

Homework Turnin

Account: 6G_06 (rgalanos@fcps.edu)
Section: 6G
Course: TJHSST APCS 2016-17
Assignment: 06-08
Receipt ID: 38d92ea6a910c4556d45d64a103e0a6e

Warning: Your turnin is 1 day late. Assignment 06-08 was due Monday, January 30, 2017, 12:00 AM.

Turnin Successful!

The following file(s) were received:

McDonaldVIP.java (8015 bytes)

```
1. //name :   date:
2.
3. import java.util.*;
4. public class McDonaldVIP
5. {
6.     public static final int TIME = 1079; //18 hrs * 60 min
7.     public static void main(String[] args)
8.     {
9.         int numberCustomers = 0;
10.        int totalWait = 0;
11.        int longestWait = 0;
12.        int currentSize = 0;
13.        int longestQueue = 0;
14.        int numberVIP = 0;
15.        int VIPWait = 0;
16.        int[] servingTime = {100, 100, 100};
17.        int[] count = {0, 0, 0};
18.        boolean[] isVIP = {false, false, false};
19.
20.        Queue<Integer> customers = new LinkedList<Integer>();
21.        Queue<Integer> VIP = new LinkedList<Integer>();
22.        Queue<Integer> service1 = new LinkedList<Integer>();
23.        Queue<Integer> service2 = new LinkedList<Integer>();
24.        Queue<Integer> service3 = new LinkedList<Integer>();
25.
26.        for(int i=0; i<TIME; i++)
27.        {
28.            if(Math.random()<0.75)
29.            {
30.                customers.add(new Integer(i));
31.                currentSize++;
32.                numberCustomers++;
33.                if(currentSize>longestQueue)
34.                    longestQueue = currentSize;
35.            }
36.            else if(Math.random()<0.01)
37.            {
38.                VIP.add(new Integer(i));
39.                currentSize++;
40.                numberCustomers++;
41.                numberVIP++;
42.                if(currentSize>longestQueue)
```

```

43.         longestQueue = currentSize;
44.     }
45.
46.     if(!service1.isEmpty())
47.     {
48.         count[0]++;
49.         if(servingTime[0]==count[0])
50.         {
51.             int x = i - service1.remove();
52.             totalWait+=x;
53.             if(isVIP[0])
54.                 VIPWait+=x;
55.             if(x>longestWait)
56.                 longestWait = x;
57.         }
58.     }
59.     if(!service2.isEmpty())
60.     {
61.         count[1]++;
62.         if(servingTime[1]==count[1])
63.         {
64.             int x = i - service2.remove();
65.             totalWait+=x;
66.             if(isVIP[1])
67.                 VIPWait+=x;
68.             if(x>longestWait)
69.                 longestWait = x;
70.         }
71.     }
72.     if(!service3.isEmpty())
73.     {
74.         count[2]++;
75.         if(servingTime[2]==count[2])
76.         {
77.             int x = i - service3.remove();
78.             totalWait+=x;
79.             if(isVIP[2])
80.                 VIPWait+=x;
81.             if(x>longestWait)
82.                 longestWait = x;
83.         }
84.     }
85.
86.
87.     if(service1.isEmpty()&&!customers.isEmpty())
88.     {
89.         if(VIP.isEmpty())
90.         {
91.             service1.add(customers.remove());
92.             isVIP[0] = false;
93.         }
94.         else
95.         {
96.             service1.add(VIP.remove());
97.             isVIP[0] = true;
98.             servingTime[0] = (int)(Math.random()*6+2);
99.             count[0] = 0;
100.            currentSize--;
101.        }
102.        if(service2.isEmpty()&&!customers.isEmpty())
103.        {
104.            if(VIP.isEmpty())
105.            {
106.                service2.add(customers.remove());
107.                isVIP[1] = false;
108.            }
109.            else
110.            {
111.                service2.add(VIP.remove());
112.                isVIP[1] = true;
113.            }
114.            servingTime[1] = (int)(Math.random()*6+2);
115.            count[1] = 0;
116.            currentSize--;
117.        }
118.        if(service3.isEmpty()&&!customers.isEmpty())
119.        {
120.            if(VIP.isEmpty())
121.            {
122.                service3.add(customers.remove());
123.                isVIP[2] = false;

