****

**OPERATING SYSTEM**

**CPU SCHEDULING ALGORITHM SIMULATOR**

**GRADE**

**Submitted by:**

*Signature*

*Student Names* : Jan Joseph Sarmiento \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Ian Emmanuel Velasco \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Roy Bautista \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

JofetVillaflor \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**ENGR. JAY-AR P. LALATA**

*Professor*

**November 2016**

1. Introduction

An operating systems is one of the most important software that runs on a computer. It manages the computer memory and processes as well as all of its software and hardware. Your computer's operating system (OS) manages all of the software and hardware on the computer. Most of the time, there are several different computer programs running at the same time, and they all need to access your computer's central processing unit (CPU), memory, and storage.

1. Description of the Project

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. The aim of CPU scheduling is to make the system efficient, fast and fair. The technology involved in developing the project are Javascript, jQuery and HTML.

The scheduler is mainly concerned with:

* CPU utilization

To make out the best use of CPU and not to waste any CPU cycle, CPU would be working most of the time (Ideally 100% of the time). Considering a real system, CPU usage should range from 40% (lightly loaded) to 90% (heavily loaded.)

* Throughput

It is the total number of processes completed per unit time or rather say total amount of work done in a unit of time. This may range from 10/second to 1/hour depending on the specific processes.

* Turnaround time

It is the amount of time taken to execute a particular process, i.e. The interval from time of submission of the process to the time of completion of the process (Wall clock time).

* Waiting time

The sum of the periods spent waiting in the ready queue amount of time a process has been waiting in the ready queue to acquire get control on the CPU.

* Load average

It is the average number of processes residing in the ready queue waiting for their turn to get into the CPU.

* Response time

Amount of time it takes from when a request was submitted until the first response is produced. Remember, it is the time till the first response and not the completion of process execution (final response).

Different algorithms used in CPU scheduling

* First-Come, First-Served(FCFS) Scheduling

With this Scheduling it is a non-preemptive.Jobs are executed on the first come first serve basis

* Shortest-Job-Next(SJN) Scheduling

With this Scheduling it is a non-preemptive.The shortest Burst time will be the first to be executed

* Priority Scheduling(PS) Scheduling

With This Scheduling it is a non-preemptive. Each process is assigned a priority and the process with the highest priority will be the first to be executed

* Shortest Remaining Time(SRT) Scheduling

Shortest remaining time (SRT) is the preemptive version of the SJN algorithm. The processor is allocated to the job closest to completion but it can be preempted by a newer ready job with shorter time to completion

* Round Robing(RR) Scheduling

Is the preemptive process scheduling algorithm.Each process is provided a fix time to execute, it is called a **quantum.** Once a process is executed for a given time period, it is preempted and other process executes for a given time period.

These algorithms are non-preemptive or preemptive. Non-preemptive algorithms are designed so that once a process enters the running state, it cannot be preempted until it completes its allotted time, whereas the preemptive scheduling is based on priority where a scheduler may preempt a low priority running process anytime when a high priority process enters into a ready state.

.

1. Objectives

The Objectives of the study is toImplementation of the CPU scheduling algorithms using JavaScript to simulates the Following different algorithms

1. FCFS (First-Come First Serve)
2. SJF (Shortest Job First)
3. SRTF (Shortest Remaining Time First)
4. PS - Pre-emptive and Non pre-emptive (Priority Scheduling)
5. RR (Round Robin)

That will Compute the average turnaround time and average waiting time and compare the performances of the scheduling algorithm For the purpose of analysis and testing the user first specifies each process along with its information such as arrival times and burst time and then algorithms can be computed producing output in appropriate format readability and to discover the best suited algorithm to use.

1. Significance of the Study

This study had been a significant in promoting Operating system and what it can do. This study also served as a future reference for researchers.

The proponents are very confident that this study is beneficial to the following:

**For future researchers,** they will be able to learn about operating system and what the scheduler is all about and learn the different algorithms that will help them in their future research

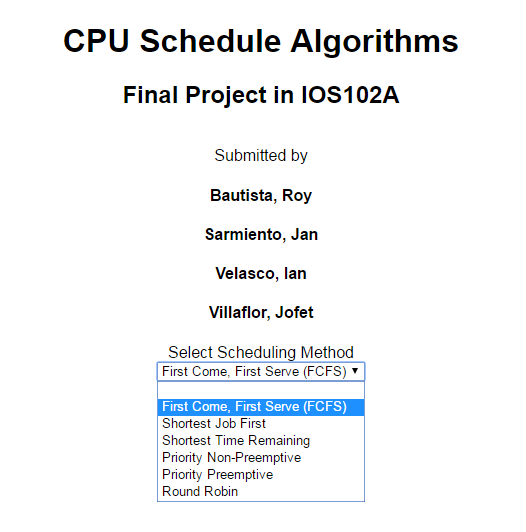
**For proponents,** the Project taught them the values they need to work as a team and as they research the topic they learn about thing that they do not know about operating system and how it works andlearn all the aspects needed in a project development team also it also serves as reference to the future researchers.

1. Scope and Delimitations

The scope and the Delimitations of this study is limited to html. At the evaluation phase, the performance and stability were compared with other programing language.

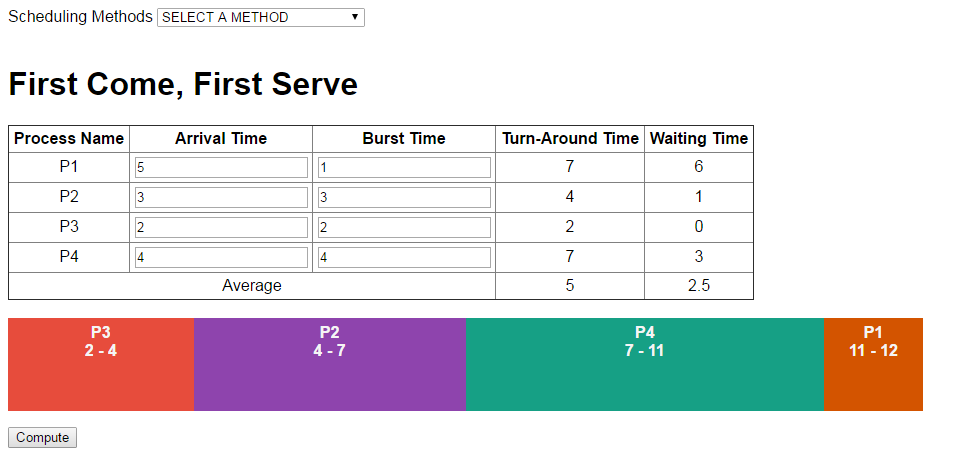
We use JavaScript on the html language in the development of the scheduler. JavaScript has many desirable features however it has security issues once the appended onto web pages execute on client servers immediately and therefore can also be used to exploit the user’s system and different layout engines may render JavaScript differently resulting in inconsistency in terms of functionality and interface. The software was develop and tested on based personal computer.

1. Screen Output



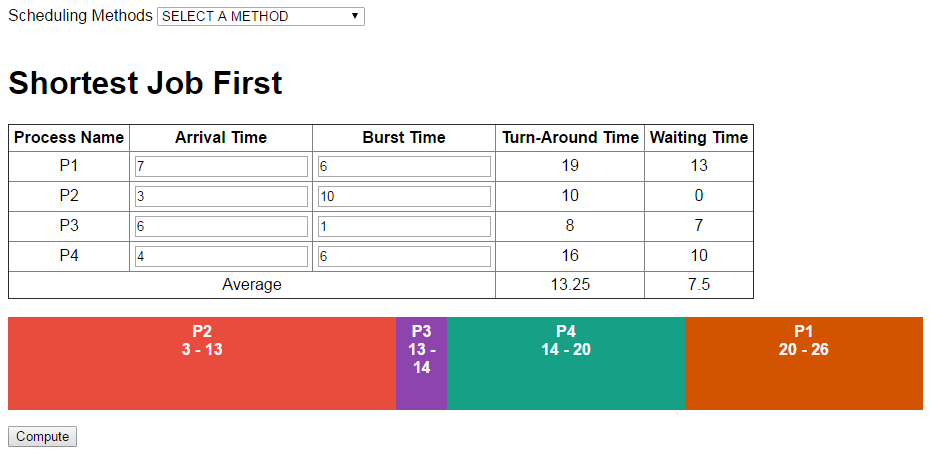
**Figure 1. Index Page**

Figure 1 shows the index page of the project which contains the title, members of the team and a dropdown list which consists of the scheduling methods available

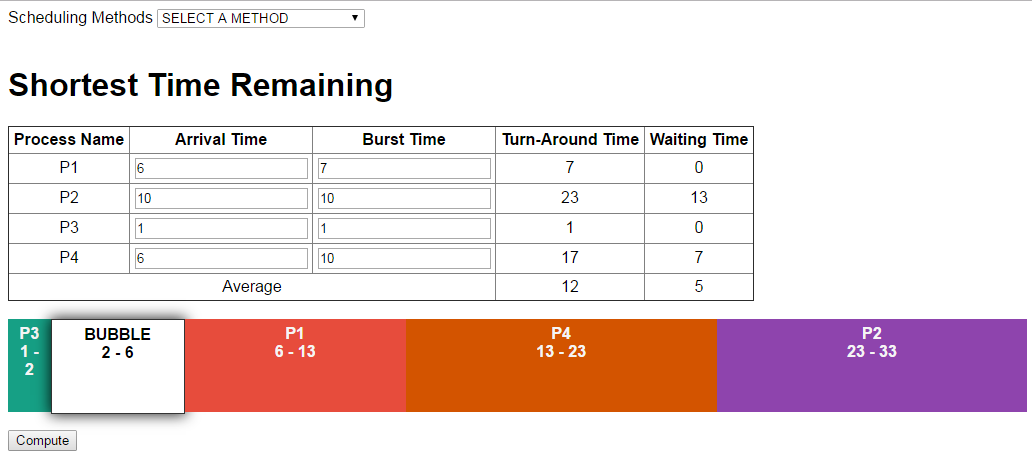


**Figure 2. First Come First Serve Page**

This page contains the title of the algorithm being simulated, a table where you can input parameters for arrival time and burst time for 4 (four) processes, a button to initialize computation and a Gantt Chart that is shown only after the computation is completed. The table's columns for Turn-Around time, Waiting Time, and Average for both, are automatically filled out after computation, and a drop down menu at the top to navigate to different algorithms available.

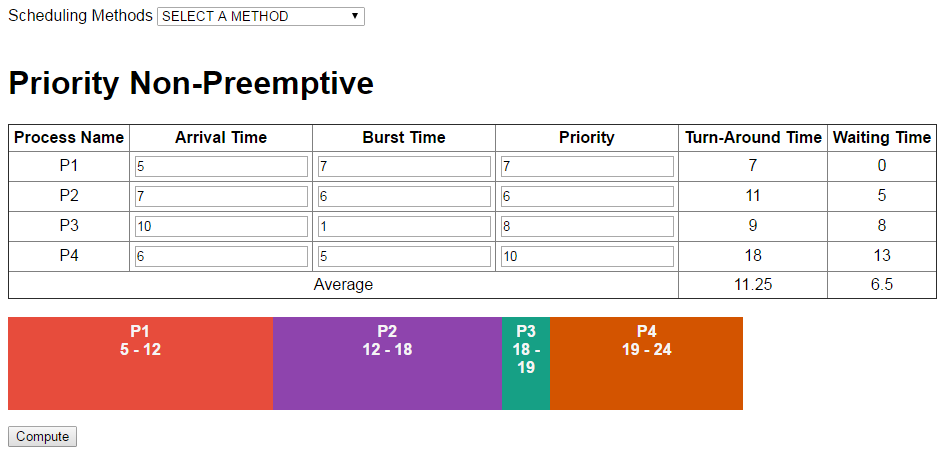
 **Figure 3. Shortest Job First Page**

This page has the same content as **Figure 2** but is labeled for the ***Shortest Job First*** algorithm and works differently



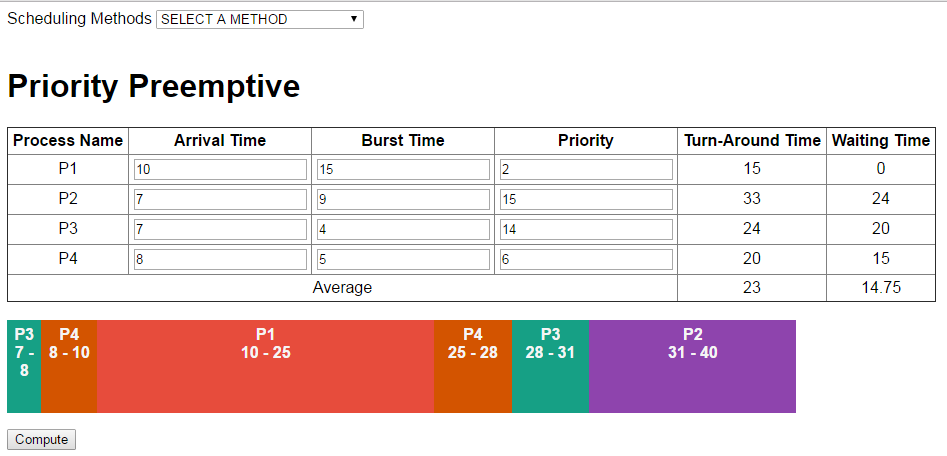
**Figure 4. Shortest Time Remaining Page**

This page has the same content as **Figure 2** but is labeled for the ***Shortest Time Remaining*** algorithm and works differently.



