Republic of the Philippines
Technological University of the Philippines
COLLEGE OF SCIENCE
Manila

BACHELOR OF SCIENCE IN INFORMATION TECHNOLOGY

## Project 1:

# CPU Scheduling



**SANTOS, RAMINO JAKE H.  PANALIGAN, FRANCIS EDIAN M.  VICTORIA, JERICO M.**

# TABLE OF CONTENTS

# Background

CPU Scheduling is the process of determining which process will have exclusive use of the CPU while another is paused. The main goal of CPU scheduling is to ensure that whenever the CPU is idle, the OS chooses at least one of the processes in the ready queue to run. We, students, had a hard time back then because we only did it manually, which was also time consuming, so we decided to create a program that will speed up the computation while also generating the Gantt chart that its users will require. The CPU scheduling algorithms we developed for this program is Non-preemptive Priority.

Non-preemptive Priority is an algorithm that schedules processes based on the priority number that has been assigned to them. When a process is scheduled, it will run until it is completed. In general, the lower the priority number, the more important the process is.

# Objectives

Our team's goal is to create a program that can compute and generate a correct data table, including the Average Turnaround time and Average Wait time, as well as a Gantt chart, for Non-preemptive algorithm. The program might compute and generate Gantt chart for the algorithm by using the values inserted by the user.

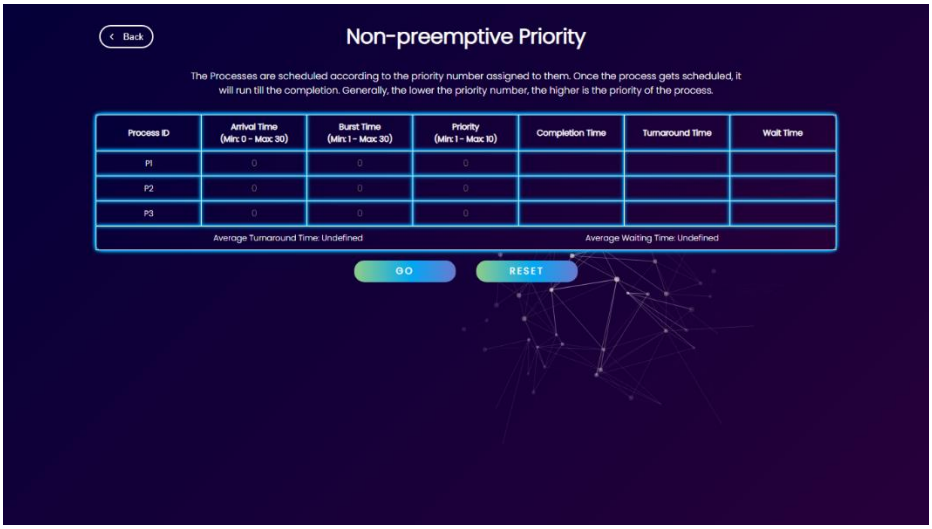
Our Non-preemptive priority algorithm was implemented with the use of PHP and JavaScript programming language because it is easy to understand and it is very similar to C programming. It has built-in functions which is useful for implementing this type of Algorithm. We used CSS and HTML for the design and architecture of the program.

# Scopes and Delimitations

Our system is **Non-preemptive priority**. It can handle one to ten processes with a minimum **Arrival time of 0-30**, a **Burst time of 1-30**, and a **priority of 1-10**. Inserting numbers below or above the limit will prompt an alert box stating that there is an error with the user input. Our system doesn't have a process sequence. Each process was being sorted using their process number if their Arrival time and Priority is the same.

