Homework Turnin

Account:	6G_06 (rgalanos@fcps.edu)
Section:	6G
Course:	TJHSST APCS 2016-17
Assignment:	06-07
Receipt ID:	cb803e22be7dafcc9a61e4d01f5a98ef

Turnin Successful!

The following file(s) were received:

McRonald3.java (5650 bytes)

```java
//name :   date:

import java.util.*;
public class McRonald3
{
   public static final int TIME = 1079;  //18 hrs * 60 min
   public static void main(String[] args)
   {
      int numberCustomers = 0;
      int totalWait = 0;
      int longestWait = 0;
      int currentSize = 0;
      int longestQueue = 0;
      int[] servingTime = {100, 100, 100};
      int[] count = {0, 0, 0};

      Queue<Integer> customers = new LinkedList<Integer>();
      Queue<Integer> service1 = new LinkedList<Integer>();
      Queue<Integer> service2 = new LinkedList<Integer>();
      Queue<Integer> service3 = new LinkedList<Integer>();

      for(int i=0; i<TIME; i++)
      {
         if(Math.random()<0.5)
         {
            customers.add(new Integer(i));
            currentSize++;
            numberCustomers++;
            if(currentSize>longestQueue)
               longestQueue = currentSize;
         }

         if(!service1.isEmpty())
         {
            count[0]++;
            if(servingTime[0]==count[0])
            {
               int x = i - service1.remove();
               totalWait+=x;
               if(x>longestWait)
                  longestWait = x;
            }
         }
         if(!service2.isEmpty())
         {
            count[1]++;
            if(servingTime[1]==count[1])
            {
               int x = i - service2.remove();
               totalWait+=x;
               if(x>longestWait)
                  longestWait = x;
            }
         }
         if(!service3.isEmpty())
         {
            count[2]++;
            if(servingTime[2]==count[2])
            {
               int x = i - service3.remove();
               totalWait+=x;
               if(x>longestWait)
                  longestWait = x;
            }
         }


         if(service1.isEmpty()&&!customers.isEmpty())
         {
            service1.add(customers.remove());
            servingTime[0] = (int)(Math.random()*6+2);
            count[0] = 0;
            currentSize--;
         }
         if(service2.isEmpty()&&!customers.isEmpty())
         {
            service2.add(customers.remove());
            servingTime[1] = (int)(Math.random()*6+2);
            count[1] = 0;
            currentSize--;
         }
         if(service3.isEmpty()&&!customers.isEmpty())
         {
            service3.add(customers.remove());
            servingTime[2] = (int)(Math.random()*6+2);
            count[2] = 0;
            currentSize--;
         }

         System.out.print(i+": ");
       //  display(customers);
         display(merge(service1, service2, service3, customers));
      }

      int counter = 1079;
      while(!(customers.isEmpty()&&service1.isEmpty()&&service2.isEmpty()&&service3.isEmpty()))
      {
         if(!service1.isEmpty())
         {
            count[0]++;
            if(servingTime[0]==count[0])
            {
               int x = counter - service1.remove();
               totalWait+=x;
               if(x>longestWait)
                  longestWait = x;
            }
         }
         if(!service2.isEmpty())
         {
            count[1]++;
            if(servingTime[1]==count[1])
            {
               int x = counter - service2.remove();
               totalWait+=x;
               if(x>longestWait)
                  longestWait = x;
            }
         }
         if(!service3.isEmpty())
         {
            count[2]++;
            if(servingTime[2]==count[2])
            {
               int x = counter - service3.remove();
               totalWait+=x;
               if(x>longestWait)
                  longestWait = x;
            }
         }


         if(service1.isEmpty()&&!customers.isEmpty())
         {
            service1.add(customers.remove());
            servingTime[0] = (int)(Math.random()*6+2);
            count[0] = 0;
            currentSize--;
         }
         if(service2.isEmpty()&&!customers.isEmpty())
         {
            service2.add(customers.remove());
            servingTime[1] = (int)(Math.random()*6+2);
            count[1] = 0;
            currentSize--;
         }
         if(service3.isEmpty()&&!customers.isEmpty())
         {
            service3.add(customers.remove());
            servingTime[2] = (int)(Math.random()*6+2);
            count[2] = 0;
            currentSize--;
         }

         System.out.print(counter+": ");
       //  display(customers);
         display(merge(service1, service2, service3, customers));
         counter++;
      }

      System.out.println("Total customers served = " + numberCustomers);
      System.out.println("Average wait time = " + (double)totalWait/numberCustomers);
      System.out.println("Longest wait time = " + longestWait);
      System.out.println("Longest queue = " + longestQueue);
   }
   public static Queue<Integer> merge(Queue<Integer> a, Queue<Integer> b, Queue<Integer> c, Queue<Integer> d)
   {
      Queue<Integer>  temp = new LinkedList<Integer>();
      Queue<Integer>  temp1 = new LinkedList(a);
      Queue<Integer>  temp2= new LinkedList(b);
      Queue<Integer>  temp3= new LinkedList(c);
      Queue<Integer>  temp4 =new LinkedList(d);


      while(!temp1.isEmpty())
         temp.add(temp1.remove());
      while(!temp2.isEmpty())
         temp.add(temp2.remove());
      while(!temp3.isEmpty())
         temp.add(temp3.remove());
      while(!temp4.isEmpty())
         temp.add(temp4.remove());

      return temp;
   }
   public static void display(Queue<Integer> q)
   {
      System.out.println(q);
   }
}

```