPU Scheduling Algorithm Implemented in Java

////////////////////////////////////

Process ID: 2
Process remaining burst time: 1
Time until ready to enter ready queue: 0
Process ready to enter ready queue: true
Process Context switch: 3
Process total wait time: 9
Process total turnaround time: 16

////////////////////////////////////

Process 2 completed execution

Process 2 removed from ready queue

////Analysis////

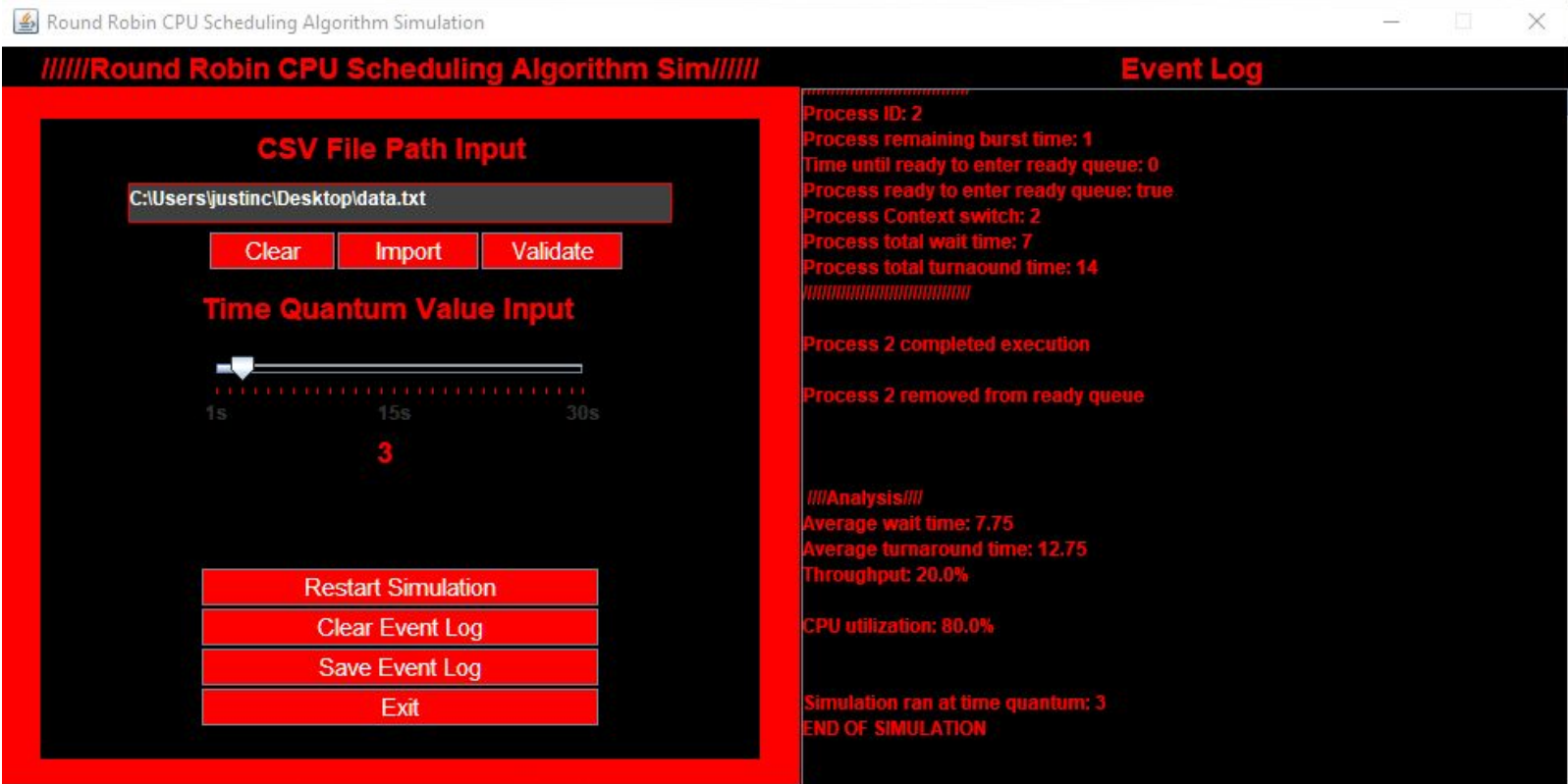
Average wait time: 8.5
Average turnaround time: 13.5
Throughput: 20.0%