```

```

124.     }
125.     else
126.     {
127.         service3.add(VIP.remove());
128.         isVIP[2] = true;
129.     }
130.     servingTime[2] = (int)(Math.random()*6+2);
131.     count[2] = 0;
132.     currentSize--;
133. }
134.
135. System.out.print(i+": ");
136. // display(customers);
137. display(merge(service1, service2, service3, VIP, customers));
138. }
139.
140. int counter = 1079;
141. while(!(customers.isEmpty()&&service1.isEmpty()&&service2.isEmpty()&&service3.isEmpty()&&VIP.isEmpty()))
142. {
143.     if(!service1.isEmpty())
144.     {
145.         count[0]++;
146.         if(servingTime[0]==count[0])
147.         {
148.             int x = counter - service1.remove();
149.             totalWait+=x;
150.             if(isVIP[0])
151.                 VIPWait+=x;
152.             if(x>longestWait)
153.                 longestWait = x;
154.         }
155.     }
156.     if(!service2.isEmpty())
157.     {
158.         count[1]++;
159.         if(servingTime[1]==count[1])
160.         {
161.             int x = counter - service2.remove();
162.             totalWait+=x;
163.             if(isVIP[1])
164.                 VIPWait+=x;
165.             if(x>longestWait)
166.                 longestWait = x;
167.         }
168.     }
169.     if(!service3.isEmpty())
170.     {
171.         count[2]++;
172.         if(servingTime[2]==count[2])
173.         {
174.             int x = counter - service3.remove();
175.             totalWait+=x;
176.             if(isVIP[2])
177.                 VIPWait+=x;
178.             if(x>longestWait)
179.                 longestWait = x;
180.         }
181.     }
182.
183.     if(service1.isEmpty()&&!customers.isEmpty())
184.     {
185.         if(VIP.isEmpty())
186.         {
187.             service1.add(customers.remove());
188.             isVIP[0] = false;
189.         }
190.         else
191.         {
192.             service1.add(VIP.remove());
193.             isVIP[0] = true;
194.         }
195.         servingTime[0] = (int)(Math.random()*6+2);
196.         count[0] = 0;
197.         currentSize--;
198.     }
199.     if(service2.isEmpty()&&!customers.isEmpty())
200.     {
201.         if(VIP.isEmpty())
202.         {
203.             service2.add(customers.remove());
204.

```

```

205.         isVIP[1] = false;
206.     }
207.     else
208.     {
209.         service2.add(VIP.remove());
210.         isVIP[1] = true;
211.     }
212.     servingTime[1] = (int)(Math.random()*6+2);
213.     count[1] = 0;
214.     currentSize--;
215. }
216. if(service3.isEmpty()&&!customers.isEmpty())
217. {
218.     if(VIP.isEmpty())
219.     {
220.         service3.add(customers.remove());
221.         isVIP[2] = false;
222.     }
223.     else
224.     {
225.         service3.add(VIP.remove());
226.         isVIP[2] = true;
227.     }
228.     servingTime[2] = (int)(Math.random()*6+2);
229.     count[2] = 0;
230.     currentSize--;
231. }
232.
233.     System.out.print(counter+": ");
234.     // display(customers);
235.     display(merge(service1, service2, service3, VIP, customers));
236.     counter++;
237. }
238.
239. System.out.println("Total customers served = " + numberCustomers);
240. System.out.println("Average wait time = " + (double)totalWait/numberCustomers);
241. System.out.println("Longest wait time = " + longestWait);
242. System.out.println("Longest queue = " + longestQueue);
243. System.out.println("Number of VIPs = " + numberVIP);
244. System.out.println("Average VIP wait time = " + (double)VIPWait/numberVIP);
245. }
246. public static Queue<Integer> merge(Queue<Integer> a, Queue<Integer> b, Queue<Integer> c, Queue<Integer> d, Queue<Integer> e)
247. {
248.     Queue<Integer> temp = new LinkedList<Integer>();
249.     Queue<Integer> temp1 = new LinkedList<Integer>(a);
250.     Queue<Integer> temp2 = new LinkedList<Integer>(b);
251.     Queue<Integer> temp3 = new LinkedList<Integer>(c);
252.     Queue<Integer> temp4 = new LinkedList<Integer>(d);
253.     Queue<Integer> temp5 = new LinkedList<Integer>(e);
254.
255.     while(!temp1.isEmpty())
256.         temp.add(temp1.remove());
257.     while(!temp2.isEmpty())
258.         temp.add(temp2.remove());
259.     while(!temp3.isEmpty())
260.         temp.add(temp3.remove());
261.     while(!temp4.isEmpty())
262.         temp.add(temp4.remove());
263.     while(!temp5.isEmpty())
264.         temp.add(temp5.remove());
265.
266.     return temp;
267. }
268.
269. public static void display(Queue<Integer> q)
270. {
271.     System.out.println(q);
272. }
273. }
274.

```