```
import java.util.*;
import java.io.*;

public class LRTF
{

	static int Turn_around;
	static int avg_turn_around;
	public static void schedule(int[] pid,int[] run,int[] arrival)
	{

		int current_time = arrival[0];
		int count = 0;
		int[] q = new int[100];
		boolean allArrived = false;
		int qcount = 0;

		for(int i=0;i<arrival.length;i++)
		{
			if(arrival[i] == current_time)
			{
				q[qcount] = pid[i];
				qcount++;
				count++;
				System.out.println("Turn around time",(current_time+arrival[i]));
				System.out.println("Average turnaround
time",((current_time+arrival[i])/count++));
			}
			else if(arrival[i]>current_time)
				break;
		}

		if(count == arrival.length)
				allArrived = true;

		while(!allArrived || qcount !=0)
		{
			int index = findMax(q, run, count);

			int pidmax = index;
			System.out.println("Chosen PIDMAX: "+pidmax);
			if(!allArrived)
			{

			if(qcount == 0)
			{
				current_time = arrival[count];
				q[qcount] = pid[count];
				qcount++;
				count++;

				if(count == arrival.length)
					allArrived = true;
				continue;
			}
			if(arrival[count]<(current_time+run[pidmax]))
			{
```

```
                        //Check, execute pre-empt if required. Loop back.
                        int updated_time = arrival[count];
                        q[qcount] = pid[count];

                        count++;
                        qcount++;
                        run[pidmax] = run[pidmax] - (updated_time - current_time);
                        current_time = updated_time;
                        System.out.println("Turn around time",(current_time+arrival[i]));
                        System.out.println("Average turnaround
time",((current_time+arrival[i])/count++));

                        if(count == arrival.length)
                              allArrived = true;
                        continue;
                        System.out.println("Turn around time",(current_time+arrival[i]));
                        System.out.println("Average turnaround
time",((current_time+arrival[i])/count++));


                  }

                  }
                  current_time = current_time+run[pidmax];
                  run[pidmax] = 0;

                  int i = q[qcount-1];
                  q[index] = i;
                  qcount--;


                  System.out.println("Current time: "+current_time);
                  System.out.println("Turn around time",(current_time+arrival[i]));
                  System.out.println("Average turnaround time",((current_time+arrival[i])/count++));

            }
      }

      public static int findMax(int q[],int run[], int count)
      {
            int index = -1;
            int max = 0;

            for(int i=0;i<count;i++)
            {
                  if(run[i]!=0 && run[i]>max)
                  {
                        max = run[i];
                        index = i;
                  }
            }

            return index;
      }

      public static void main(String args[])
```

```
        {
        Scanner s=new Scanner(System.in);
        static int n;
        int arrival[] = new int[n];
        int run[]=new int[n];
        int pid=new int[n];
              System.out.println("Enter the pid of entering processes:");
              for(int i = 0;i < n;i++)
              {
                    pid[i] = s.nextInt();

              }
              System.out.println("Enter the running time for all the processes:");
              for(int i = 0;i < n;i++)
              {
                 int n;
                   n=count++;
               int waiting_time=0;
                    run[i] = s.nextInt();
                    waiting_time = waiting_time+ a[i];
               avg_waiting_time=(waiting_time/n);
              }

              System.out.println("Enter the arrival time:");
              for(int i = 0;i < n;i++)
              {
                    arrival[i] = s.nextInt();

              }
               schedule(pid, run, arrival);

        }
}
```