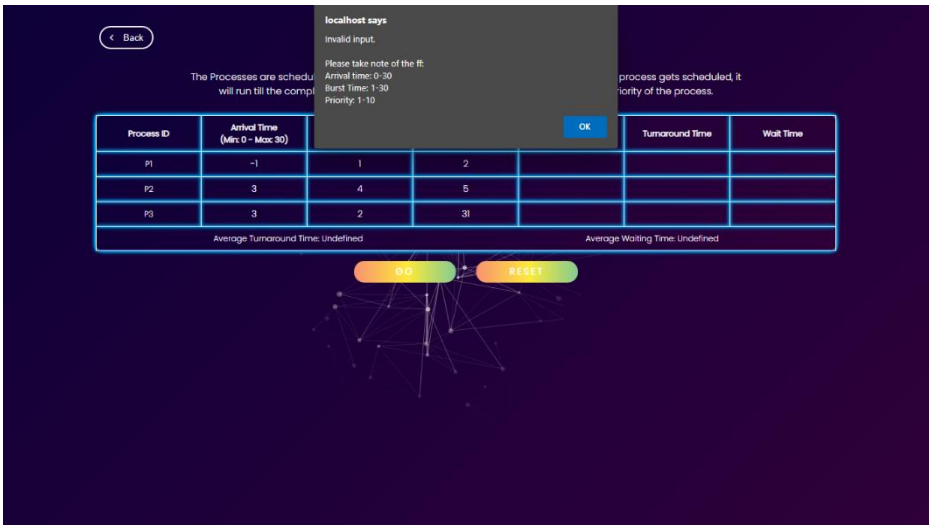
# Results and Discussions

After testing our system for several times, we can say that our program does its job. It can compute the Average Turnaround Time and Average Burst Time correctly as well as generate a Gantt chart which is easy to understand. The user can insert numbers inside a number inside the text boxes with a placeholder of "0". After inserting, they must click the "GO" button to generate the result.




*This is what the program looks like at the start.*

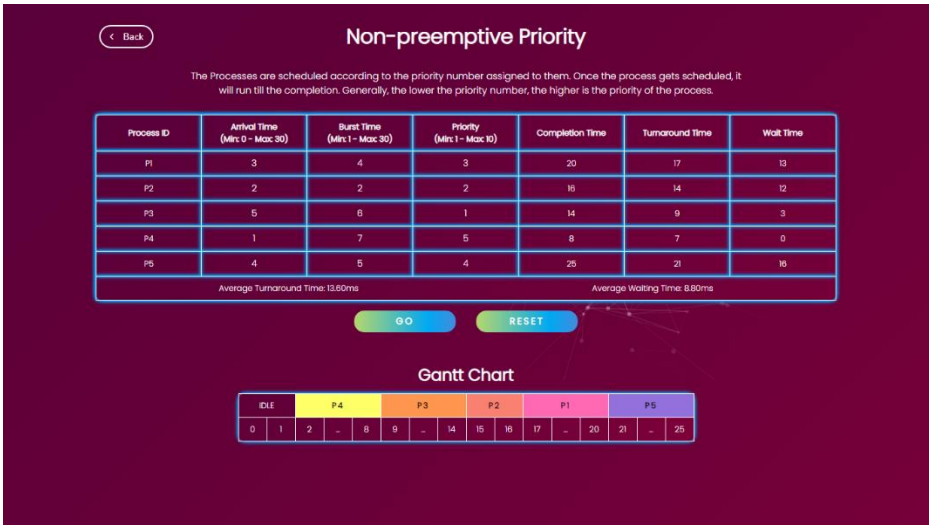
Stated in our scope and delimitations are the numbers that our program is allowed to handle. Let's try to insert numbers above and below the limit.




*User input Prompt for an Error*

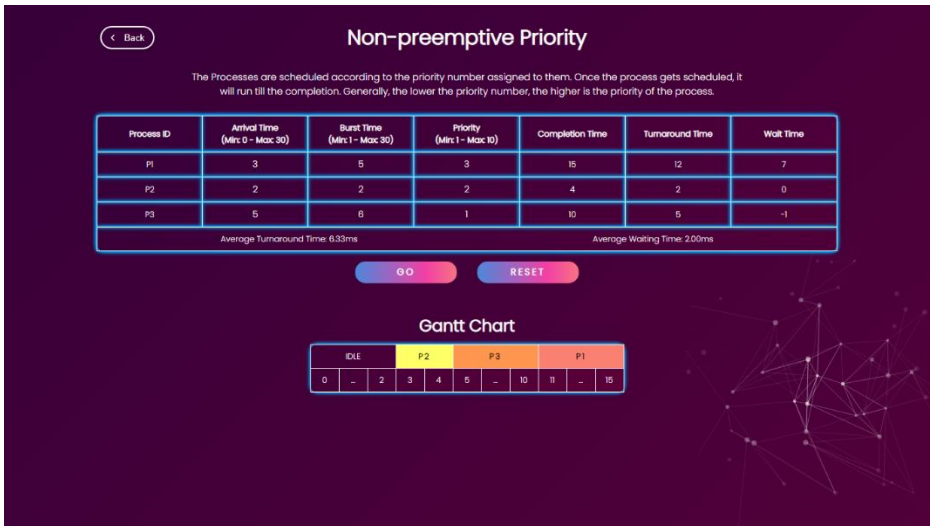
As you can see in the image above, the negative number and the number higher than 30 triggered the alert box stating that there is something wrong with the user input. Let's proceed with the 4-test data.