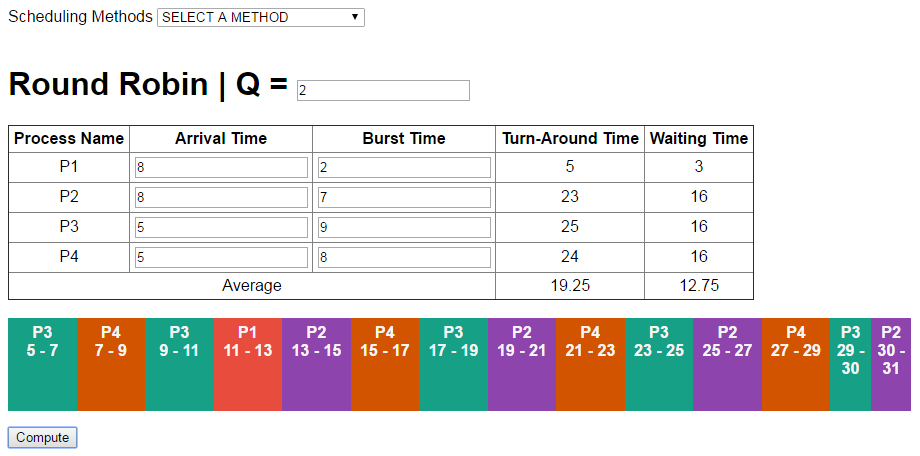
**Figure 5. Priority - Non Preemptive**

This page is identical to **Figure 2** but has an additional column on the table which is labeled ***Priority,*** providing an input field to indicate the priority value for each process.

****

**Figure 6. Priority - Preemptive**

The contents of this page is exactly the same as **Figure 5** but it is labeled differently for the ***Priority - Preemptive algorithm*** and works differently

****

**Figure 7. Priority - Preemptive**

This page has the same content as **Figure 2** but is labeled for the ***Shortest Job First*** algorithm and works differently. This page also has an additional input field for the ***Q (Quantum)*** value for the ***Round Robin algorithm***

1. Source Code

**index.html**

<html>  
 <head>  
 <title>CPU Scheduling Algorithm - Final Project for IOS102A</title>  
 <style>  
 body {  
 font-family : Arial;  
 }  
 td {  
 padding : 0.3em;  
 text-align : center;  
 }  
 thead {  
 font-weight : bold;  
 }  
 .gantt\_block {  
 text-align: center;  
 padding: 0.4em;  
 height: 5em;  
 float: left;  
 display: block;  
 color: whitesmoke;  
 font-weight: bold;  
 }  
 table {  
 border-collapse : collapse;  
 }  
 .bubble {  
 -webkit-box-shadow: 0px 0px 21px 1px rgba(0,0,0,0.84);  
 -moz-box-shadow: 0px 0px 21px 1px rgba(0,0,0,0.84);  
 box-shadow: 0px 0px 21px 1px rgba(0,0,0,0.84);  
 }  
 </style>  
 <script src="js/jquery-3.1.0.min.js"></script>  
 <script>  
 $(document).ready(function () {  
 *// this code handles the page changing according to the selected value of the dropdown* $('#methods').change(function(){ location.href = $(this).val(); });  
 })  
 </script>  
 </head>  
 <body>  
 <div style="margin-top: 8em;">  
 <center>  
 <h1>CPU Scheduling Algorithms</h1>  
 <h2>Final Project in IOS102A</h2>  
 <br/>  
 Submitted by  
 <h4>Bautista, Roy</h4>  
 <h4>Sarmiento, Jan</h4>  
 <h4>Velasco, Ian</h4>  
 <h4>Villaflor, Jofet</h4>  
 <div>  
 Select Scheduling Method<br/>  
 <select id="methods">  
 <option value=""></option>  
 <option value="FirstComeFirstServe.html">First Come, First Serve (FCFS)</option>  
 <option value="ShortestJobFirst.html">Shortest Job First</option>  
 <option value="ShortestTimeRemaining.html">Shortest Time Remaining</option>  
 <option value="PriorityNonPreemptive.html">Priority Non-Preemptive</option>  
 <option value="PriorityPreemptive.html">Priority Preemptive</option>  
 <option value="RoundRobin.html">Round Robin</option>  
 </select>  
 </div>  
 </center>  
 </div>  
 </body>  
</html>

**FirstComeFirstServe.html**

<html>  
 <head>  
 <title>First Come, First Serve - Final Project in IOS102A</title>  
 <style>  
 body {  
 font-family : Arial;  
 }  
 td {  
 padding : 0.3em;  
 text-align : center;  
 }  
 thead {  
 font-weight : bold;  
 }  
 .gantt\_block {  
 text-align: center;  
 padding: 0.4em;  
 height: 5em;  
 float: left;  
 display: block;  
 color: whitesmoke;  
 font-weight: bold;  
 }  
 table {  
 border-collapse : collapse;  
 }  
 .bubble {  
 -webkit-box-shadow: 0px 0px 21px 1px rgba(0,0,0,0.84);  
 -moz-box-shadow: 0px 0px 21px 1px rgba(0,0,0,0.84);  
 box-shadow: 0px 0px 21px 1px rgba(0,0,0,0.84);  
 }  
 </style>  
 <script src="js/jquery-3.1.0.min.js"></script>  
 <script>  
 $(document).ready(function(){  
 $('#INIT\_COMPUTE').click(function(){ *// listener for the compute button* if(*checkValues*()){ *// checks if the values of the input fields are acceptable (numerical values only)* var ready\_queue = []; *// variable to contain process that cannot yet go directly to the cpu* var my\_gantt\_chart = $('#gantt\_chart'); *// listener for the gantt chart div* var my\_colors = [ *// array of colors for the gantt chart* '#E74C3C',  
 '#8E44AD',  
 '#16A085',  
 '#D35400'  
 ];  
  
 my\_gantt\_chart.empty();  
  
 *// populate `ready\_queue` array of process via arrival time values* for(var x=0; x <= *GET\_BURSTTIME\_TOTAL*(); x++){  
 $('.arrival\_time').each(function(index){  
 if(*parseFloat*(x) == *parseFloat*($(this).val())){  
 ready\_queue.push((index+1));  
 }  
 });  
 }  
  
 *// start processing* var start\_time = *GET\_ARRIVALTIME\_LOWEST*(); *// create and initialize start\_time (for individual div of the gantt chart) based on the lowest arrival time existing* var end\_time = 0; *// create and initialize end\_time (for individual div of the gantt chart) to 0 and listen and change for incrementing values* $.each(ready\_queue, function(index, value){ *// loops through the read\_queue variable to get new process* var curr\_process = 'P'+value; *// get current process on top of queue* var curr\_arrivaltime = $('[data-process="P'+value+'"][class="arrival\_time"]').val(); *// jQuery select the corresponding arrival time of the selected process* var curr\_bursttime = $('[data-process="P'+value+'"][class="burst\_time"]').val(); *// jQuery select the corresponding arrival time of the selected process* var curr\_width = (curr\_bursttime / *parseFloat*(*GET\_BURSTTIME\_TOTAL*())) \* 80; *// compute width of the div for the gantt chart* var elem\_TAT = $('#'+curr\_process+'\_TAT'); *// jQuery select the Turn-Around Time cell for the current process* var elem\_WT = $('#'+curr\_process+'\_WT'); *// jQuery select the Waiting Time cell for the current process  
  
 // check for BUBBLE* if(curr\_arrivaltime > start\_time){ *// if current process' arrival time is GREATER than the start\_time then a BUBBLE exists* my\_gantt\_chart.append('<div class="gantt\_block bubble" style="background-color: white; width: 10%; border: 1px solid #333333; color: black;">BUBBLE<br/>'+start\_time+' - '+curr\_arrivaltime+'</div>'); *// append bubble div to gantt chart* start\_time = *parseFloat*(curr\_arrivaltime); *// re-initialize start\_time to the current process' arrival time* }  
  
 end\_time = start\_time + *parseFloat*(curr\_bursttime); *// compute end\_time by adding current process' burst time to start\_time  
 // append process' div to gantt chart* my\_gantt\_chart.append('<div class="gantt\_block" style="background-color: '+my\_colors[index]+'; width: '+curr\_width+'%;">'+curr\_process+'<br/>'+start\_time+' - '+end\_time+'</div>');  
 *// compute and append TAT* var TAT = end\_time - curr\_arrivaltime;  
 elem\_TAT.empty().append(TAT);  
 *// compute and append WT* var WT = start\_time - curr\_arrivaltime;  
 elem\_WT.empty().append(WT);  
 start\_time = end\_time; *// re-initialize start\_time to end\_time value to for the next process starting point or computation of bubble if existing* });  
  
 *// compute average of TAT* var total\_tat = 0;  
 $('.TAT').each(function(index, value){  
 total\_tat += *parseFloat*($(this).text());  
 });  
 $('#AVG\_TAT').empty().append((*parseFloat*(total\_tat)/*parseFloat*(ready\_queue.length)));  
 *// compute average WT* var total\_wt = 0;  
 $('.WT').each(function(index, value){  
 total\_wt += *parseFloat*($(this).text());  
 });  
 $('#AVG\_WT').empty().append((*parseFloat*(total\_wt)/*parseFloat*(ready\_queue.length)));  
  
 }  
 });  
 $('#methods').change(function(){  
 location.href = $(this).val();  
 })  
 });  
  
 function *GET\_BURSTTIME\_TOTAL*(){  
 var total = 0.0;  
 $('.burst\_time').each(function(index){  
 total += *parseFloat*($(this).val());  
 });  
  
