University of Bahrain

College of Information Technology

Operating Systems (ITCS323/325) – Project
CPU Priority Scheduling-preemptive

Prepared by:

- Name: Maged Hussain Masaad Aljashoobi ID: 202004484
- Name: Najim Abdulkarem Musaed Alfutini ID: 202003215
- Name: Belal Moustafa Ibrahim ID: 20187717
- Name: Nawaf Ahmed Alruffai ID: 20173311
- Name: Hussain Ali Hassan Mearaj ID: 20193912

Section: 1

Date: 21-12-2022

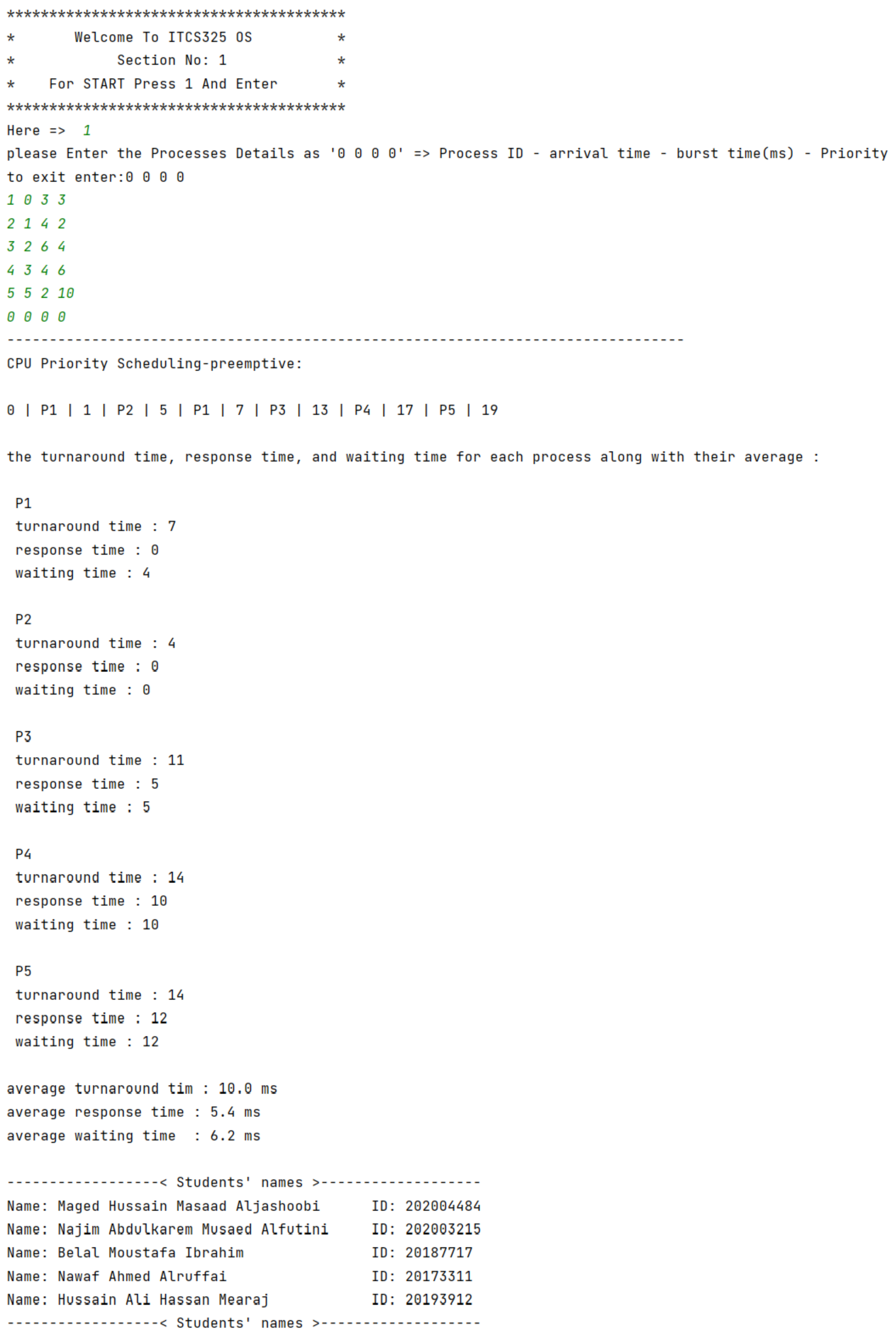
Java code:

```
import java.util.*;

/*
                Name: Maged Hussain Masaad Aljashoobi       ID: 202004484
                Name: Najim Abdulkarem Musaed Alfutini      ID: 202003215
                Name: Belal Moustafa Ibrahim                ID: 20187717
                Name: Nawaf Ahmed Alruffai                  ID: 20173311
                Name: Hussain Ali Hassan Mearaj             ID: 20193912
 */

class processes {
    // variables
    private int ID;
    private int AT;
    private int BT;
    private int priority;
    private int response_time = -1;
    private int turnAroundTime = -1;
    private int waitingTime = 0;
    private boolean complete = false;
    private int complete_time = 0;
    private int finishing_time;
    private int start = -1;

    // to Create an object from the Process class
    public processes(int ID, int AT, int BT, int priority) {
        this.ID = ID;
        this.AT = AT;
        this.BT = BT;
        this.priority = priority;
    }

    // getters and setter for each variables;
    public int getStart() {
        return start;
    }

    public void setStart(int start) {
        this.start = start;
    }

    public int getID() {
        return ID;
    }

    public int getAT() {
        return AT;
    }

    public int getBT() {
        return BT;
    }

    public int getPriority() {
        return priority;
    }

    public int getComplete_time() {
        return complete_time;
    }

    public boolean isComplete() {
        return complete;
    }

    public void setComplet_time() {
        complete_time++;
        if (complete_time == BT)
            complete = true;
    }

    public void setFinshing_time(int finishing_time) {
        this.finishing_time = finishing_time;
    }

    public int getResponse_time() {
        return response_time;
    }

    public void setResponse_time(int response_time) {
        this.response_time = response_time;
    }

    public int getTurnAroundTime() {
        return turnAroundTime;
    }

    public int getWaitingTime() {
        return waitingTime;
    }

    public int getFinishing_time() {
        return finishing_time;
    }

    public void setTurnAroundTime(int turnAroundTime) {
        this.turnAroundTime = turnAroundTime;
    }

    public void setWaitingTime(int waitingTime) {
        this.waitingTime = waitingTime;
    }

    public void setFinishing_time(int finishing_time) {
        this.finishing_time = finishing_time;
    }

}

class array_process {
    private processes[] thedata; // Processes array
    private int size ;   // the number of processes
    public double avrg_TT, avrg_WT, avrg_RT;

    // to create a process array
    public array_process() {
        size = 0;
        thedata = new processes[0];
    }

    //Sort by arrival time;
    public void reSort() {
```

```java
        processes temp1;
        for (int i = 0; i < size ; i++)
            for (int j = i+1; j < size; j++)
                if (thedata[i].getAT() > thedata[j].getAT()){
                    temp1 = thedata[i];
                    thedata[i]=thedata[j];
                    thedata[j]=temp1;
                }
    }

    // to check if the id is already exist
    public boolean validInput(int id) {
        for (int i = 0; i < size; i++) {
            if (thedata[i].getID() == id) {
                return false;
            }
        }