| **Test Data #1** | | | |
|---|---|---|---|
| Process ID | Arrival Time | Burst Time | Priority |
| P1 | 3 | 4 | 3 |
| P2 | 2 | 2 | 2 |
| P3 | 5 | 6 | 1 |
| P4 | 1 | 7 | 5 |
| P5 | 4 | 5 | 4 |



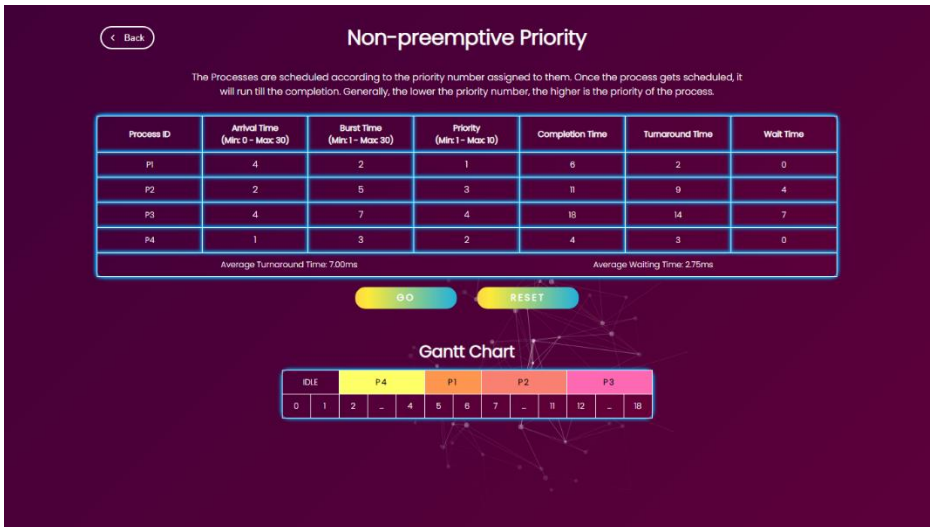

*Test Data #1 Average TAT, Average WT, and Gantt Chart was accurate.*

| Test Data #2 | | | |
|---|---|---|---|
| Process ID | Arrival Time | Burst Time | Priority |
| P1 | 3 | 5 | 3 |
| P2 | 2 | 2 | 2 |
| P3 | 5 | 6 | 1 |



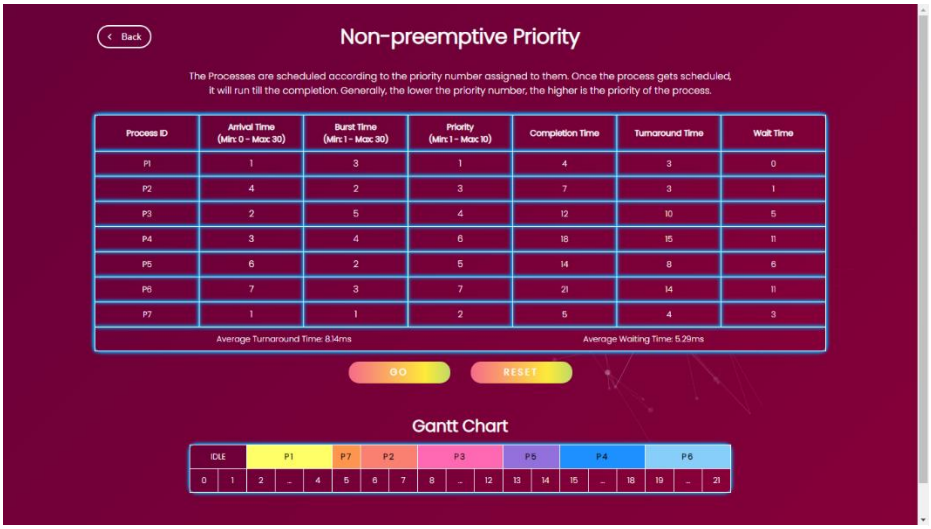

*Test Data #2 Average TAT, Average WT, and Gantt Chart was accurate.*

| Test Data #3 | | | |
|---|---|---|---|
| Process ID | Arrival Time | Burst Time | Priority |
| P1 | 4 | 2 | 1 |
| P2 | 2 | 5 | 3 |
| P3 | 4 | 7 | 4 |
| P4 | 1 | 3 | 2 |