 *// check if for cpu waiting time* if(*parseFloat*(total) < *GET\_ARRIVALTIME\_HIGHEST*()){  
 total = *GET\_ARRIVALTIME\_HIGHEST*();  
 }  
  
 return *parseFloat*(total);  
 }  
  
 function *GET\_ARRIVALTIME\_HIGHEST*(){  
 var highest = 0;  
 $('.arrival\_time').each(function(){  
 if(highest == 0){  
 highest = *parseFloat*($(this).val());  
 }  
 if(*parseFloat*($(this).val()) > highest){  
 highest = *parseFloat*($(this).val());  
 }  
 });  
 return *parseFloat*(highest);  
 }  
  
 function *GET\_ARRIVALTIME\_LOWEST*(){  
 var lowest = *GET\_ARRIVALTIME\_HIGHEST*();  
 $('.arrival\_time').each(function(){  
 var at = *parseFloat*($(this).val());  
 if(at < lowest){  
 lowest = at;  
 }  
 });  
  
 return *parseFloat*(lowest);  
 }  
  
 function *checkValues*(){  
 var flag = true;  
 $('#cust\_console').empty();  
 $('.arrival\_time').each(function(index){  
 *// check if arrival\_time is filled out* if($(this).val() == '' || !$.isNumeric($(this).val())){  
 $('#cust\_console').append('Please input a number for Arrival Time for Process P'+(index+1)+'<br/>');  
 flag = false;  
 }  
 });  
 $('.burst\_time').each(function(index){  
 *// check if burst\_time is filled out* if($(this).val() == '' || !$.isNumeric($(this).val())){  
 $('#cust\_console').append('Please input a number for Burst Time for Process P'+(index+1)+'<br/>');  
 flag = false;  
 }  
 });  
  
 return flag;  
 }  
 </script>  
 </head>  
 <body>  
 Scheduling Methods  
 <select id="methods">  
 <option value="">SELECT A METHOD</option>  
 <option value="FirstComeFirstServe.html">First Come, First Serve (FCFS)</option>  
 <option value="ShortestJobFirst.html">Shortest Job First</option>  
 <option value="ShortestTimeRemaining.html">Shortest Time Remaining</option>  
 <option value="PriorityNonPreemptive.html">Priority Non-Preemptive</option>  
 <option value="PriorityPreemptive.html">Priority Preemptive</option>  
 <option value="RoundRobin.html">Round Robin</option>  
 </select>  
 <br/>  
 <br/>  
 <h1>First Come, First Serve</h1>  
 <table border="1">  
 <thead>  
 <tr>  
 <td>Process Name</td>  
 <td>Arrival Time</td>  
 <td>Burst Time</td>  
 <td>Turn-Around Time</td>  
 <td>Waiting Time</td>  
 </tr>  
 </thead>  
 <tbody>  
 <tr>  
 <td>P1</td>  
 <td><input data-process="P1" type="text" class="arrival\_time" /></td>  
 <td><input data-process="P1" type="text" class="burst\_time" /></td>  
 <td><span class="TAT" id="P1\_TAT"></span></td>  
 <td><span class="WT" id="P1\_WT"></span></td>  
 </tr>  
 <tr>  
 <td>P2</td>  
 <td><input data-process="P2" type="text" class="arrival\_time" /></td>  
 <td><input data-process="P2" type="text" class="burst\_time" /></td>  
 <td><span class="TAT" id="P2\_TAT"></span></td>  
 <td><span class="WT" id="P2\_WT"></span></td>  
 </tr>  
 <tr>  
 <td>P3</td>  
 <td><input data-process="P3" type="text" class="arrival\_time" /></td>  
 <td><input data-process="P3" type="text" class="burst\_time" /></td>  
 <td><span class="TAT" id="P3\_TAT"></span></td>  
 <td><span class="WT" id="P3\_WT"></span></td>  
 </tr>  
 <tr>  
 <td>P4</td>  
 <td><input data-process="P4" type="text" class="arrival\_time" /></td>  
 <td><input data-process="P4" type="text" class="burst\_time" /></td>  
 <td><span class="TAT" id="P4\_TAT"></span></td>  
 <td><span class="WT" id="P4\_WT"></span></td>  
 </tr>  
 <tr>  
 <td colspan="3">Average</td>  
 <td><span id="AVG\_TAT"></span></td>  
 <td><span id="AVG\_WT"></span></td>  
 </tr>  
 </tbody>  
 </table>  
 <br/>  
 <div style="width: 80%">  
 <div id="gantt\_chart">  
 </div>  
 </div>  
 <div style="clear: both;"></div>  
 <p id="cust\_console" style="color:red;"></p>  
 <button id="INIT\_COMPUTE">Compute</button>  
 </body>  
</html>

**ShortestTimeRemaining.html**

<html>  
 <head>  
 <title>Shortest Time Remaining - Final Project in IOS102A</title>  
 <style>  
 body {  
 font-family : Arial;  
 }  
 td {  
 padding : 0.3em;  
 text-align : center;  
 }  
 thead {  
 font-weight : bold;  
 }  
 .gantt\_block {  
 text-align: center;  
 padding: 0.4em;  
 height: 5em;  
 float: left;  
 display: block;  
 color: whitesmoke;  
 font-weight: bold;  
 }  
 table {  
 border-collapse : collapse;  
 }  
 .bubble {  
 -webkit-box-shadow: 0px 0px 21px 1px rgba(0,0,0,0.84);  
 -moz-box-shadow: 0px 0px 21px 1px rgba(0,0,0,0.84);  
 box-shadow: 0px 0px 21px 1px rgba(0,0,0,0.84);  
 text-align: center;  
 padding: 0.4em;  
 height: 5em;  
 float: left;  
 display: block;  
 color: whitesmoke;  
 font-weight: bold;  
 }  
 </style>  
 <script src="js/jquery-3.1.0.min.js"></script>  
 <script>  
 var *ready\_queue* = [];  
 var *cpu\_process* = null;  
 var *cpu\_bursttime* = null;  
 var *GLOBAL\_startTime* = null;  
 var *GLOBAL\_endTime* = null;  
 var *GLOBAL\_bubbleStart* = null;  
 var *GLOBAL\_bubbleEnd* = null;  
 var *my\_console* = $('#cust\_console');  
 var *my\_gantt\_chart* = $('#gantt\_chart');  
 var *my\_colors* = [  
 '#E74C3C',  
 '#8E44AD',  
 '#16A085',  
 '#D35400'  
 ];  
 var *pr\_done* = 0;  
  
 $(document).ready(function(){  
  
 $('input').each(function(){  
 $(this).val(Math.*floor*(Math.*random*() \* 10) + 1);  
 });  
  
 $('#INIT\_COMPUTE').click(function(){  
 if(*checkValues*()){  
 var i = *GET\_ARRIVALTIME\_LOWEST*();  
 do{  
 *PROCESS\_ARRIVALS*(i); *// insert process to `ready\_queue`* if(*cpu\_process* != null){  
 *cpu\_bursttime*--; *// decrement value* console.log(i+'\t: DCRMNT BT : P'+*cpu\_process*+'/'+*cpu\_bursttime*);  
  
 if(*cpu\_bursttime* == 0){ *// cpu has 0 bt value  
 GLOBAL\_endTime* = i; *// change end time to loop value `i`* var bt\_of\_process = *parseFloat*(*GLOBAL\_endTime* - *GLOBAL\_startTime*);  
 var curr\_width = ((bt\_of\_process / *GET\_BURSTTIME\_SUM*()) \* 80); *// compute width for gantt chart* $('#gantt\_chart').append('<div data-process="'+*cpu\_process*+'" data-start="'+*GLOBAL\_startTime*+'" data-end="'+*GLOBAL\_endTime*+'" class="gantt\_block" style="background-color: '+*my\_colors*[(*cpu\_process*-1)]+'; width: '+curr\_width+'%;">P'+*cpu\_process*+'<br/>'+*GLOBAL\_startTime*+' - '+*GLOBAL\_endTime*+'</div>');  
 console.log(i+'\t: '+' ADD\_GANTT\_1 = '+*cpu\_process*+'/'+*cpu\_bursttime*);  
 *cpu\_process* = null;  
 *cpu\_bursttime* = null;  
 *pr\_done*++;  
 }  
 }  
  
 if(*cpu\_process* == null){  
 if(*ready\_queue*.length > 0){*SORT\_READY\_QUEUE*(); *// sort ready queue for smallest process first  
 cpu\_process* = *ready\_queue*[0].split('?')[0]; *// add process to cpu  
 cpu\_bursttime* = *ready\_queue*[0].split('?')[1];  
 *ready\_queue*.shift();  
 *GLOBAL\_startTime* = i;  
  
 if(*GLOBAL\_bubbleStart* != null){  
 console.log(i+'\t: BUBBLE END');  
 var bubble\_width = ((i - *GLOBAL\_bubbleStart*) / *GET\_BURSTTIME\_SUM*()) \* 80;  
 $('#gantt\_chart').append('<div class="bubble" style="background-color: white; width: '+bubble\_width+'%; border: 1px solid #333333; color: black;">BUBBLE<br/>'+*GLOBAL\_bubbleStart*+' - '+i+'</div>');  
 *GLOBAL\_bubbleStart* = null;  
 }  
  
 console.log(i+'\t: ADD PR to CPU : P'+*cpu\_process*+'/'+*cpu\_bursttime*);  
 }else{ *// bubble* if(*GLOBAL\_bubbleStart* == null){  
 *GLOBAL\_bubbleStart* = i;  
 }  
 console.log(i+'\t: BUBBLE INC - Start = '+*GLOBAL\_bubbleStart*);  
 }  
 }else{  
 *// algo for change process - PREEMPTIVE NATURE* if(*ready\_queue*.length > 0){  
 var og\_temp\_pr = *ready\_queue*[0].split('?')[0]; *// get and put process from ready\_queue to temp value* var og\_temp\_bt = *ready\_queue*[0].split('?')[1]; *// get and put burst time from ready\_queue to temp value* if(og\_temp\_bt < *cpu\_bursttime*){ *// determines if cpu will change to lower bt process  
 GLOBAL\_endTime* = i; *// change end time to loop value `i`* var bt\_of\_process = *parseFloat*(*GLOBAL\_endTime* - *GLOBAL\_startTime*); *// get burst time of process* var curr\_width = ((bt\_of\_process / *GET\_BURSTTIME\_SUM*()) \* 80); *// compute width for gantt chart* $('#gantt\_chart').append('<div data-process="'+*cpu\_process*+'" data-start="'+*GLOBAL\_startTime*+'" data-end="'+*GLOBAL\_endTime*+'" class="gantt\_block" style="background-color: '+*my\_colors*[(*cpu\_process*-1)]+'; width: '+curr\_width+'%;">P'+*cpu\_process*+'<br/>'+*GLOBAL\_startTime*+' - '+*GLOBAL\_endTime*+'</div>');  
 console.log(i+'\t: '+' ADD\_GANTT\_2 = '+*cpu\_process*+'/'+*cpu\_bursttime*);  
  
 *// change process* console.log(i+'\t: CHNG PR : P'+*cpu\_process*+'/'+*cpu\_bursttime*+' to '+og\_temp\_pr+'/'+og\_temp\_bt);  
 *ready\_queue*.push(*cpu\_process*+'?'+*cpu\_bursttime*); *// put back process from cpu to ready\_queue  
 cpu\_process* = og\_temp\_pr; *// apply new cpu process  
 cpu\_bursttime* = og\_temp\_bt; *// apply new cpu burst time  
 ready\_queue*.shift(); *// remove applied process from ready\_queue  
 SORT\_READY\_QUEUE*(); *// sort ready\_queue for safety  
 GLOBAL\_startTime* = i;  
 }  
 }  
 }  
 i++;  
 }while(*pr\_done* < 4);  
  
 var et\_array\_p = [];  
 var et\_array\_e = [];  
 $('.gantt\_block').each(function (index) { *// loop through items in the gantt chart for computation of the  
 // get gantt item's process, start and end time, and arrival time* var tmp\_process = *parseFloat*($(this).data('process'));  
 var tmp\_start = *parseFloat*($(this).data('start'));  
 var tmp\_end = *parseFloat*($(this).data('end'));  
 var tmp\_arrival = *parseFloat*($('[data-process="'+(tmp\_process)+'"][class="arrival\_time"]').val());  
 *// selectors for the turnaround time and waiting time* var slctr\_tat = $('#P'+tmp\_process+'\_TAT');  
 var slctr\_wt = $('#P'+tmp\_process+'\_WT');  
 var inArray = $.inArray(tmp\_process, et\_array\_p); *// check if the current process is in array* slctr\_tat.empty().append(tmp\_end - tmp\_arrival);  
  
 var curr\_wt = slctr\_wt.text();  
 if(inArray > -1){  
 slctr\_wt.empty().append(*parseFloat*(curr\_wt) + (tmp\_start - et\_array\_e[inArray]));  
 et\_array\_e[inArray] = tmp\_end;  
 }else{  
 slctr\_wt.append(tmp\_start - tmp\_arrival);  
 et\_array\_p.push(tmp\_process);  
 et\_array\_e.push(tmp\_end);  
 }  
 });  
  