        return true;
    }

    //to add process to the array
    public void addprocess( int id, int at, int bt, int pri) {
        thedata = Arrays.copyOf(thedata,thedata.length+1);
        thedata[size] = new processes(id, at, bt, pri);
        size++;
    }

    // To find out the index of the process in the array using id
    public int indexOf(int id) {
        for (int i = 0; i < size; i++)
            if (thedata[i].getID() == id)
                return i;

        return -1;
    }

    // Scheduling the processes and draw a Gantt chart (as text)
    public boolean Scheduling() {
        if (size <= 0) return false; // if there is no process in the array;

        int total_time = 0;
        // Calculate the total time to completion
        for (processes pro : thedata) {
            total_time += pro.getBT();
        }

        int Cpri = 999, Cid = -1, Nid = -1 ;  // current priority - current id - New id;
        // this for loop used to draw a Gantt chart; Ctime = current time;
        for (int Ctime = 0; Ctime < total_time; Ctime++) {

            for (processes Pro : thedata) {
                // check The arrival time is within the current time and is not complete
to put it in the waiting state;
                if (Pro.getAT() <= Ctime && !Pro.isComplete()) {
                    //to get the process with highest priority
                    if (Pro.getPriority() < Cpri) {
                            Cpri = Pro.getPriority();
                            Nid = Pro.getID();

                    }
                }
            }
            // If the new id = -1 there is no process in current time and print "no
process" increment total time by one
            if (Nid == -1) {
                System.out.print(Ctime + " | no process" + " | ");
                total_time++;
                Nid =-2;
            }
            // if the new id = -2 there is no process in current time and dont print and
increment total time by one
            else if (Nid == -2){
                total_time++;
            }
            // if new process load to running print and increment complete time by one
and set the start time for process;
            else if (Nid >= 0 && Nid != Cid) {
                System.out.print(Ctime + " | P" + Nid + " | ");
                Cid = Nid;
                thedata[indexOf(Nid)].setComplet_time();
                // to set start time
                if (thedata[indexOf(Nid)].getStart() == -1) {
                    thedata[indexOf(Nid)].setStart(Ctime);

                }

            }
            // if same process still in running state dont print and increment complete
time by one;
            else if (Cid == Nid) {
                thedata[indexOf(Nid)].setComplet_time();
            }


            //if the process terminated => set finishing time and reset the priority;
            if (Nid >= 0)
                if (thedata[indexOf(Nid)].isComplete()) {
                    Cpri = 999;
                    thedata[indexOf(Nid)].setFinshing_time((Ctime) + 1);
                    Nid = -1;
                }

            //if all processes are terminated print the total time
            // and set the turnAround time, response time, and waiting time for each
process.
            if (Ctime + 1 == total_time) {
                System.out.println(total_time);
                for (processes pro : thedata) {
                    pro.setResponse_time(pro.getStart() - pro.getAT()); // set Respones
time for each process
                    pro.setTurnAroundTime(pro.getFinishing_time() - pro.getAT());
                    pro.setWaitingTime(pro.getTurnAroundTime() - pro.getBT());
                }
            }

        }


        return true;
    }

    public boolean AVRG() {
        if (size <= 0) return false;
        // the total for each time
```