*Test Data #3 Average TAT, Average WT, and Gantt Chart was accurate.*

| Test Data #4 | | | |
|---|---|---|---|
| Process ID | Arrival Time | Burst Time | Priority |
| P1 | 1 | 3 | 1 |
| P2 | 4 | 2 | 3 |
| P3 | 2 | 5 | 4 |
| P4 | 3 | 4 | 6 |
| P5 | 6 | 2 | 5 |
| P6 | 7 | 3 | 7 |
| P7 | 1 | 1 | 2 |




*Test Data #4 Average TAT, Average WT, and Gantt Chart was accurate.*

# Technical Difficulties

Here is a list of the technical difficulties that we have encountered:

- One of our team member's server has a problem with rendering the CSS on his browser. The solution that we did is we reinstalled XAMPP on his computer.
- We also encountered a problem with the algorithm when there are IDLE and same arrival time. Idle is not being followed so in order to fix it, we fixed the sorting algorithm and added a new sorting rule wherein processes with the same arrival time will be compare using their Priority and then the arriving processes will not be changed.

# Group Evaluation

- **Functionality (25 PTS):** When it comes to the average of TAT and WT, we can tell that the accuracy is 100 percent because we have tested it several times and every time, we encounter a bug, we immediately fix it, which is how we knew the system's accuracy was 100 percent.

- **Reliability (25 PTS):** Our program is user-friendly and simple to use, which contributes to its dependability. If the user forgets to insert some value or they inserted a number lower than the minimum and higher than the maximum value, the program will automatically display and prompt the user to insert or edit a value before proceeding to the next procedure.

- **Usability (25 PTS):** When it comes to the system's presentation, we can say that it is user friendly because it is simple to use and follows a single path. It is also entertaining because of eye-catching features such as the effect of the cursor and the animation in the button when the user hovers over it. The user can learn something because there is a meaning of the algorithm.

- **Efficiency (23 PTS):** In terms of efficiency, our program code is very long, and if carefully examined, it also takes up a lot of JavaScript memory. Despite the fact that the execution time is short, we believe that our algorithm can be improved, allowing us to save more resources.

| | |
|---|---|
| **FUNCTIONALITY** | 25 PTS. |
| **RELIABILITY** | 25 PTS. |
| **USABILITY** | 25 PTS. |
| **EFFICIENCY** | 23 PTS. |
| **TOTAL** | 98 PTS. |

# Conclusion and Recommendation

We can say that our program does its job after we put it through various tests. The user can insert values that will not exceed or below the limit. Minimum and maximum limit was indicated at the table header. It's output in the table and Gantt charts is accurate after we tried to solve it manually. So far, we haven't encountered any issues with the algorithm we implemented. It is simple to use because it is a website. This program will benefit both professors and students.

In the future, we believe we will need to improve our algorithm because there are some built-in functions that we can use which has faster execution time than our current algorithm. We also have to ensure that no bugs might appear in it in order to achieve 100 percent accuracy, which is the goal of our program.

# Appendices

## Job Distribution:

- Victoria, Jerico M.
  - Lead Programmer
- Panaligan, Francis Edian M.
  - Programmer
  - Tester
  - Designer
- Santos, Ramino Jake H.
  - Programmer
  - Tester
  - Quality Assurance