 *// computes average turnaround time* var total\_tat = 0;  
 $('.TAT').each(function (index) {  
 total\_tat += *parseFloat*($(this).text());  
 });  
 $('#AVG\_TAT').empty().append((*parseFloat*(total\_tat)/$('.TAT').length));  
  
 *// computes average waiting time* var total\_wt = 0;  
 $('.WT').each(function (index) {  
 total\_wt += *parseFloat*($(this).text());  
 });  
 $('#AVG\_WT').empty().append((*parseFloat*(total\_wt)/$('.WT').length));  
 }  
 });  
  
 $('#methods').change(function(){  
 location.href = $(this).val();  
 })  
 });  
  
 function *checkValues*(){  
 var flag = true;  
 $('#cust\_console').empty();  
 $('.arrival\_time').each(function(index){  
 *// check if arrival\_time is filled out* if($(this).val() == '' || !$.isNumeric($(this).val())){  
 $('#cust\_console').append('Please input a number for Arrival Time for Process P'+(index+1)+'<br/>');  
 flag = false;  
 }  
 })  
 $('.burst\_time').each(function(index){  
 *// check if burst\_time is filled out* if($(this).val() == '' || !$.isNumeric($(this).val())){  
 $('#cust\_console').append('Please input a number for Burst Time for Process P'+(index+1)+'<br/>');  
 flag = false;  
 }  
 })  
 $('.priority').each(function(index){  
 *// check if burst\_time is filled out* if($(this).val() == '' || !$.isNumeric($(this).val())){  
 $('#cust\_console').append('Please input a number for Priority for Process P'+(index+1)+'<br/>');  
 flag = false;  
 }  
 })  
  
 return flag;  
 }  
  
 function *GET\_BT\_OF\_PROCESS*(cpu){  
 return *parseFloat*(Math.*round*($('[data-process="'+(cpu)+'"][class="burst\_time"]').val()));  
 }  
   
 function *GET\_ARRIVALTIME\_LOWEST*(){  
 var lowest = *GET\_ARRIVALTIME\_HIGHEST*();  
 $('.arrival\_time').each(function(){  
 if(*parseFloat*($(this).val()) < lowest){  
 lowest = *parseFloat*($(this).val());  
 }  
 });  
  
 return lowest;  
 }  
  
 function *GET\_PROCESSTIME*(){  
   
 }  
  
 function *GET\_ARRIVALTIME\_HIGHEST*(){  
 var highest = 0;  
 $('.arrival\_time').each(function(){  
 if(highest == 0){  
 highest = *parseFloat*($(this).val());  
 }  
 if(*parseFloat*($(this).val()) > highest){  
 highest = *parseFloat*($(this).val());  
 }  
 });  
 return *parseFloat*(highest);  
 }  
  
 function *GET\_BURSTTIME\_SUM*(){  
 var total = 0.0;  
 $('.burst\_time').each(function(index){  
 total += *parseFloat*($(this).val());  
 });  
  
 return (total + *GET\_ARRIVALTIME\_LOWEST*());  
 }  
  
 function *GET\_PR\_WITH\_HIGHEST\_AT\_AND\_BT*(){  
 var procAndBT = null;  
 $('.arrival\_time').each(function(index){ *// check for arrivals* var curr\_arrival\_time = Math.*round*(*parseFloat*($(this).val()));  
 var highest = 0;  
 if(curr\_arrival\_time > highest){  
 highest = curr\_arrival\_time;  
 procAndBT = [$(this).data('process'), *parseFloat*(highest)];  
 }  
 });  
   
 return procAndBT;  
 }  
   
 function *PROCESS\_ARRIVALS*(time){  
 var arrival\_flag = false;  
 $('.arrival\_time').each(function(index){ *// check for arrivals* var curr\_arrival\_time = Math.*round*(*parseFloat*($(this).val()));  
 if(curr\_arrival\_time == *parseFloat*(time)){  
 var process\_number = index+1;  
 var curr\_bursttime = *parseFloat*($('[data-process="'+(process\_number)+'"][class="burst\_time"]').val());  
 *ready\_queue*.push(process\_number+'?'+curr\_bursttime);  
 console.log(time+'\t: PR ARRVD : '+process\_number+'/'+curr\_bursttime+' | '+*ready\_queue*);  
 *SORT\_READY\_QUEUE*();  
 arrival\_flag = true;  
 }  
 });  
 return arrival\_flag;  
 }  
   
 function *GET\_BURSTTIME\_TOTAL*(){  
 var total = 0.0;  
 $('.burst\_time').each(function(index){  
 total += *parseFloat*($(this).val());  
 });  
  
 if(*GET\_ARRIVALTIME\_HIGHEST*() > total){  
 total = *GET\_ARRIVALTIME\_HIGHEST*();  
 }  
  
 return *parseFloat*(total);  
 }  
   
 function *SORT\_READY\_QUEUE*(){  
 *ready\_queue*.sort(function(a,b){ *// sort queue by lowest bt first* return a.split('?')[1] - b.split('?')[1]  
 });  
 }  
 </script>  
 </head>  
 <body>  
 Scheduling Methods  
 <select id="methods">  
 <option value="">SELECT A METHOD</option>  
 <option value="FirstComeFirstServe.html">First Come, First Serve (FCFS)</option>  
 <option value="ShortestJobFirst.html">Shortest Job First</option>  
 <option value="ShortestTimeRemaining.html">Shortest Time Remaining</option>  
 <option value="PriorityNonPreemptive.html">Priority Non-Preemptive</option>  
 <option value="PriorityPreemptive.html">Priority Preemptive</option>  
 <option value="RoundRobin.html">Round Robin</option>  
 </select>  
 <br/>  
 <br/>  
 <h1>Shortest Time Remaining</h1>  
 <table border="1">  
 <thead>  
 <tr>  
 <td>Process Name</td>  
 <td>Arrival Time</td>  
 <td>Burst Time</td>  
 <td>Turn-Around Time</td>  
 <td>Waiting Time</td>  
 </tr>  
 </thead>  
 <tbody>  
 <tr>  
 <td>P1</td>  
 <td><input data-process="1" type="text" class="arrival\_time" /></td>  
 <td><input data-process="1" type="text" class="burst\_time" /></td>  
 <td><span class="TAT" id="P1\_TAT"></span></td>  
 <td><span class="WT" id="P1\_WT"></span></td>  
 </tr>  
 <tr>  
 <td>P2</td>  
 <td><input data-process="2" type="text" class="arrival\_time" /></td>  
 <td><input data-process="2" type="text" class="burst\_time" /></td>  
 <td><span class="TAT" id="P2\_TAT"></span></td>  
 <td><span class="WT" id="P2\_WT"></span></td>  
 </tr>  
 <tr>  
 <td>P3</td>  
 <td><input data-process="3" type="text" class="arrival\_time" /></td>  
 <td><input data-process="3" type="text" class="burst\_time" /></td>  
 <td><span class="TAT" id="P3\_TAT"></span></td>  
 <td><span class="WT" id="P3\_WT"></span></td>  
 </tr>  
 <tr>  
 <td>P4</td>  
 <td><input data-process="4" type="text" class="arrival\_time" /></td>  
 <td><input data-process="4" type="text" class="burst\_time" /></td>  
 <td><span class="TAT" id="P4\_TAT"></span></td>  
 <td><span class="WT" id="P4\_WT"></span></td>  
 </tr>  
 <tr>  
 <td colspan="3">Average</td>  
 <td><span id="AVG\_TAT"></span></td>  
 <td><span id="AVG\_WT"></span></td>  
 </tr>  
 </tbody>  
 </table>  
 <br/>  
 <div style="width: 80%">  
 <div id="gantt\_chart">  
 </div>  
 </div>  
 <div style="clear: both;"></div>  
 <p id="cust\_console" style="color:red;"></p>  
 <button id="INIT\_COMPUTE">Compute</button>  
 </body>  
</html>

**PriorityNonPreemptive.html**

<**html**>  
 <**head**>  
 <**style**>  
 **body** {  
 **font-family** : **Arial**;  
 }  
 **td** {  
 **padding** : 0.3**em**;  
 **text-align** : **center**;  
 }  
 **thead** {  
 **font-weight** : **bold**;  
 }  
 .**gantt\_block** {  
 **text-align**: **center**;  
 **padding**: 0.4**em**;  
 **height**: 5**em**;  
 **float**: **left**;  
 **display**: **block**;  
 **color**: **whitesmoke**;  
 **font-weight**: **bold**;  
 }  
 **table** {  
 **border-collapse** : **collapse**;  
 }  
 .**bubble** {  
 **-webkit-box-shadow**: 0**px** 0**px** 21**px** 1**px rgba**(0,0,0,0.84);  
 **-moz-box-shadow**: 0**px** 0**px** 21**px** 1**px rgba**(0,0,0,0.84);  
 **box-shadow**: 0**px** 0**px** 21**px** 1**px rgba**(0,0,0,0.84);  
 **text-align**: **center**;  
 **padding**: 0.4**em**;  
 **height**: 5**em**;  
 **float**: **left**;  
 **display**: **block**;  
 **color**: **whitesmoke**;  
 **font-weight**: **bold**;  
 }  
 </**style**>  
 <**script src="js/jquery-3.1.0.min.js"**></**script**>  
 <**script**>  
 **var *ready\_queue*** = [];  
 **var *cpu\_process*** = **null**;  
 **var *cpu\_bursttime*** = **null**;  
 **var *GLOBAL\_startTime*** = **null**;  
 **var *GLOBAL\_endTime*** = **null**;  
 **var *GLOBAL\_bubbleStart*** = **null**;  
 **var *GLOBAL\_bubbleEnd*** = **null**;  
 **var *my\_console*** = $(**'#cust\_console'**);  
 **var *my\_gantt\_chart*** = $(**'#gantt\_chart'**);  
 **var *my\_colors*** = [  
 **'#E74C3C'**,  
 **'#8E44AD'**,  
 **'#16A085'**,  
 **'#D35400'** ];  
 **var *i*** = **null**;  
 **var *pr\_done*** = 0;  
  
 $(**document**).ready(**function**(){  
 $(**'#methods'**).change(**function**(){  
 **location**.**href** = $(**this**).val();  
 });  
 $(**'input'**).each(**function**(){  
 $(**this**).val(**Math**.*floor*(**Math**.*random*() \* 10) + 1);  
 });  
 $(**'#INIT\_COMPUTE'**).click(**function** () {  
 **if**(*checkValues*()){ *// check if input is acceptable* ***i*** = *GET\_ARRIVALTIME\_LOWEST*(); *// get lowest arrival time* **do**{ *// used do while - loop only executes when ALL the process are done  
 PROCESS\_ARRIVALS*(***i***);  
 **if**(***cpu\_process*** != **null**){ *// check if the cpu process has no ongoing job* **if**(***i*** == ***GLOBAL\_endTime***){ *// if the time(i) is equal to GLOBAL\_endtime then a process has just ended* **var** curr\_width = (((***GLOBAL\_endTime*** - ***GLOBAL\_startTime***) / *GET\_BURSTTIME\_SUM*()) \* 80); *// compute width for gantt chart* $(**'#gantt\_chart'**).append(**'<div data-process="'**+***cpu\_process***+**'" data-start="'**+***GLOBAL\_startTime***+**'" data-end="'**+***GLOBAL\_endTime***+**'" class="gantt\_block" style="background-color: '**+***my\_colors***[(***cpu\_process***-1)]+**'; width: '**+curr\_width+**'%;">P'**+***cpu\_process***+**'<br/>'**+***GLOBAL\_startTime***+**' - '**+***GLOBAL\_endTime***+**'</div>'**);  
 ***pr\_done***++; *// increment pr\_done for every finished process  
  