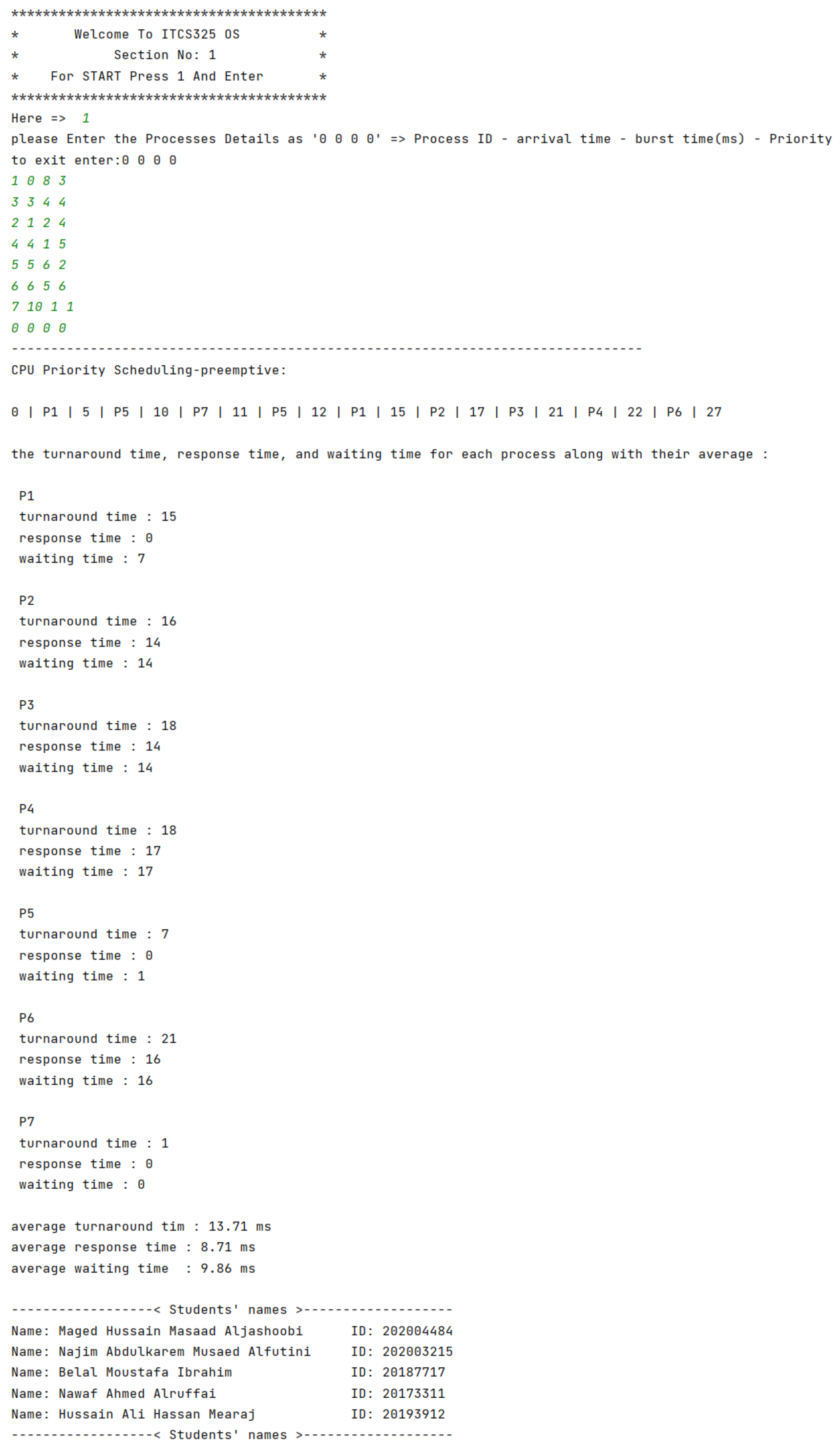
```java
        double WT = 0, TT = 0, RT = 0;
        for (processes pro : thedata) {
            WT += pro.getWaitingTime();
            TT += pro.getTurnAroundTime();
            RT += pro.getResponse_time();

        }
        // math.round to print value in two decimal digit after point
        avrg_RT =Math.round((RT / size) * 100.0) / 100.0;
        avrg_TT =Math.round((TT / size) * 100.0) / 100.0;;
        avrg_WT = Math.round((WT / size) * 100.0) / 100.0;
        return true;
    }

    public void print() {
        // to print the turnAround time, response time, and waiting time for each
process.
        for (processes pro : thedata) {
            System.out.println(" P"+pro.getID()+
                    "\n turnaround time : "+pro.getTurnAroundTime()+
                    "\n response time : "+pro.getResponse_time()+
                    "\n waiting time : "+pro.getWaitingTime()+"\n");
        }

        this.AVRG();
        // print the average for all processes
        System.out.println("average turnaround tim : " + avrg_TT+" ms");
        System.out.println("average response time : " + avrg_RT+" ms");
        System.out.println("average waiting time  : " + avrg_WT+" ms");

    }


}

public class priority {
   // welcome screen
    static {
        for (int i = 0; i < 5; i++) {
            for (int j = 0; j < 40; j++) {
                if (i == 0 || i == 4)
                    System.out.print('*');
                else if (j == 0 || j == 39) System.out.print('*');
                else {
                    if (i == 1 && j == 8) {
                        System.out.print("Welcome To ITCS325 OS");
                        j = 28;
                    }//first line
                    else if (i == 2 && j == 13) {
                        System.out.print("Section No: 1");
                        j = 25;
                    }//second line
                    else if (i == 3 && j == 5) {
                        System.out.print("For START Press 1 And Enter");
                        j = 31;
                    }//last line
                    else System.out.print(" ");
                }
            }
            System.out.println();
        }
    }
    public static void main(String[] args) {
        Scanner kbd = new Scanner(System.in);
        int id, pri, at, bt;
        System.out.print("Here =>  ");
        String IN = kbd.next();
        while (!IN.equals("1")) {
            System.out.println("Please Press 1 And Enter For Services: ");
            IN = kbd.next();
        }

        // Create an array
        array_process process1 = new array_process();

        System.out.println("please Enter the Processes Details as '0 0 0 0' => Process
ID - arrival time - burst time(ms) - Priority\n" +
                "to exit enter:0 0 0 0");

        int inputNumber = 0;
        while (true) {
            inputNumber++;
            id = kbd.nextInt();
            at = kbd.nextInt();
            bt = kbd.nextInt();
            pri = kbd.nextInt();
            //  if the input = 0 0 0 0 exit
            if ((id + at + bt + pri) == 0) {
                process1.reSort();
                break;
            }

            boolean done = process1.validInput(id);
            //check if the valus is valid and add process to array;
             if (done && at >= 0 && bt > 0){
                            process1.addprocess(id, at, bt, pri);

                        }
            //check if the value is not valid and print error massege
            if (!done || at < 0  || bt <= 0)  {
                System.out.print("\nthe input number " + (inputNumber) + " has an
incorrect value => ");

                if (!done) System.out.print("the ID is already exist ");
                if (at < 0) System.out.print("the Arrival time is not valid ");
                if (bt < 0) System.out.print("the burst time is not valid ");
                System.out.println("---->Please enter a valid value ---- (for exit Enter
0 0 0 0)");
            }
        }

        System.out.println("------------------------------------------------------------
--------------------");
        System.out.println("CPU Priority Scheduling-preemptive: \n");
        if(process1.Scheduling()){
            System.out.println();
            System.out.println("the turnaround time, response time, and waiting time for
each process along with their average : \n");
            process1.print();
        }
        else System.out.println("there is no process ");

        System.out.println();

        // close kbd Scanner
        kbd.close();
        // Students' names
        System.out.println("------------------< Students' names >-------------------");
        System.out.println("""
                Name: Maged Hussain Masaad Aljashoobi      ID: 202004484
                Name: Najim Abdulkarem Musaed Alfutini     ID: 202003215
                Name: Belal Moustafa Ibrahim               ID: 20187717
                Name: Nawaf Ahmed Alruffai                 ID: 20173311
                Name: Hussain Ali Hassan Mearaj            ID: 20193912""");
        System.out.println("------------------< Students' names >------------------
\n");

    } // end main method



}
```