## Source Code:

cpuscheduling.php

```
<!DOCTYPE html>
<meta charset="utf-8" name="viewport" content="width=device-width, initial-scale=1.0">
<html>
<head>
  <link rel="stylesheet" href="process.css">
  <link rel="stylesheet" href="colors.css">
  <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.7.0/css/font-awesome.min.css">
</head>
<body onload="load_title()">

  <?php
  session_start();
  $numprocess = $_SESSION['nump'];
  $algorithm = $_SESSION['algorithm'];
  ?>
  <script type="text/javascript">
    var numprocess = "<?php echo $numprocess; ?>";
    var algo = "<?php echo $algorithm; ?>";
  </script>
  <div class="container demo">
   <div class="content">
      <div id="large-header" class="large-header">
         <canvas id="demo-canvas"></canvas>
      </div>
   </div>
  </div>
  <section id="input-page">
    <div id="nav">
      <button id="back" onclick="back_main()"><i class="fa fa-angle-left" style="margin-right: 15px; font-size: 15px;"></i>Back</button>
      <h1 id="algo-title"> CPU Scheduling </h1>
      <button id="back" style="visibility: hidden;"><i class="fa fa-angle-left" style="margin-right: 15px; font-size: 15px;"></i>Back</button>
    </div>
    <p id="desc-title"> Description </p>
    <div id="input-table">
      <table style="width:80%" align="center" id="main-table">
      <tr>
        <th>Process ID</th>
        <th>Arrival Time</br>(Min: 0 - Max: 30)</th>
        <th>Burst Time</br>(Min: 1 - Max: 30)</th>
        <?php
        if ($algorithm == "npp"){
          echo '<th>Priority</br>(Min: 1 - Max: 10)</th>';
        }
        ?>
        <th>Completion Time</th>
        <th>Turnaround Time</th>
        <th>Wait Time</th>
```

```
      </tr>

      <?php
      for ($i = 0; $i < $numprocess; $i++){
        echo '<tr id="table-val">';
        echo '<td> P'.$i +  1 .'</td>';
        echo '<td> <input type="number" placeholder="0" id="process-id"
min="1" maxlength="2"> </td>';
        echo '<td> <input type="number" placeholder="0" id="process-id"
min="0" maxlength="2"> </td>';
        if ($algorithm == "npp"){
          echo '<td> <input type="number" placeholder="0" id="process-id"
min="0" maxlength="2"> </td>';
          $colspan_bottom = 7;
        }
        else{
          $colspan_bottom = 6;
        }
        echo '<td> </td>';
        echo '<td> </td>';
        echo '<td> </td>';
        echo '</tr>';
      }
      ?>
      <tr>
        <?php
          echo '<td colspan="'.$colspan_bottom.'"><div id="ave"><span
id="ave-tat">Average Turnaround Time: Undefined</span> <span id="ave-wt">
Average Waiting Time: Undefined</span></div></td>';
        ?>
      </tr>
    </table>
    <div id="buttons">
      <input type="submit" onclick="getValue()" value="GO" id="go-button">
      <input type="submit" onclick="resetAll()" value="RESET" id="reset-
button">
    </div>
    </div>
  </section>
  <script src="https://s3-us-west-
2.amazonaws.com/s.cdpn.io/499416/TweenLite.min.js"></script>
  <script src="https://s3-us-west-
2.amazonaws.com/s.cdpn.io/499416/EasePack.min.js"></script>
  <script src="mouseeffect.js"></script>
  <script src="index.js" type="text/javascript"></script>
</body>
</html>
```