CPU utilization: 65.0%

Simulation ran at time quantum: 2
END OF SIMULATION

By: Jcook03266
<https://github.com/jcook03266>
Round Robin CPU Scheduling Algorithm Implemented in Java

Time Quantum $t = 3$



Readout from the Event Log:

//////////Validating CSV File//////////
//////////Parsing data from CSV File//////////

Parsed Group 1:
Process ID: 1
Arrival Time: 0
Burst time: 5

Parsed Group 2:
Process ID: 2
Arrival Time: 1
Burst time: 7

Parsed Group 3:
Process ID: 3
Arrival Time: 0
Burst time: 2

By: Jcook03266
<https://github.com/jcook03266>
Round Robin CPU Scheduling Algorithm Implemented in Java

Parsed Group 4:

Process ID: 4

Arrival Time: 2

Burst time: 6

/////////////////Validating CSV File////////////////

/////////////////Parsing data from CSV File////////////////

Parsed Group 1:

Process ID: 1

Arrival Time: 0

Burst time: 5

Parsed Group 2:

Process ID: 2

Arrival Time: 1

Burst time: 7

Parsed Group 3:

Process ID: 3

Arrival Time: 0

Burst time: 2

Parsed Group 4:

Process ID: 4

Arrival Time: 2

Burst time: 6

New Process: 1 Added to Ready Queue

New Process: 3 Added to Ready Queue

New Process: 2 Added to Ready Queue

New Process: 4 Added to Ready Queue

////////////////////////////////////

Process ID: 3

Process remaining burst time: 2

Time until ready to enter ready queue: 0

Process ready to enter ready queue: true

Process Context switch: 0

Process total wait time: 0

Process total turnaround time: 2

By: Jcook03266
<https://github.com/jcook03266>
Round Robin CPU Scheduling Algorithm Implemented in Java

////////////////////////////////////

Process 3 completed execution

Process 3 removed from ready queue

////////////////////////////////////

Process ID: 1
Process remaining burst time: 5
Time until ready to enter ready queue: 0
Process ready to enter ready queue: true
Process Context switch: 0
Process total wait time: 0
Process total turnaround time: 5

////////////////////////////////////

////////////////////////////////////

Process ID: 1
Process remaining burst time: 2
Time until ready to enter ready queue: 0
Process ready to enter ready queue: true
Process Context switch: 1
Process total wait time: 0
Process total turnaround time: 5

////////////////////////////////////

Process 1 completed execution

Process 1 removed from ready queue

////////////////////////////////////

Process ID: 4
Process remaining burst time: 6
Time until ready to enter ready queue: 0
Process ready to enter ready queue: true
Process Context switch: 0
Process total wait time: 0
Process total turnaround time: 6

////////////////////////////////////

////////////////////////////////////

Process ID: 4
Process remaining burst time: 3
Time until ready to enter ready queue: 0
Process ready to enter ready queue: true
Process Context switch: 1
Process total wait time: 5
Process total turnaround time: 11

////////////////////////////////////

Process 4 completed execution

By: Jcook03266
<https://github.com/jcook03266>
Round Robin CPU Scheduling Algorithm Implemented in Java

Process 4 removed from ready queue

////////////////////////////////

Process ID: 2
Process remaining burst time: 7
Time until ready to enter ready queue: 0
Process ready to enter ready queue: true
Process Context switch: 0
Process total wait time: 0
Process total turnaround time: 7

////////////////////////////////

////////////////////////////////

Process ID: 2
Process remaining burst time: 4
Time until ready to enter ready queue: 0
Process ready to enter ready queue: true
Process Context switch: 1
Process total wait time: 2
Process total turnaround time: 9

////////////////////////////////

////////////////////////////////

Process ID: 2
Process remaining burst time: 1
Time until ready to enter ready queue: 0
Process ready to enter ready queue: true
Process Context switch: 2
Process total wait time: 7
Process total turnaround time: 14