Example 1:



```
****************************************
*       Welcome To ITCS325 OS          *
*            Section No: 1             *
*    For START Press 1 And Enter       *
****************************************
Here =>  1
please Enter the Processes Details as '0 0 0 0' => Process ID - arrival time - burst time(ms) - Priority
to exit enter:0 0 0 0
1 0 3 3
2 1 4 2
3 2 6 4
4 3 4 6
5 5 2 10
0 0 0 0
--------------------------------------------------------------------------------
CPU Priority Scheduling-preemptive:

0 | P1 | 1 | P2 | 5 | P1 | 7 | P3 | 13 | P4 | 17 | P5 | 19

the turnaround time, response time, and waiting time for each process along with their average :

 P1
 turnaround time : 7
 response time : 0
 waiting time : 4

 P2
 turnaround time : 4
 response time : 0
 waiting time : 0

 P3
 turnaround time : 11
 response time : 5
 waiting time : 5

 P4
 turnaround time : 14
 response time : 10
 waiting time : 10

 P5
 turnaround time : 14
 response time : 12
 waiting time : 12

average turnaround tim : 10.0 ms
average response time : 5.4 ms
average waiting time  : 6.2 ms

------------------< Students' names >-------------------
Name: Maged Hussain Masaad Aljashoobi      ID: 202004484
Name: Najim Abdulkarem Musaed Alfutini     ID: 202003215
Name: Belal Moustafa Ibrahim               ID: 20187717
Name: Nawaf Ahmed Alruffai                 ID: 20173311
Name: Hussain Ali Hassan Mearaj            ID: 20193912
------------------< Students' names >-------------------
```