 // compute turnaround time and waiting time* **var** curr\_arrivaltime = *parseFloat*($(**'[data-process="'**+***cpu\_process***+**'"][class="arrival\_time"]'**).val());  
 $(**'#P'**+***cpu\_process***+**'\_TAT'**).append(***GLOBAL\_endTime*** - curr\_arrivaltime);  
 $(**'#P'**+***cpu\_process***+**'\_WT'**).append(***GLOBAL\_startTime*** - curr\_arrivaltime);  
 *// reinitialize cpu to null (no job)* ***cpu\_process*** = **null**;  
 ***cpu\_bursttime*** = **null**;  
 }  
 }  
  
 **if**(***cpu\_process*** == **null**){ *// check if cpu has no job* **if**(***ready\_queue***.**length** > 0){ *// check if ready\_queue has a job waiting* **if**(***GLOBAL\_bubbleStart*** != **null**){  
 **var** bubble\_width = ((***i*** - ***GLOBAL\_bubbleStart***) / *GET\_BURSTTIME\_SUM*()) \* 80;  
 $(**'#gantt\_chart'**).append(**'<div class="bubble" style="background-color: white; width: '**+bubble\_width+**'%; border: 1px solid #333333; color: black;">BUBBLE<br/>'**+***GLOBAL\_bubbleStart***+**' - '**+***i***+**'</div>'**);  
 ***GLOBAL\_bubbleStart*** = **null**;  
 }  
 ***cpu\_process*** = ***ready\_queue***[0].split(**'?'**)[0]; *// get and put process from ready\_queue to temp value* ***cpu\_bursttime*** = *parseFloat*(***ready\_queue***[0].split(**'?'**)[1]); *// get burst time of process* ***ready\_queue***.shift(); *// remove contents of index 0 and shift the array* ***GLOBAL\_startTime*** = ***i***; *// initialize start time of process based on time(i)* ***GLOBAL\_endTime*** = ***GLOBAL\_startTime*** + ***cpu\_bursttime***; *// compute the end time for the process by adding the burst time to start time* }**else**{ *// if no job waiting in ready\_queue AND the cpu has no process, BUBBLE occurs* **if**(***GLOBAL\_bubbleStart*** == **null**){ *// initialize bubble counter* ***GLOBAL\_bubbleStart*** = ***i***;  
 }  
 }  
 }  
 ***i***++;  
 **if**(***i*** > 100){  
 **break**;  
 }  
 }**while**(***pr\_done*** < 4);  
  
 *// compute average of turnaround time* **var** total\_TAT = 0;  
 $(**'.TAT'**).each(**function** (index) {  
 total\_TAT += *parseFloat*($(**this**).text());  
 });  
 $(**'#AVG\_TAT'**).empty().append((total\_TAT/4));  
 *// compute average of waiting time* **var** total\_WT = 0;  
 $(**'.WT'**).each(**function** (index) {  
 total\_WT += *parseFloat*($(**this**).text());  
 });  
 $(**'#AVG\_WT'**).empty().append((total\_WT/4));  
 }  
 })  
 });  
  
 **function** *GET\_BURSTTIME\_SUM*(){  
 **var** total = 0.0;  
 $(**'.burst\_time'**).each(**function**(index){  
 total += *parseFloat*($(**this**).val());  
 });  
  
 **return** (total + *GET\_ARRIVALTIME\_LOWEST*());  
 }  
  
 **function** *PROCESS\_ARRIVALS*(time){  
 **var** arrival\_flag = **false**;  
 $(**'.arrival\_time'**).each(**function**(index){ *// check for arrivals* **var** curr\_arrival\_time = **Math**.*round*(*parseFloat*($(**this**).val()));  
 **if**(curr\_arrival\_time == *parseFloat*(time)){  
 **var** process\_number = index+1;  
 **var** curr\_bursttime = *parseFloat*($(**'[data-process="'**+(process\_number)+**'"][class="burst\_time"]'**).val());  
 **var** curr\_prio = *parseFloat*($(**'[data-process="'**+(process\_number)+**'"][class="priority"]'**).val());  
 ***ready\_queue***.push(process\_number+**'?'**+curr\_bursttime+**'?'**+curr\_prio);  
 **console**.log(time+**'\t: PR ARRVD : '**+process\_number+**'/'**+curr\_bursttime+**' | '**+***ready\_queue***);  
 *SORT\_READY\_QUEUE*();  
 arrival\_flag = **true**;  
 }  
 });  
 **return** arrival\_flag;  
 }  
  
 **function** *GET\_ARRIVALTIME\_LOWEST*(){  
 **var** lowest = *GET\_ARRIVALTIME\_HIGHEST*();  
 $(**'.arrival\_time'**).each(**function**(){  
 **if**(*parseFloat*($(**this**).val()) < lowest){  
 lowest = *parseFloat*($(**this**).val());  
 }  
 });  
  
 **return** lowest;  
 }  
  
 **function** *GET\_ARRIVALTIME\_HIGHEST*(){  
 **var** highest = 0;  
 $(**'.arrival\_time'**).each(**function**(){  
 **if**(highest == 0){  
 highest = *parseFloat*($(**this**).val());  
 }  
 **if**(*parseFloat*($(**this**).val()) > highest){  
 highest = *parseFloat*($(**this**).val());  
 }  
 });  
 **return** *parseFloat*(highest);  
 }  
  
 **function** *checkValues*(){  
 **var** flag = **true**;  
 $(**'.arrival\_time'**).each(**function**(index){  
 *// check if arrival\_time is filled out* **if**($(**this**).val() == **''** || !$.isNumeric($(**this**).val())){  
 $(**'#cust\_console'**).append(**'Please input a number for Arrival Time for Process P'**+(index+1)+**'<br/>'**);  
 flag = **false**;  
 }  
 })  
 $(**'.burst\_time'**).each(**function**(index){  
 *// check if burst\_time is filled out* **if**($(**this**).val() == **''** || !$.isNumeric($(**this**).val())){  
 $(**'#cust\_console'**).append(**'Please input a number for Burst Time for Process P'**+(index+1)+**'<br/>'**);  
 flag = **false**;  
 }  
 })  
 $(**'.priority'**).each(**function**(index){  
 *// check if priority is filled out* **if**($(**this**).val() == **''** || !$.isNumeric($(**this**).val())){  
 $(**'#cust\_console'**).append(**'Please input a number for Priority for Process P'**+(index+1)+**'<br/>'**);  
 flag = **false**;  
 }  
 })  
  
 **return** flag;  
 }  
  
 **function** *SORT\_READY\_QUEUE*(){  
 ***ready\_queue***.sort(**function**(a,b){ *// sort queue by lowest priority first* **return** a.split(**'?'**)[2] - b.split(**'?'**)[2]  
 });  
 }  
 </**script**>  
 </**head**>  
 <**body**>  
 Scheduling Methods  
 <**select id="methods"**>  
 <**option value=""**>SELECT A METHOD</**option**>  
 <**option value="FirstComeFirstServe.html"**>First Come, First Serve (FCFS)</**option**>  
 <**option value="ShortestJobFirst.html"**>Shortest Job First</**option**>  
 <**option value="ShortestTimeRemaining.html"**>Shortest Time Remaining</**option**>  
 <**option value="PriorityNonPreemptive.html"**>Priority Non-Preemptive</**option**>  
 <**option value="PriorityPreemptive.html"**>Priority Preemptive</**option**>  
 <**option value="RoundRobin.html"**>Round Robin</**option**>  
 </**select**>  
 <**br**/>  
 <**br**/>  
 <**h1**>Priority Non-Preemptive</**h1**>  
 <**table border="1"**>  
 <**thead**>  
 <**tr**>  
 <**td**>Process Name</**td**>  
 <**td**>Arrival Time</**td**>  
 <**td**>Burst Time</**td**>  
 <**td**>Priority</**td**>  
 <**td**>Turn-Around Time</**td**>  
 <**td**>Waiting Time</**td**>  
 </**tr**>  
 </**thead**>  
 <**tbody**>  
 <**tr**>  
 <**td**>P1</**td**>  
 <**td**><**input data-process="1" type="text" class="arrival\_time"** /></**td**>  
 <**td**><**input data-process="1" type="text" class="burst\_time"** /></**td**>  
 <**td**><**input data-process="1" type="text" class="priority"** /></**td**>  
 <**td**><**span class="TAT" id="P1\_TAT"**></**span**></**td**>  
 <**td**><**span class="WT" id="P1\_WT"**></**span**></**td**>  
 </**tr**>  
 <**tr**>  
 <**td**>P2</**td**>  
 <**td**><**input data-process="2" type="text" class="arrival\_time"** /></**td**>  
 <**td**><**input data-process="2" type="text" class="burst\_time"** /></**td**>  
 <**td**><**input data-process="2" type="text" class="priority"** /></**td**>  
 <**td**><**span class="TAT" id="P2\_TAT"**></**span**></**td**>  
 <**td**><**span class="WT" id="P2\_WT"**></**span**></**td**>  
 </**tr**>  
 <**tr**>  
 <**td**>P3</**td**>  
 <**td**><**input data-process="3" type="text" class="arrival\_time"** /></**td**>  
 <**td**><**input data-process="3" type="text" class="burst\_time"** /></**td**>  
 <**td**><**input data-process="3" type="text" class="priority"** /></**td**>  
 <**td**><**span class="TAT" id="P3\_TAT"**></**span**></**td**>  
 <**td**><**span class="WT" id="P3\_WT"**></**span**></**td**>  
 </**tr**>  
 <**tr**>  
 <**td**>P4</**td**>  
 <**td**><**input data-process="4" type="text" class="arrival\_time"** /></**td**>  
 <**td**><**input data-process="4" type="text" class="burst\_time"** /></**td**>  
 <**td**><**input data-process="4" type="text" class="priority"** /></**td**>  
 <**td**><**span class="TAT" id="P4\_TAT"**></**span**></**td**>  
 <**td**><**span class="WT" id="P4\_WT"**></**span**></**td**>  
 </**tr**>  
 <**tr**>  
 <**td colspan="4"**>Average</**td**>  
 <**td**><**span id="AVG\_TAT"**></**span**></**td**>  
 <**td**><**span id="AVG\_WT"**></**span**></**td**>  
 </**tr**>  
 </**tbody**>  
 </**table**>  
 <**br**/>  
 <**div style="width**: 80%**"**>  
 <**div id="gantt\_chart"**>  
 </**div**>  
 </**div**>  
 <**div style="clear**: **both**;**"**></**div**>  
 <**p id="cust\_console" style="color**:**red**;**"**></**p**>  
 <**button id="INIT\_COMPUTE"**>Compute</**button**>  
 </**body**>  
</**html**>

**PriorityPreemptive.html**

<**html**>  
<**head**>  
 <**style**>  
 **body** {  
 **font-family** : **Arial**;  
 }  
 **td** {  
 **padding** : 0.3**em**;  
 **text-align** : **center**;  
 }  
 **thead** {  
 **font-weight** : **bold**;  
 }  
 .**gantt\_block** {  
 **text-align**: **center**;  
 **padding**: 0.4**em**;  
 **height**: 5**em**;  
 **float**: **left**;  
 **display**: **block**;  
 **color**: **whitesmoke**;  
 **font-weight**: **bold**;  
 }  
 **table** {  
 **border-collapse** : **collapse**;  
 }  
 .**bubble** {  
 **-webkit-box-shadow**: 0**px** 0**px** 21**px** 1**px rgba**(0,0,0,0.84);  
 **-moz-box-shadow**: 0**px** 0**px** 21**px** 1**px rgba**(0,0,0,0.84);  
 **box-shadow**: 0**px** 0**px** 21**px** 1**px rgba**(0,0,0,0.84);  
 **text-align**: **center**;  
 **padding**: 0.4**em**;  
 **height**: 5**em**;  
 **float**: **left**;  
 **display**: **block**;  
 **color**: **whitesmoke**;  
 **font-weight**: **bold**;  
 }  
 </**style**>  
 <**script src="js/jquery-3.1.0.min.js"**></**script**>  
  