////////////////////////////////

Process 2 completed execution

Process 2 removed from ready queue

///Analysis///

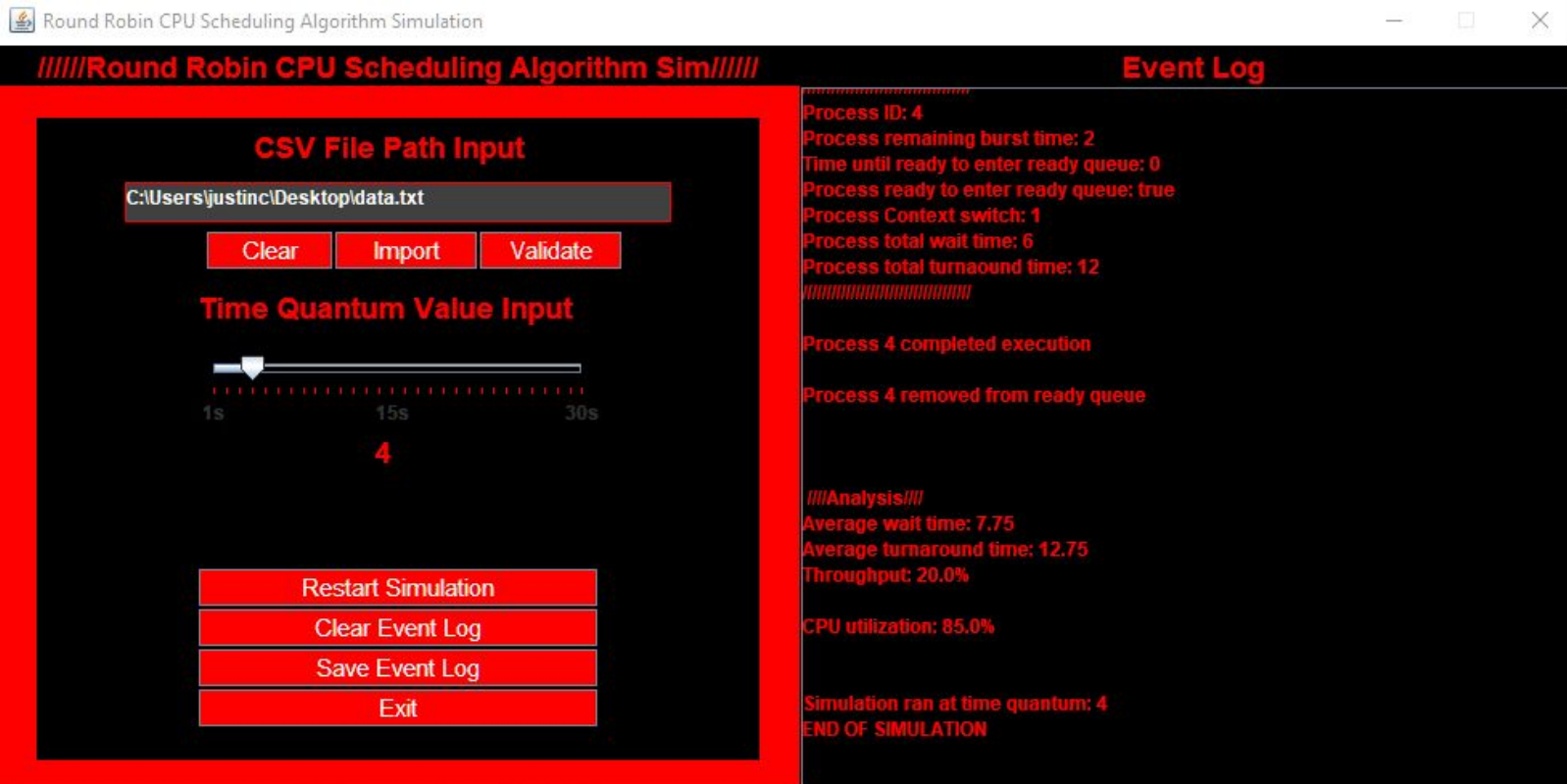
Average wait time: 7.75
Average turnaround time: 12.75
Throughput: 20.0%

CPU utilization: 80.0%

Simulation ran at time quantum: 3
END OF SIMULATION

By: Jcook03266
<https://github.com/jcook03266>
 Round Robin CPU Scheduling Algorithm Implemented in Java

Time Quantum $t = 4$



Readout from the Event Log:

//////////Validating CSV File//////////
 //////////Parsing data from CSV File//////////