index.js

```
var numprocess;
var mainscreen = document.getElementById('processes');
var tmpid, tmpat, tmpbt, tmpprio, atContainer;
var y = [],
  backupOfY = [],
  secondaryBackupOfY = [],
  addToNext = [],
  chartArray = [],
  chartArrayTemp = [],
  completionTime = [],
  finalCT = [],
  tat = [],
  wt = [],
  ppContainer = [];
var totalBT = 0;
var inputValidator;
var repeat = false;
var colors = ["yellow", "orange", "red", "pink", "violet", "blue", "lblue",
"lgreen", "green", "lime"];
var algoTitle, descTitle;
var colorSetter;

function load_title() {
  if (algo == "fcfs") {
    algoTitle = "First come, First Served";
    descTitle = "First Come First Serve (FCFS) is an operating system
scheduling algorithm that automatically executes queued requests and
processes in order of their arrival.  It is the easiest and simplest CPU
scheduling algorithm.";
  } else if (algo == "sjf") {
    algoTitle = "Shortest Job First";
    descTitle = "Shortest Job First (SJF) is an algorithm in which the
process having the smallest execution time is chosen for the next execution.
This scheduling method can be preemptive or non-preemptive.  It
significantly reduces the average waiting time for other processes awaiting
execution. The full form of SJF is Shortest Job First.";
  } else if (algo == "npp") {
    algoTitle = "Non-preemptive Priority";
    descTitle = " The Processes are scheduled according to the priority
number assigned to them. Once the process gets scheduled, it will run till
the completion. Generally, the lower the priority number, the higher is the
priority of the process.";
  }
  document.getElementById('algo-title').innerHTML = algoTitle;
  document.getElementById('desc-title').innerHTML = descTitle;
}

function processes() {
  numprocess = document.getElementById('num-process').value;
  window.location = 'cpuscheduling.php';
}

function back_main() {
  numprocess = 0;
```

```
  window.location.href = 'process.php';
}

function getValue() {
  //empty variable array
  y = [], backupOfY = [], secondaryBackupOfY = [], chartArray = [],
completionTime = [], finalCT = [], tat = [], wt = [];
  inputValidator = true;
  //check kung may table na ba sa site, kung meron idedelete niya para
mareplace.
  var myElem = document.getElementById('myTable');
  if (myElem != null) { //tanggalin kung present sa website yung table
output.
    var parentEl = myElem.parentElement;
    parentEl.removeChild(myElem);
  }
  var myElem = document.getElementById('GanttChartTitle');
  if (myElem != null) { //tanggalin kung present sa website yung table
output.
    var parentEl = myElem.parentElement;
    parentEl.removeChild(myElem);
  }

  //get the value of every textbox then ilalagay sa array
  for (i = 1; i <= numprocess; i++) {
    tmpid = document.getElementById("main-
table").rows[i].cells[0].innerHTML;
    tmpat = document.getElementById("main-
table").rows[i].cells[1].getElementsByTagName('input')[0].value;
    tmpbt = document.getElementById("main-
table").rows[i].cells[2].getElementsByTagName('input')[0].value;
    //insert at the end of the array
    if (algo == "npp") {
      tmpprio = document.getElementById("main-
table").rows[i].cells[3].getElementsByTagName('input')[0].value;
      if (tmpprio == "" || tmpprio < 1 || tmpprio > 10) {
        inputValidator = false;
        break;
      }
    }
    if (tmpat == "" || tmpbt == "" || tmpat < 0 || tmpbt < 1 || tmpat > 30
|| tmpbt > 30) {
      inputValidator = false;
      break;
    } else {
      if (algo == "npp") {
        y.push([tmpid, parseInt(tmpat), parseInt(tmpbt), parseInt(tmpprio),
colors[i - 1]]);
        backupOfY.push([tmpid, parseInt(tmpat), parseInt(tmpbt),
parseInt(tmpprio), colors[i - 1]]);
        secondaryBackupOfY.push([tmpid, parseInt(tmpat), parseInt(tmpbt),
parseInt(tmpprio), colors[i - 1]]);
      } else {
        y.push([tmpid, parseInt(tmpat), parseInt(tmpbt)]);
        backupOfY.push([tmpid, parseInt(tmpat), parseInt(tmpbt)]);
        secondaryBackupOfY.push([tmpid, parseInt(tmpat), parseInt(tmpbt)]);
      }
```

```
    }
  }
  //run function para magenerate na yung table.
  if (inputValidator == true) {
    if (algo == "fcfs")
      generateTableFCFS();
    else if (algo == "sjf")
      generateTableSJF();
    else if (algo == "npp")
      generateTableNPP();
  } else {
    alert("Invalid input.\n\nPlease take note of the ff:\nArrival time: 0-
30\nBurst Time: 1-30\nPriority: 1-10");
  }
}

function resetAll(){
  var confirmReset = confirm("Are you sure?");
  if (confirmReset == true){
    var myElem = document.getElementById('myTable');
    if (myElem != null) { //tanggalin kung present sa website yung table
output.