 <**script**>  
 **var *ready\_queue*** = [];  
 **var *cpu\_process*** = **null**;  
 **var *cpu\_bursttime*** = **null**;  
 **var *cpu\_priority*** = **null**;  
 **var *GLOBAL\_startTime*** = **null**;  
 **var *GLOBAL\_endTime*** = **null**;  
 **var *GLOBAL\_bubbleStart*** = **null**;  
 **var *GLOBAL\_bubbleEnd*** = **null**;  
 **var *my\_console*** = $(**'#cust\_console'**);  
 **var *my\_gantt\_chart*** = $(**'#gantt\_chart'**);  
 **var *my\_colors*** = [  
 **'#E74C3C'**,  
 **'#8E44AD'**,  
 **'#16A085'**,  
 **'#D35400'** ];  
 **var *pr\_done*** = 0;  
  
 $(**document**).ready(**function**(){  
  
 $(**'input'**).each(**function**(){  
 $(**this**).val(**Math**.*floor*(**Math**.*random*() \* 15) + 1);  
 });  
  
 $(**'#INIT\_COMPUTE'**).click(**function**(){  
 **if**(*checkValues*()){  
 **var** i = *GET\_ARRIVALTIME\_LOWEST*();  
 **do**{  
 *PROCESS\_ARRIVALS*(i); *// insert process to `ready\_queue`* **if**(***cpu\_process*** != **null**){  
 ***cpu\_bursttime***--; *// decrement value* **console**.log(i+**'\t: DCRMNT BT : P'**+***cpu\_process***+**'/'**+***cpu\_bursttime***);  
  
 **if**(***cpu\_bursttime*** == 0){ *// cpu has 0 bt value* ***GLOBAL\_endTime*** = i; *// change end time to loop value `i`* **var** bt\_of\_process = *parseFloat*(***GLOBAL\_endTime*** - ***GLOBAL\_startTime***);  
 **var** curr\_width = ((bt\_of\_process / *GET\_BURSTTIME\_SUM*()) \* 80); *// compute width for gantt chart* $(**'#gantt\_chart'**).append(**'<div data-process="'**+***cpu\_process***+**'" data-start="'**+***GLOBAL\_startTime***+**'" data-end="'**+***GLOBAL\_endTime***+**'" class="gantt\_block" style="background-color: '**+***my\_colors***[(***cpu\_process***-1)]+**'; width: '**+curr\_width+**'%;">P'**+***cpu\_process***+**'<br/>'**+***GLOBAL\_startTime***+**' - '**+***GLOBAL\_endTime***+**'</div>'**);  
 **console**.log(i+**'\t: '**+**' ADD\_GANTT\_1 = '**+***cpu\_process***+**'/'**+***cpu\_bursttime***);  
 ***cpu\_process*** = **null**;  
 ***cpu\_bursttime*** = **null**;  
 ***pr\_done***++;  
 }  
 }  
  
 **if**(***cpu\_process*** == **null**){  
 **if**(***ready\_queue***.**length** > 0){*SORT\_READY\_QUEUE*(); *// sort ready queue for smallest process first* ***cpu\_process*** = ***ready\_queue***[0].split(**'?'**)[0]; *// add process to cpu* ***cpu\_bursttime*** = ***ready\_queue***[0].split(**'?'**)[1];  
 ***cpu\_priority*** = *parseFloat*(***ready\_queue***[0].split(**'?'**)[2]);  
 ***ready\_queue***.shift();  
 ***GLOBAL\_startTime*** = i;  
  
 **if**(***GLOBAL\_bubbleStart*** != **null**){  
 **console**.log(i+**'\t: BUBBLE END'**);  
 **var** bubble\_width = ((i - ***GLOBAL\_bubbleStart***) / *GET\_BURSTTIME\_SUM*()) \* 80;  
 $(**'#gantt\_chart'**).append(**'<div class="bubble" style="background-color: white; width: '**+bubble\_width+**'%; border: 1px solid #333333; color: black;">BUBBLE<br/>'**+***GLOBAL\_bubbleStart***+**' - '**+i+**'</div>'**);  
 ***GLOBAL\_bubbleStart*** = **null**;  
 }  
  
 **console**.log(i+**'\t: ADD PR to CPU : P'**+***cpu\_process***+**'/'**+***cpu\_bursttime***);  
 }**else**{ *// bubble* **if**(***GLOBAL\_bubbleStart*** == **null**){  
 ***GLOBAL\_bubbleStart*** = i;  
 }  
 **console**.log(i+**'\t: BUBBLE INC - Start = '**+***GLOBAL\_bubbleStart***);  
 }  
 }**else**{  
 *// algo for change process - PREEMPTIVE NATURE* **if**(***ready\_queue***.**length** > 0){  
 *SORT\_READY\_QUEUE*();  
 **var** og\_temp\_pr = ***ready\_queue***[0].split(**'?'**)[0]; *// get and put process from ready\_queue to temp value* **var** og\_temp\_bt = ***ready\_queue***[0].split(**'?'**)[1]; *// get and put burst time from ready\_queue to temp value* **var** og\_temp\_prio = *parseFloat*(***ready\_queue***[0].split(**'?'**)[2]); *// get and put priority value from ready\_queue to temp value* **console**.log(i+**'\t:'**+og\_temp\_prio+**' < '**+***cpu\_priority***+**' = '**+(og\_temp\_prio <= ***cpu\_priority***));  
 **if**(og\_temp\_prio < ***cpu\_priority***){ *// determines if cpu will change to lower bt process* ***GLOBAL\_endTime*** = i; *// change end time to loop value `i`* **var** bt\_of\_process = *parseFloat*(***GLOBAL\_endTime*** - ***GLOBAL\_startTime***); *// get burst time of process* **var** curr\_width = ((bt\_of\_process / *GET\_BURSTTIME\_SUM*()) \* 80); *// compute width for gantt chart* $(**'#gantt\_chart'**).append(**'<div data-process="'**+***cpu\_process***+**'" data-start="'**+***GLOBAL\_startTime***+**'" data-end="'**+***GLOBAL\_endTime***+**'" class="gantt\_block" style="background-color: '**+***my\_colors***[(***cpu\_process***-1)]+**'; width: '**+curr\_width+**'%;">P'**+***cpu\_process***+**'<br/>'**+***GLOBAL\_startTime***+**' - '**+***GLOBAL\_endTime***+**'</div>'**);  
 **console**.log(i+**'\t: '**+**' ADD\_GANTT\_2 = '**+***cpu\_process***+**'/'**+***cpu\_bursttime***);  
  
 *// change process* **console**.log(i+**'\t: CHNG PR : P'**+***cpu\_process***+**'/'**+***cpu\_bursttime***+**' to '**+og\_temp\_pr+**'/'**+og\_temp\_bt);  
 ***ready\_queue***.push(***cpu\_process***+**'?'**+***cpu\_bursttime***+**'?'**+***cpu\_priority***); *// put back process from cpu to ready\_queue* ***cpu\_process*** = og\_temp\_pr; *// apply new cpu process* ***cpu\_bursttime*** = og\_temp\_bt; *// apply new cpu burst time* ***cpu\_priority*** = og\_temp\_prio; *// apply new cpu burst time* ***ready\_queue***.shift(); *// remove applied process from ready\_queue  
 SORT\_READY\_QUEUE*(); *// sort ready\_queue for safety* ***GLOBAL\_startTime*** = i;  
 }  
 }  
 }  
 i++;  
 }**while**(***pr\_done*** < 4);  
  
 **var** et\_array\_p = [];  
 **var** et\_array\_e = [];  
 $(**'.gantt\_block'**).each(**function** (index) { *// loop through the items of the gantt chart  
 // put process details in temporary variables* **var** tmp\_process = *parseFloat*($(**this**).data(**'process'**));  
 **var** tmp\_start = *parseFloat*($(**this**).data(**'start'**));  
 **var** tmp\_end = *parseFloat*($(**this**).data(**'end'**));  
 **var** tmp\_arrival = *parseFloat*($(**'[data-process="'**+(tmp\_process)+**'"][class="arrival\_time"]'**).val());  
 *// selectors* **var** slctr\_tat = $(**'#P'**+tmp\_process+**'\_TAT'**);  
 **var** slctr\_wt = $(**'#P'**+tmp\_process+**'\_WT'**);  
 **var** inArray = $.inArray(tmp\_process, et\_array\_p);  
  
 *// compute and append the turnaround time* slctr\_tat.empty().append(tmp\_end - tmp\_arrival);  
  
 *// compute and append waiting time* **var** curr\_wt = slctr\_wt.text();  
 **if**(inArray > -1){  
 slctr\_wt.empty().append(*parseFloat*(curr\_wt) + (tmp\_start - et\_array\_e[inArray]));  
 et\_array\_e[inArray] = tmp\_end;  
 }**else**{  
 slctr\_wt.append(tmp\_start - tmp\_arrival);  
 et\_array\_p.push(tmp\_process);  
 et\_array\_e.push(tmp\_end);  
 }  
 });  
  
 *// compute average of turnaround time* **var** total\_tat = 0;  
 $(**'.TAT'**).each(**function** (index) {  
 total\_tat += *parseFloat*($(**this**).text());  
 });  
 $(**'#AVG\_TAT'**).empty().append((*parseFloat*(total\_tat)/$(**'.TAT'**).**length**));  
  
 *// compute average of waiting time* **var** total\_wt = 0;  
 $(**'.WT'**).each(**function** (index) {  
 total\_wt += *parseFloat*($(**this**).text());  
 });  
 $(**'#AVG\_WT'**).empty().append((*parseFloat*(total\_wt)/$(**'.WT'**).**length**));  
 }  
 });  
  
 $(**'#methods'**).change(**function**(){  
 **location**.**href** = $(**this**).val();  
 })  
 });  
  
 **function** *checkValues*(){  
 **var** flag = **true**;  
 $(**'#cust\_console'**).empty();  
 $(**'.arrival\_time'**).each(**function**(index){  
 *// check if arrival\_time is filled out* **if**($(**this**).val() == **''** || !$.isNumeric($(**this**).val())){  
 $(**'#cust\_console'**).append(**'Please input a number for Arrival Time for Process P'**+(index+1)+**'<br/>'**);  
 flag = **false**;  
 }  
 })  
 $(**'.burst\_time'**).each(**function**(index){  
 *// check if burst\_time is filled out* **if**($(**this**).val() == **''** || !$.isNumeric($(**this**).val())){  
 $(**'#cust\_console'**).append(**'Please input a number for Burst Time for Process P'**+(index+1)+**'<br/>'**);  
 flag = **false**;  
 }  
 })  
 $(**'.priority'**).each(**function**(index){  
 *// check if burst\_time is filled out* **if**($(**this**).val() == **''** || !$.isNumeric($(**this**).val())){  
 $(**'#cust\_console'**).append(**'Please input a number for Priority for Process P'**+(index+1)+**'<br/>'**);  
 flag = **false**;  
 }  
 })  
  
 **return** flag;  
 }  
  
 **function** *GET\_BT\_OF\_PROCESS*(cpu){  
 **return** *parseFloat*(**Math**.*round*($(**'[data-process="'**+(cpu)+**'"][class="burst\_time"]'**).val()));  
 }  
  
 **function** *GET\_ARRIVALTIME\_LOWEST*(){  
 **var** lowest = *GET\_ARRIVALTIME\_HIGHEST*();  
 $(**'.arrival\_time'**).each(**function**(){  
 **if**(*parseFloat*($(**this**).val()) < lowest){  
 lowest = *parseFloat*($(**this**).val());  
 }  
 });  
  
 **return** lowest;  
 }  
  
 **function** *GET\_PROCESSTIME*(){  
  
 }  
  
 **function** *GET\_ARRIVALTIME\_HIGHEST*(){  
 **var** highest = 0;  
 $(**'.arrival\_time'**).each(**function**(){  
 **if**(highest == 0){  
 highest = *parseFloat*($(**this**).val());  
 }  
 **if**(*parseFloat*($(**this**).val()) > highest){  
 highest = *parseFloat*($(**this**).val());  
 }  
 });  
 **return** *parseFloat*(highest);  
 }  
  