Parsed Group 1:
 Process ID: 1
 Arrival Time: 0
 Burst time: 5

Parsed Group 2:
 Process ID: 2
 Arrival Time: 1
 Burst time: 7

Parsed Group 3:
 Process ID: 3
 Arrival Time: 0
 Burst time: 2

By: Jcook03266
<https://github.com/jcook03266>
Round Robin CPU Scheduling Algorithm Implemented in Java

Parsed Group 4:

Process ID: 4

Arrival Time: 2

Burst time: 6

//////////Validating CSV File//////////

//////////Parsing data from CSV File//////////

Parsed Group 1:

Process ID: 1

Arrival Time: 0

Burst time: 5

Parsed Group 2:

Process ID: 2

Arrival Time: 1

Burst time: 7

Parsed Group 3:

Process ID: 3

Arrival Time: 0

Burst time: 2

Parsed Group 4:

Process ID: 4

Arrival Time: 2

Burst time: 6

New Process: 1 Added to Ready Queue

New Process: 3 Added to Ready Queue

New Process: 2 Added to Ready Queue

New Process: 4 Added to Ready Queue

////////////////////////////////////

Process ID: 3

Process remaining burst time: 2

Time until ready to enter ready queue: 0

Process ready to enter ready queue: true

Process Context switch: 0

Process total wait time: 0

Process total turnaround time: 2

////////////////////////////////////

By: Jcook03266
<https://github.com/jcook03266>
Round Robin CPU Scheduling Algorithm Implemented in Java

Process 3 completed execution

Process 3 removed from ready queue

////////////////////////////////////

Process ID: 1
Process remaining burst time: 5
Time until ready to enter ready queue: 0
Process ready to enter ready queue: true
Process Context switch: 0
Process total wait time: 0
Process total turnaround time: 5

////////////////////////////////////

////////////////////////////////////

Process ID: 1
Process remaining burst time: 1
Time until ready to enter ready queue: 0
Process ready to enter ready queue: true
Process Context switch: 1
Process total wait time: 0
Process total turnaround time: 5

////////////////////////////////////

Process 1 completed execution

Process 1 removed from ready queue

////////////////////////////////////

Process ID: 2
Process remaining burst time: 7
Time until ready to enter ready queue: 0
Process ready to enter ready queue: true
Process Context switch: 0
Process total wait time: 0
Process total turnaround time: 7

////////////////////////////////////

////////////////////////////////////

Process ID: 2
Process remaining burst time: 3
Time until ready to enter ready queue: 0
Process ready to enter ready queue: true
Process Context switch: 1
Process total wait time: 2
Process total turnaround time: 9

////////////////////////////////////

Process 2 completed execution

By: Jcook03266
<https://github.com/jcook03266>
Round Robin CPU Scheduling Algorithm Implemented in Java

Process 2 removed from ready queue

////////////////////////////////////

Process ID: 4
Process remaining burst time: 6
Time until ready to enter ready queue: 0
Process ready to enter ready queue: true
Process Context switch: 0
Process total wait time: 0
Process total turnaround time: 6

////////////////////////////////////

////////////////////////////////////

Process ID: 4
Process remaining burst time: 2
Time until ready to enter ready queue: 0
Process ready to enter ready queue: true
Process Context switch: 1
Process total wait time: 6
Process total turnaround time: 12