Example 2:



```
****************************************
*       Welcome To ITCS325 OS          *
*            Section No: 1             *
*    For START Press 1 And Enter       *
****************************************
Here =>  1
please Enter the Processes Details as '0 0 0 0' => Process ID - arrival time - burst time(ms) - Priority
to exit enter:0 0 0 0
1 0 8 3
3 3 4 4
2 1 2 4
4 4 1 5
5 5 6 2
6 6 5 6
7 10 1 1
0 0 0 0
--------------------------------------------------------------------------------
CPU Priority Scheduling-preemptive:

0 | P1 | 5 | P5 | 10 | P7 | 11 | P5 | 12 | P1 | 15 | P2 | 17 | P3 | 21 | P4 | 22 | P6 | 27

the turnaround time, response time, and waiting time for each process along with their average :

 P1
 turnaround time : 15
 response time : 0
 waiting time : 7

 P2
 turnaround time : 16
 response time : 14
 waiting time : 14

 P3
 turnaround time : 18
 response time : 14
 waiting time : 14

 P4
 turnaround time : 18
 response time : 17
 waiting time : 17

 P5
 turnaround time : 7
 response time : 0
 waiting time : 1

 P6
 turnaround time : 21
 response time : 16
 waiting time : 16

 P7
 turnaround time : 1
 response time : 0
 waiting time : 0

average turnaround tim : 13.71 ms
average response time : 8.71 ms
average waiting time  : 9.86 ms

------------------< Students' names >-------------------
Name: Maged Hussain Masaad Aljashoobi      ID: 202004484
Name: Najim Abdulkarem Musaed Alfutini     ID: 202003215
Name: Belal Moustafa Ibrahim               ID: 20187717
Name: Nawaf Ahmed Alruffai                 ID: 20173311
Name: Hussain Ali Hassan Mearaj            ID: 20193912
------------------< Students' names >-------------------
```