 **function** *GET\_BURSTTIME\_SUM*(){  
 **var** total = 0.0;  
 $(**'.burst\_time'**).each(**function**(index){  
 total += *parseFloat*($(**this**).val());  
 });  
  
 **return** (total + *GET\_ARRIVALTIME\_LOWEST*());  
 }  
  
 **function** *GET\_PR\_WITH\_HIGHEST\_AT\_AND\_BT*(){  
 **var** procAndBT = **null**;  
 $(**'.arrival\_time'**).each(**function**(index){ *// check for arrivals* **var** curr\_arrival\_time = **Math**.*round*(*parseFloat*($(**this**).val()));  
 **var** highest = 0;  
 **if**(curr\_arrival\_time > highest){  
 highest = curr\_arrival\_time;  
 procAndBT = [$(**this**).data(**'process'**), *parseFloat*(highest)];  
 }  
 });  
  
 **return** procAndBT;  
 }  
  
 **function** *PROCESS\_ARRIVALS*(time){  
 **var** arrival\_flag = **false**;  
 $(**'.arrival\_time'**).each(**function**(index){ *// check for arrivals* **var** curr\_arrival\_time = **Math**.*round*(*parseFloat*($(**this**).val()));  
 **if**(curr\_arrival\_time == *parseFloat*(time)){  
 **var** process\_number = index+1;  
 **var** curr\_bursttime = *parseFloat*($(**'[data-process="'**+(process\_number)+**'"][class="burst\_time"]'**).val());  
 **var** curr\_prio = *parseFloat*($(**'[data-process="'**+(process\_number)+**'"][class="priority"]'**).val());  
 ***ready\_queue***.push(process\_number+**'?'**+curr\_bursttime+**'?'**+curr\_prio);  
 **console**.log(time+**'\t: PR ARRVD : '**+process\_number+**'/'**+curr\_bursttime+**' | '**+***ready\_queue***);  
 *SORT\_READY\_QUEUE*();  
 arrival\_flag = **true**;  
 }  
 });  
 **return** arrival\_flag;  
 }  
  
 **function** *GET\_BURSTTIME\_TOTAL*(){  
 **var** total = 0.0;  
 $(**'.burst\_time'**).each(**function**(index){  
 total += *parseFloat*($(**this**).val());  
 });  
  
 **if**(*GET\_ARRIVALTIME\_HIGHEST*() > total){  
 total = *GET\_ARRIVALTIME\_HIGHEST*();  
 }  
  
 **return** *parseFloat*(total);  
 }  
  
 **function** *SORT\_READY\_QUEUE*(){  
 ***ready\_queue***.sort(**function**(a,b){ *// sort queue by lowest bt first* **return** a.split(**'?'**)[2] - b.split(**'?'**)[2]  
 });  
 }  
 </**script**>  
</**head**>  
<**body**>  
Scheduling Methods  
<**select id="methods"**>  
 <**option value=""**>SELECT A METHOD</**option**>  
 <**option value="FirstComeFirstServe.html"**>First Come, First Serve (FCFS)</**option**>  
 <**option value="ShortestJobFirst.html"**>Shortest Job First</**option**>  
 <**option value="ShortestTimeRemaining.html"**>Shortest Time Remaining</**option**>  
 <**option value="PriorityNonPreemptive.html"**>Priority Non-Preemptive</**option**>  
 <**option value="PriorityPreemptive.html"**>Priority Preemptive</**option**>  
 <**option value="RoundRobin.html"**>Round Robin</**option**>  
</**select**>  
<**br**/>  
<**br**/>  
<**h1**>Priority Preemptive</**h1**>  
<**table border="1"**>  
 <**thead**>  
 <**tr**>  
 <**td**>Process Name</**td**>  
 <**td**>Arrival Time</**td**>  
 <**td**>Burst Time</**td**>  
 <**td**>Priority</**td**>  
 <**td**>Turn-Around Time</**td**>  
 <**td**>Waiting Time</**td**>  
 </**tr**>  
 </**thead**>  
 <**tbody**>  
 <**tr**>  
 <**td**>P1</**td**>  
 <**td**><**input data-process="1" type="text" class="arrival\_time"** /></**td**>  
 <**td**><**input data-process="1" type="text" class="burst\_time"** /></**td**>  
 <**td**><**input data-process="1" type="text" class="priority"** /></**td**>  
 <**td**><**span class="TAT" id="P1\_TAT"**></**span**></**td**>  
 <**td**><**span class="WT" id="P1\_WT"**></**span**></**td**>  
 </**tr**>  
 <**tr**>  
 <**td**>P2</**td**>  
 <**td**><**input data-process="2" type="text" class="arrival\_time"** /></**td**>  
 <**td**><**input data-process="2" type="text" class="burst\_time"** /></**td**>  
 <**td**><**input data-process="2" type="text" class="priority"** /></**td**>  
 <**td**><**span class="TAT" id="P2\_TAT"**></**span**></**td**>  
 <**td**><**span class="WT" id="P2\_WT"**></**span**></**td**>  
 </**tr**>  
 <**tr**>  
 <**td**>P3</**td**>  
 <**td**><**input data-process="3" type="text" class="arrival\_time"** /></**td**>  
 <**td**><**input data-process="3" type="text" class="burst\_time"** /></**td**>  
 <**td**><**input data-process="3" type="text" class="priority"** /></**td**>  
 <**td**><**span class="TAT" id="P3\_TAT"**></**span**></**td**>  
 <**td**><**span class="WT" id="P3\_WT"**></**span**></**td**>  
 </**tr**>  
 <**tr**>  
 <**td**>P4</**td**>  
 <**td**><**input data-process="4" type="text" class="arrival\_time"** /></**td**>  
 <**td**><**input data-process="4" type="text" class="burst\_time"** /></**td**>  
 <**td**><**input data-process="4" type="text" class="priority"** /></**td**>  
 <**td**><**span class="TAT" id="P4\_TAT"**></**span**></**td**>  
 <**td**><**span class="WT" id="P4\_WT"**></**span**></**td**>  
 </**tr**>  
 <**tr**>  
 <**td colspan="4"**>Average</**td**>  
 <**td**><**span id="AVG\_TAT"**></**span**></**td**>  
 <**td**><**span id="AVG\_WT"**></**span**></**td**>  
 </**tr**>  
 </**tbody**>  
</**table**>  
<**br**/>  
<**div style="width**: 80%**"**>  
 <**div id="gantt\_chart"**>  
 </**div**>  
</**div**>  
<**div style="clear**: **both**;**"**></**div**>  
<**p id="cust\_console" style="color**:**red**;**"**></**p**>  
<**button id="INIT\_COMPUTE"**>Compute</**button**>  
</**body**>  
</**html**>

**RoundRobin.html**

<**html**>  
<**head**>  
<**style**>  
 **body** {  
 **font-family** : **Arial**;  
 }  
 **td** {  
 **padding** : 0.3**em**;  
 **text-align** : **center**;  
 }  
 **thead** {  
 **font-weight** : **bold**;  
 }  
 .**gantt\_block** {  
 **text-align**: **center**;  
 **padding**: 0.4**em**;  
 **height**: 5**em**;  
 **float**: **left**;  
 **display**: **block**;  
 **color**: **whitesmoke**;  
 **font-weight**: **bold**;  
 }  
 **table** {  
 **border-collapse** : **collapse**;  
 }  
 .**bubble** {  
 **-webkit-box-shadow**: 0**px** 0**px** 21**px** 1**px rgba**(0,0,0,0.84);  
 **-moz-box-shadow**: 0**px** 0**px** 21**px** 1**px rgba**(0,0,0,0.84);  
 **box-shadow**: 0**px** 0**px** 21**px** 1**px rgba**(0,0,0,0.84);  
 **text-align**: **center**;  
 **padding**: 0.4**em**;  
 **height**: 5**em**;  
 **float**: **left**;  
 **display**: **block**;  
 **color**: **whitesmoke**;  
 **font-weight**: **bold**;  
 }  
</**style**>  
<**script src="js/jquery-3.1.0.min.js"**></**script**>  
<**script**>  
**var *ready\_queue*** = [];  
**var *cpu\_process*** = **null**;  
**var *cpu\_bursttime*** = **null**;  
**var *GLOBAL\_startTime*** = **null**;  
**var *GLOBAL\_endTime*** = **null**;  
**var *GLOBAL\_bubbleStart*** = **null**;  
**var *GLOBAL\_bubbleEnd*** = **null**;  
**var *my\_console*** = $(**'#cust\_console'**);  
**var *my\_gantt\_chart*** = $(**'#gantt\_chart'**);  
**var *my\_colors*** = [  
 **'#E74C3C'**,  
 **'#8E44AD'**,  
 **'#16A085'**,  
 **'#D35400'**];  
**var *pr\_done*** = 0;  
**var *quantum*** = 0;  
**var *burst\_counter*** = 0;  
$(**document**).ready(**function**(){  
  
 $(**'input'**).each(**function**(){  
 $(**this**).val(**Math**.*floor*(**Math**.*random*() \* 10) + 1);  
 });  
 $(**'#quantum'**).val(**Math**.*floor*(**Math**.*random*() \* 5) + 1);  
  
 $(**'#INIT\_COMPUTE'**).click(**function**(){  
 **if**(*checkValues*()){  
 **var** i = *GET\_ARRIVALTIME\_LOWEST*();  
 ***quantum*** = $(**'#quantum'**).val();  
  
 **do**{ *// do-while loop | loop doesn't end until ALL PROCESS ARE DONE  
 PROCESS\_ARRIVALS*(i); *// function to insert process to `ready\_queue`* **if**(***cpu\_process*** != **null**){ *// checks if cpu an ongoing job* ***cpu\_bursttime***--; *// decrement burst time value* ***burst\_counter***++; *// increment burst counter* **if**(***burst\_counter*** == ***quantum*** || ***cpu\_bursttime*** == 0){ *// if cpu has 0 bt value left and if burst\_counter is equal to Q (quantum)* ***GLOBAL\_endTime*** = i; *// change end time to loop value `i`* **var** bt\_of\_process = *parseFloat*(***GLOBAL\_endTime*** - ***GLOBAL\_startTime***); *// put burst time into a temporary value* **var** curr\_width = ((bt\_of\_process / *GET\_BURSTTIME\_SUM*()) \* 80); *// compute width for gantt chart* $(**'#gantt\_chart'**).append(**'<div data-process="'**+***cpu\_process***+**'" data-start="'**+***GLOBAL\_startTime***+**'" data-end="'**+***GLOBAL\_endTime***+**'" class="gantt\_block" style="background-color: '**+***my\_colors***[(***cpu\_process***-1)]+**'; width: '**+curr\_width+**'%;">P'**+***cpu\_process***+**'<br/>'**+***GLOBAL\_startTime***+**' - '**+***GLOBAL\_endTime***+**'</div>'**);  
 **if**(***cpu\_bursttime*** > 0){ *// check if there is cpu burst remaining* ***ready\_queue***.push(***cpu\_process***+**'?'**+***cpu\_bursttime***); *// put back process to ready\_queue with decreased burst time left* }**else if**(***cpu\_bursttime*** == 0){  
 ***pr\_done***++; *// indicate a finished process* }  
 *// re-initialize cpu to null (no job)* ***cpu\_process*** = **null**;  
 ***cpu\_bursttime*** = **null**;  
  
 *// re-initialize starttime to time(i) and burst\_counter to 0* ***GLOBAL\_startTime*** = i;  
 ***burst\_counter*** = 0;  
 }  
 }  
  
 **if**(***cpu\_process*** == **null**){ *// check if cpu has no ongoing job* **if**(***ready\_queue***.**length** > 0){ *// check if there is a job waiting in ready\_queue* ***cpu\_process*** = ***ready\_queue***[0].split(**'?'**)[0]; *// add process to cpu* ***cpu\_bursttime*** = ***ready\_queue***[0].split(**'?'**)[1]; *// add burst time to cpu* ***ready\_queue***.shift(); *// shift ready\_queue array removing the 0 index* ***GLOBAL\_startTime*** = i; *// initialize start time to correspoding time(i) which is also the arrival time* ***burst\_counter*** = 0;  
  