      var parentEl = myElem.parentElement;
      parentEl.removeChild(myElem);
    }
    var myElem = document.getElementById('GanttChartTitle');
    if (myElem != null) { //tanggalin kung present sa website yung table
output.
      var parentEl = myElem.parentElement;
      parentEl.removeChild(myElem);
    }
    if (algo == "npp"){
      for (i = 1; i <= numprocess; i++) {
        document.getElementById("main-
table").rows[i].cells[1].getElementsByTagName('input')[0].value = "";
        document.getElementById("main-
table").rows[i].cells[2].getElementsByTagName('input')[0].value = "";
        document.getElementById("main-
table").rows[i].cells[3].getElementsByTagName('input')[0].value = "";
        document.getElementById("main-table").rows[i].cells[4].innerHTML =
"";
        document.getElementById("main-table").rows[i].cells[5].innerHTML =
"";
        document.getElementById("main-table").rows[i].cells[6].innerHTML =
"";
      }
    } else {
      for (i = 1; i <= numprocess; i++) {
        document.getElementById("main-
table").rows[i].cells[1].getElementsByTagName('input')[0].value = "";
        document.getElementById("main-
table").rows[i].cells[2].getElementsByTagName('input')[0].value = "";
        document.getElementById("main-table").rows[i].cells[3].innerHTML =
"";
        document.getElementById("main-table").rows[i].cells[4].innerHTML =
"";
```

```
        document.getElementById("main-table").rows[i].cells[5].innerHTML =
"";
      }
    }
    document.getElementById("ave-tat").innerHTML = "Average Turnaround Time:
Undefined";
    document.getElementById("ave-wt").innerHTML = "Average Waiting Time:
Undefined";
  }
}

function generateTableFCFS() {
  chartArray = [];
  //code to sort array based on their arrival time
  y.sort(function(a, b) {
    return a[1] - b[1];
  });

  var ctr = 0;
  var i = 0;

  //main algorithm
  while (ctr != y.length) {
    var bt = y[ctr][2];
    var btctr = 0;
    atContainer = y[ctr][1] + 1 ;

    if (i >= atContainer) { // Kapag parehas or greater than yung AT at
value ni i, papasok sa IF na ito.
      while (btctr < bt) {
        chartArray.push(y[ctr][0]);
        btctr++;
        i++;
      }
      ctr++;
    } else { //Kung di naman parehas or greater than yung value, dito siya
papasok para magdagdag ng empty array.
      chartArray.push("");
      i++;
    }
  }

  //create main table. Yung <table> </table>
  generateTable();
}

function generateTableSJF() {
  chartArray = [];

  var ctr = 0;
  var i = 0;
  while (ctr != secondaryBackupOfY.length) {
    //sort arrival time
    secondaryBackupOfY.sort(function(a, b) {
      return a[1] - b[1];
    });
    var bt = 0;
```

```
    var btctr = 0;
    atContainer = secondaryBackupOfY[ctr][1] + 1 ;
    if (i == atContainer) {
      for (var compare = 1; compare < secondaryBackupOfY.length; compare++)
{
        if (i == secondaryBackupOfY[compare][1]) {
          //sort bt
          repeat = true;
        }
      }
      if (repeat == true) {
        secondaryBackupOfY.sort(function(a, b) {
          return a[1] - b[1] || a[2] - b[2];
        });
      }
      bt = secondaryBackupOfY[ctr][2];
      while (btctr < bt) {
        chartArray.push(secondaryBackupOfY[ctr][0]);
        btctr++;
        i++;
      }
      secondaryBackupOfY.shift();
    } else if (i > atContainer) {
      //sort burst time
      secondaryBackupOfY.sort(function(a, b) {
        return a[2] - b[2];
      });
      if(secondaryBackupOfY[ctr][1] > i){
        secondaryBackupOfY.sort(function(a, b) {
          return a[1] - b[1];
        });
      }

      bt = secondaryBackupOfY[ctr][2];
      while (btctr < bt) {
        chartArray.push(secondaryBackupOfY[ctr][0]);
        btctr++;
        i++;
      }
      secondaryBackupOfY.shift();
    } else {
      chartArray.push("");
      i++;
    }
  }

  generateTable();
}

function generateTableNPP() {
  chartArray = [];

  var ctr = 0;
  var i = 0;
  while (ctr != secondaryBackupOfY.length) {
    //sort arrival time
    secondaryBackupOfY.sort(function(a, b) {
```

```
      return a[1] - b[1];
    });
    var bt = 0;
    var btctr = 0;
    atContainer = secondaryBackupOfY[ctr][1] + 1;
    if (i == atContainer) {
      var shifter = 0;
      for (var compare = 1; compare < secondaryBackupOfY.length; compare++)
{
        if (i == secondaryBackupOfY[compare][1]) {
          //sort bt
          repeat = true;
        }
      }
      if (repeat == true) {
        secondaryBackupOfY.sort(function(a, b) {
          return a[1] - b[1] || a[3] - b[3];
        });
      }
      bt = secondaryBackupOfY[ctr][2];
      while (btctr < bt) {
        chartArray.push(secondaryBackupOfY[ctr][0]);