////////////////////////////////////

Process 4 completed execution

Process 4 removed from ready queue

////Analysis////

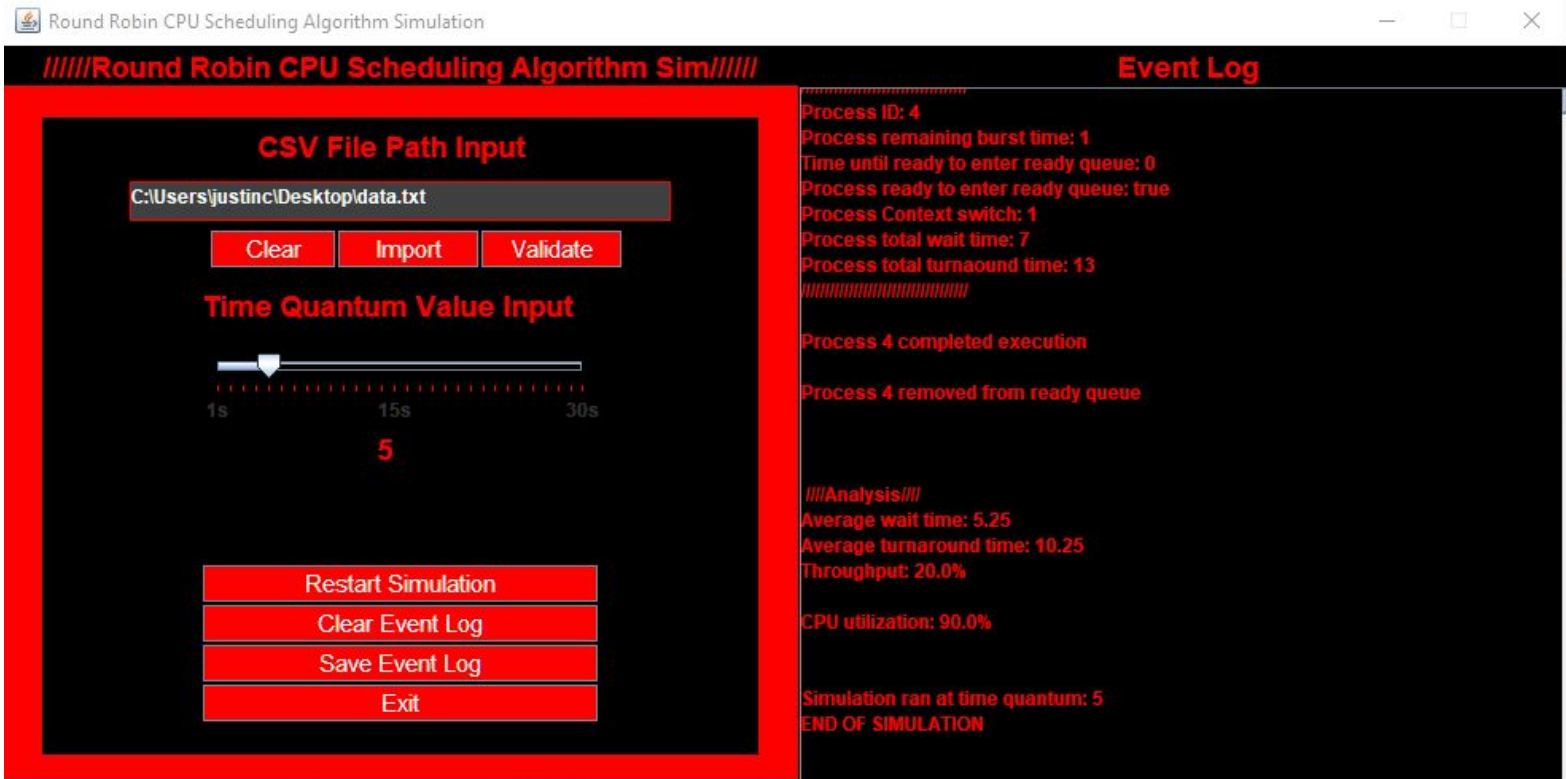
Average wait time: 7.75
Average turnaround time: 12.75
Throughput: 20.0%

CPU utilization: 85.0%

Simulation ran at time quantum: 4
END OF SIMULATION

By: Jcook03266
<https://github.com/jcook03266>
 Round Robin CPU Scheduling Algorithm Implemented in Java

Time Quantum $t = 5$



Readout from the Event Log:

/////////Validating CSV File/////////
 ///////////Parsing data from CSV File/////////

Parsed Group 1:
 Process ID: 1
 Arrival Time: 0
 Burst time: 5

Parsed Group 2:
 Process ID: 2
 Arrival Time: 1
 Burst time: 7

Parsed Group 3:
 Process ID: 3
 Arrival Time: 0
 Burst time: 2

By: Jcook03266
<https://github.com/jcook03266>
Round Robin CPU Scheduling Algorithm Implemented in Java

Parsed Group 4:

Process ID: 4

Arrival Time: 2

Burst time: 6

//////////Validating CSV File//////////

//////////Parsing data from CSV File//////////

Parsed Group 1:

Process ID: 1

Arrival Time: 0

Burst time: 5

Parsed Group 2:

Process ID: 2

Arrival Time: 1

Burst time: 7

Parsed Group 3:

Process ID: 3

Arrival Time: 0

Burst time: 2

Parsed Group 4:

Process ID: 4

Arrival Time: 2

Burst time: 6

New Process: 1 Added to Ready Queue

New Process: 3 Added to Ready Queue

New Process: 2 Added to Ready Queue

New Process: 4 Added to Ready Queue

////////////////////////////////////

Process ID: 1

Process remaining burst time: 5

Time until ready to enter ready queue: 0

Process ready to enter ready queue: true

Process Context switch: 0

Process total wait time: 0

Process total turnaround time: 5

////////////////////////////////////

By: Jcook03266
<https://github.com/jcook03266>
Round Robin CPU Scheduling Algorithm Implemented in Java

Process 1 completed execution

Process 1 removed from ready queue

////////////////////////////////////

Process ID: 3
Process remaining burst time: 2
Time until ready to enter ready queue: 0
Process ready to enter ready queue: true
Process Context switch: 0
Process total wait time: 0
Process total turnaround time: 2

////////////////////////////////////

Process 3 completed execution

Process 3 removed from ready queue

////////////////////////////////////

Process ID: 2
Process remaining burst time: 7
Time until ready to enter ready queue: 0
Process ready to enter ready queue: true
Process Context switch: 0
Process total wait time: 0
Process total turnaround time: 7

////////////////////////////////////

////////////////////////////////////

Process ID: 2
Process remaining burst time: 2
Time until ready to enter ready queue: 0
Process ready to enter ready queue: true
Process Context switch: 1
Process total wait time: 2
Process total turnaround time: 9

////////////////////////////////////

Process 2 completed execution

Process 2 removed from ready queue

////////////////////////////////////

Process ID: 4
Process remaining burst time: 6
Time until ready to enter ready queue: 0
Process ready to enter ready queue: true
Process Context switch: 0
Process total wait time: 0
Process total turnaround time: 6

By: Jcook03266
<https://github.com/jcook03266>
Round Robin CPU Scheduling Algorithm Implemented in Java

////////////////////////////////////

////////////////////////////////////

Process ID: 4
Process remaining burst time: 1
Time until ready to enter ready queue: 0
Process ready to enter ready queue: true
Process Context switch: 1
Process total wait time: 7
Process total turnaound time: 13

////////////////////////////////////

Process 4 completed execution

Process 4 removed from ready queue

////Analysis////

Average wait time: 5.25
Average turnaround time: 10.25
Throughput: 20.0%

CPU utilization: 90.0%

Simulation ran at time quantum: 5
END OF SIMULATION

By: Jcook03266
<https://github.com/jcook03266>
Round Robin CPU Scheduling Algorithm Implemented in Java

What is the significance of these results?

If we analyze the results of the 5 different time quanta on the same process input file we can see the following trends: as we give the processes a greater time quantum aka larger time to work with when evenly divided up then the average wait time decreases meaning we minimize the time processes have to wait in order to be executed, alongside this trend we also have the average turnaround time also decreasing which is the total time it takes for a process to be executed from start to finish, and lastly, we also have the cpu utilization going up and up almost to that magic number of 100% cpu utilization which all algorithms aim for because it signals that no cpu cycles are being wasted. For scheduling algorithms there are multiple criteria that must be addressed for optimization purposes: 1.) CPU Utilization, we must maximize the cpu and keep its utilization at 100%, keep it busy or waste cycles and precious time and energy that can be spent wisely, 2.) CPU throughput, we must maximize the number of jobs processed, mind you in these runs the throughput remained the same and this mainly because of the data set provided and the algorithm at hand, if you use another algorithm then you will get a different throughput, and if you use a different data set then you will indeed get a different throughput for this algorithm as well, 3.) turn around time must be kept low, we have to minimize the total time it takes to complete processes in order to have a much higher efficiency 4.) we must minimize waiting times as well, less waiting means more doing and less idling and less wasting precious cpu cycles, and lastly, 5.) we must minimize the response time aka the time from when a request is submitted until the first response is produced. With the culmination of all of these factors combined we find out just how effective round robin is in multiple areas and where it lacks in another, namely throughput but this can be further diagnosed by using different data sets and then further comparing results in order to deduce a much more concise idea. With this data set it's clear that the higher the time quanta the better and more optimized the operation becomes, with a time quantum like 5 nearing the upper echelon of performance. And this concludes the analysis of the varying trials displayed here. Thank you. - Justin Cook