 **if**(***GLOBAL\_bubbleStart*** != **null**){ *// check if bubble time has value (this indicates the END of the bubble)* **var** bubble\_width = ((i - ***GLOBAL\_bubbleStart***) / *GET\_BURSTTIME\_SUM*()) \* 80; *// bubble width* $(**'#gantt\_chart'**).append(**'<div class="bubble" style="background-color: white; width: '**+bubble\_width+**'%; border: 1px solid #333333; color: black;">BUBBLE<br/>'**+***GLOBAL\_bubbleStart***+**' - '**+i+**'</div>'**);  
 ***GLOBAL\_bubbleStart*** = **null**; *// re-initialize bubble timer to null (indicating no BUBBLE)* }  
 }**else**{ *// initialize bubble value* **if**(***GLOBAL\_bubbleStart*** == **null**){  
 ***GLOBAL\_bubbleStart*** = i;  
 }  
 }  
 }  
 i++;  
 }**while**(***pr\_done*** < 4);  
  
 **var** et\_array\_p = [];  
 **var** et\_array\_e = [];  
 $(**'.gantt\_block'**).each(**function** (index) {*// loop through the items of the gantt chart  
 // put process details in temporary variables* **var** tmp\_process = *parseFloat*($(**this**).data(**'process'**));  
 **var** tmp\_start = *parseFloat*($(**this**).data(**'start'**));  
 **var** tmp\_end = *parseFloat*($(**this**).data(**'end'**));  
 **var** tmp\_arrival = *parseFloat*($(**'[data-process="'**+(tmp\_process)+**'"][class="arrival\_time"]'**).val());  
 *// selectors* **var** slctr\_tat = $(**'#P'**+tmp\_process+**'\_TAT'**);  
 **var** slctr\_wt = $(**'#P'**+tmp\_process+**'\_WT'**);  
 **var** inArray = $.inArray(tmp\_process, et\_array\_p);  
  
 *// compute and append the turnaround time* slctr\_tat.empty().append(tmp\_end - tmp\_arrival);  
  
 *// compute and append waiting time* **var** curr\_wt = slctr\_wt.text();  
 **if**(inArray > -1){  
 slctr\_wt.empty().append(*parseFloat*(curr\_wt) + (tmp\_start - et\_array\_e[inArray]));  
 et\_array\_e[inArray] = tmp\_end;  
 }**else**{  
 slctr\_wt.append(tmp\_start - tmp\_arrival);  
 et\_array\_p.push(tmp\_process);  
 et\_array\_e.push(tmp\_end);  
 }  
 });  
  
 **var** total\_tat = 0;  
 $(**'.TAT'**).each(**function** (index) {  
 total\_tat += *parseFloat*($(**this**).text());  
 });  
 $(**'#AVG\_TAT'**).empty().append((*parseFloat*(total\_tat)/$(**'.TAT'**).**length**));  
  
 **var** total\_wt = 0;  
 $(**'.WT'**).each(**function** (index) {  
 total\_wt += *parseFloat*($(**this**).text());  
 });  
 $(**'#AVG\_WT'**).empty().append((*parseFloat*(total\_wt)/$(**'.WT'**).**length**));  
 }  
 });  
  
 $(**'#methods'**).change(**function**(){  
 **location**.**href** = $(**this**).val();  
 })  
});  
  
**function** *checkValues*(){  
 **var** flag = **true**;  
 $(**'#cust\_console'**).empty();  
 $(**'.arrival\_time'**).each(**function**(index){  
 *// check if arrival\_time is filled out* **if**($(**this**).val() == **''** || !$.isNumeric($(**this**).val())){  
 $(**'#cust\_console'**).append(**'Please input a number for Arrival Time for Process P'**+(index+1)+**'<br/>'**);  
 flag = **false**;  
 }  
 })  
 $(**'.burst\_time'**).each(**function**(index){  
 *// check if burst\_time is filled out* **if**($(**this**).val() == **''** || !$.isNumeric($(**this**).val())){  
 $(**'#cust\_console'**).append(**'Please input a number for Burst Time for Process P'**+(index+1)+**'<br/>'**);  
 flag = **false**;  
 }  
 })  
 $(**'.priority'**).each(**function**(index){  
 *// check if burst\_time is filled out* **if**($(**this**).val() == **''** || !$.isNumeric($(**this**).val())){  
 $(**'#cust\_console'**).append(**'Please input a number for Priority for Process P'**+(index+1)+**'<br/>'**);  
 flag = **false**;  
 }  
 })  
  
 **switch** ($(**'#quantum'**).val()){  
 **case ''** :  
 **case '0'** :  
 $(**'#cust\_console'**).append(**'Please input a number Q (Quantum)<br/>'**);  
 flag = **false**;  
 **break**;  
 **default** :  
 **break**;  
 }  
  
 **return** flag;  
}  
  
**function** *GET\_BT\_OF\_PROCESS*(cpu){  
 **return** *parseFloat*(**Math**.*round*($(**'[data-process="'**+(cpu)+**'"][class="burst\_time"]'**).val()));  
}  
  
**function** *GET\_ARRIVALTIME\_LOWEST*(){  
 **var** lowest = *GET\_ARRIVALTIME\_HIGHEST*();  
 $(**'.arrival\_time'**).each(**function**(){  
 **if**(*parseFloat*($(**this**).val()) < lowest){  
 lowest = *parseFloat*($(**this**).val());  
 }  
 });  
  
 **return** lowest;  
}  
  
**function** *GET\_PROCESSTIME*(){  
  
}  
  
**function** *GET\_ARRIVALTIME\_HIGHEST*(){  
 **var** highest = 0;  
 $(**'.arrival\_time'**).each(**function**(){  
 **if**(highest == 0){  
 highest = *parseFloat*($(**this**).val());  
 }  
 **if**(*parseFloat*($(**this**).val()) > highest){  
 highest = *parseFloat*($(**this**).val());  
 }  
 });  
 **return** *parseFloat*(highest);  
}  
  
**function** *GET\_BURSTTIME\_SUM*(){  
 **var** total = 0.0;  
 $(**'.burst\_time'**).each(**function**(index){  
 total += *parseFloat*($(**this**).val());  
 });  
  
 **return** (total + *GET\_ARRIVALTIME\_LOWEST*());  
}  
  
**function** *GET\_PR\_WITH\_HIGHEST\_AT\_AND\_BT*(){  
 **var** procAndBT = **null**;  
 $(**'.arrival\_time'**).each(**function**(index){ *// check for arrivals* **var** curr\_arrival\_time = **Math**.*round*(*parseFloat*($(**this**).val()));  
 **var** highest = 0;  
 **if**(curr\_arrival\_time > highest){  
 highest = curr\_arrival\_time;  
 procAndBT = [$(**this**).data(**'process'**), *parseFloat*(highest)];  
 }  
 });  
  
 **return** procAndBT;  
}  
  
**function** *PROCESS\_ARRIVALS*(time){  
 **var** arrival\_flag = **false**;  
 $(**'.arrival\_time'**).each(**function**(index){ *// check for arrivals* **var** curr\_arrival\_time = **Math**.*round*(*parseFloat*($(**this**).val()));  
 **if**(curr\_arrival\_time == *parseFloat*(time)){  
 **var** process\_number = index+1;  
 **var** curr\_bursttime = *parseFloat*($(**'[data-process="'**+(process\_number)+**'"][class="burst\_time"]'**).val());  
 ***ready\_queue***.push(process\_number+**'?'**+curr\_bursttime);  
 arrival\_flag = **true**;  
 }  
 });  
 **return** arrival\_flag;  
}  
  
**function** *GET\_BURSTTIME\_TOTAL*(){  
 **var** total = 0.0;  
 $(**'.burst\_time'**).each(**function**(index){  
 total += *parseFloat*($(**this**).val());  
 });  
  
 **if**(*GET\_ARRIVALTIME\_HIGHEST*() > total){  
 total = *GET\_ARRIVALTIME\_HIGHEST*();  
 }  
  
 **return** *parseFloat*(total);  
}  
  
**function** *SORT\_READY\_QUEUE*(){  
 ***ready\_queue***.sort(**function**(a,b){ *// sort queue by lowest bt first* **return** a.split(**'?'**)[1] - b.split(**'?'**)[1]  
 });  
}  
</**script**>  
</**head**>  
<**body**>  
Scheduling Methods  
<**select id="methods"**>  
 <**option value=""**>SELECT A METHOD</**option**>  
 <**option value="FirstComeFirstServe.html"**>First Come, First Serve (FCFS)</**option**>  
 <**option value="ShortestJobFirst.html"**>Shortest Job First</**option**>  
 <**option value="ShortestTimeRemaining.html"**>Shortest Time Remaining</**option**>  
 <**option value="PriorityNonPreemptive.html"**>Priority Non-Preemptive</**option**>  
 <**option value="PriorityPreemptive.html"**>Priority Preemptive</**option**>  
 <**option value="RoundRobin.html"**>Round Robin</**option**>  
</**select**>  
<**br**/>  
<**br**/>  
<**h1**>Round Robin | Q = <**input type="text" id="quantum"** /></**h1**>  
<**table border="1"**>  
 <**thead**>  
 <**tr**>  
 <**td**>Process Name</**td**>  
 <**td**>Arrival Time</**td**>  
 <**td**>Burst Time</**td**>  
 <**td**>Turn-Around Time</**td**>  
 <**td**>Waiting Time</**td**>  
 </**tr**>  
 </**thead**>  
 <**tbody**>  
 <**tr**>  
 <**td**>P1</**td**>  
 <**td**><**input data-process="1" type="text" class="arrival\_time"** /></**td**>  
 <**td**><**input data-process="1" type="text" class="burst\_time"** /></**td**>  
 <**td**><**span class="TAT" id="P1\_TAT"**></**span**></**td**>  
 <**td**><**span class="WT" id="P1\_WT"**></**span**></**td**>  
 </**tr**>  
 <**tr**>  
 <**td**>P2</**td**>  
 <**td**><**input data-process="2" type="text" class="arrival\_time"** /></**td**>  
 <**td**><**input data-process="2" type="text" class="burst\_time"** /></**td**>  
 <**td**><**span class="TAT" id="P2\_TAT"**></**span**></**td**>  
 <**td**><**span class="WT" id="P2\_WT"**></**span**></**td**>  
 </**tr**>  
 <**tr**>  
 <**td**>P3</**td**>  
 <**td**><**input data-process="3" type="text" class="arrival\_time"** /></**td**>  
 <**td**><**input data-process="3" type="text" class="burst\_time"** /></**td**>  
 <**td**><**span class="TAT" id="P3\_TAT"**></**span**></**td**>  
 <**td**><**span class="WT" id="P3\_WT"**></**span**></**td**>  
 </**tr**>  
 <**tr**>  
 <**td**>P4</**td**>  
 <**td**><**input data-process="4" type="text" class="arrival\_time"** /></**td**>  
 <**td**><**input data-process="4" type="text" class="burst\_time"** /></**td**>  
 <**td**><**span class="TAT" id="P4\_TAT"**></**span**></**td**>  
 <**td**><**span class="WT" id="P4\_WT"**></**span**></**td**>  
 </**tr**>  
 <**tr**>  
 <**td colspan="3"**>Average</**td**>  
 <**td**><**span id="AVG\_TAT"**></**span**></**td**>  
 <**td**><**span id="AVG\_WT"**></**span**></**td**>  
 </**tr**>  
 </**tbody**>  
</**table**>  
<**br**/>  
<**div style="width**: 80%**"**>  
 <**div id="gantt\_chart"**>  
 </**div**>  
</**div**>  
<**div style="clear**: **both**;**"**></**div**>  
<**p id="cust\_console" style="color**:**red**;**"**></**p**>  
<**button id="INIT\_COMPUTE"**>Compute</**button**>  
</**body**>  
</**html**>