        btctr++;
        i++;
      }
      secondaryBackupOfY.shift();
    } else if (i > atContainer) {
      //sort burst time
      secondaryBackupOfY.sort(function(a, b) {
        return a[3] - b[3];
      });
      if(secondaryBackupOfY[ctr][1] > i){
        secondaryBackupOfY.sort(function(a, b) {
          return a[1] - b[1];
        });
      }
      bt = secondaryBackupOfY[ctr][2];
      while (btctr < bt) {
        chartArray.push(secondaryBackupOfY[ctr][0]);
        btctr++;
        i++;
      }
      secondaryBackupOfY.shift();
    } else {
      chartArray.push("");
      i++;
    }
  }

  generateTable();
}

function generateTable(){
  var a = document.createElement("h1");
  a.setAttribute("id", "GanttChartTitle");
  a.setAttribute("align", "center");
  document.body.appendChild(a);
```

```
  document.getElementById("GanttChartTitle").innerHTML = "Gantt Chart";

  var a = document.createElement("TABLE");
  a.setAttribute("id", "myTable");
  a.setAttribute("align", "center");
  document.body.appendChild(a);
  var b = document.createElement("TR");
  b.setAttribute("id", "ganttProc");

  //table for gantt chart
  var f = document.createElement("TR");
  f.setAttribute("id", "ganttChart");
  document.getElementById("myTable").appendChild(f);

  //table for time
  var f = document.createElement("TR");
  f.setAttribute("id", "ganttTime");
  document.getElementById("myTable").appendChild(f);

  var j = 0;
  for (i = 0; i < chartArray.length; i++) {
    var c_span = 0;
    var g = document.createElement("TD");
    //para sa column span.
    var tempContainer = chartArray[i];
    var v = document.createElement("TD");
    var w = document.createTextNode(i);
    v.appendChild(w);
    document.getElementById("ganttTime").appendChild(v);
    while (chartArray[i] == tempContainer) {
      c_span++;
      i++;
    }
    i -= 1;
    if (chartArray[i] == "")
      var h = document.createTextNode("IDLE");
    else
      var h = document.createTextNode(chartArray[i]);
    if (c_span > 2){
      c_span = 3;
      //print ellipsis
      var v = document.createElement("TD");
      var w = document.createTextNode("...");
      v.appendChild(w);
      document.getElementById("ganttTime").appendChild(v);

      //print the last ms
      var v = document.createElement("TD");
      var w = document.createTextNode(i);
      v.appendChild(w);
      document.getElementById("ganttTime").appendChild(v);
    } else if (c_span <= 2 && c_span >1){
      var v = document.createElement("TD");
      var w = document.createTextNode(i);
      v.appendChild(w);
      document.getElementById("ganttTime").appendChild(v);
    }
```

```
    g.setAttribute("colspan", c_span);

    if (chartArray[i] != ""){
      g.setAttribute("class", colors[j]);
      j++; //increment j para sa next color;
    }
    g.appendChild(h);
    document.getElementById("ganttChart").appendChild(g);
  }

  //loop para makuha yung completion time
  var getIndex = 0;
  for (j = 0; j < y.length; j++) {
    for (i = 0; i < chartArray.length; i++)
      if (y[j][0] == chartArray[i]) {
        getIndex = i;
      }
    completionTime.push([y[j][0], getIndex]);
  }

  var cellCT = 3;
  var cellTat = 4;
  var cellWT = 5;

  if (algo == "npp"){
    cellCT += 1;
    cellTat += 1;
    cellWT += 1;
  }

  //display sa table yung completion time
  i = 0, j = 0;
  while (i < y.length) {
    if (completionTime[i][0] == backupOfY[j][0]) {
      document.getElementById("main-table").rows[j +
1].cells[cellCT].innerHTML = completionTime[i][1];
      finalCT.push(completionTime[i][1]);
      i++;
      j = 0;
    } else {
      j++;
    }
  }

  //display tat and store the values inside an array
  for (i = 0; i < backupOfY.length; i++) {
    tat.push(document.getElementById("main-table").rows[i +
1].cells[cellCT].innerHTML - backupOfY[i][1]);
    document.getElementById("main-table").rows[i +
1].cells[cellTat].innerHTML = tat[i];
  }

  //display WT and store the values inside an array
  for (i = 0; i < backupOfY.length; i++) {
    wt.push(tat[i] - backupOfY[i][2]);
    document.getElementById("main-table").rows[i +
1].cells[cellWT].innerHTML = wt[i];
```

```
  }

  //print average tat
  var tempavetat = 0;
  for (i = 0; i < tat.length; i++) {
    tempavetat += tat[i];
  }
  tempavetat = tempavetat / tat.length;
  document.getElementById("ave-tat").innerHTML = "Average Turnaround Time: "
+ tempavetat.toFixed(2) + "ms";

  //print average wt
  var tempavewt = 0;
  for (i = 0; i < tat.length; i++) {
    tempavewt += wt[i];
  }
  tempavewt = tempavewt / tat.length;
  document.getElementById("ave-wt").innerHTML = "Average Waiting Time: " +
tempavewt.toFixed(2) + "ms";
}
```