Casey Sobecks
Assignment 5: LaFood
Output:

```
PS C:\Users\csobe\OneDrive\Documents\Butler\(13) 2020
Spring\CS248\Assignment 5\Try2> java LaFood

*** Welcome to LaFood Restaurant Simulator ***
Enter data file name:
data.txt
Please wait at the bar, party Merlin of 3 people. (time=3)
Please wait at the bar, party Arthur Pendragon of 2 people. (time=8)
Tabel for  Merlin! (time=10)
Please wait at the bar, party Sir Lancelot of 2 people. (time=12)
Tabel for  Arthur Pendragon! (time=15)
Please wait at the bar, party The Green Knight of 3 people. (time=17)
Tabel for  Sir Lancelot! (time=20)
** Simulation Terminated **

The average waiting time was: 7.0
The following parties were never seated:
party  The Green Knight of 3 people



PS C:\Users\csobe\OneDrive\Documents\Butler\(13) 2020
Spring\CS248\Assignment 5\Try2>
```

A 3 3 Merlin
A 8 2 Arthur Pendragon
T 10
A 12 2 Sir Lancelot
T 15
A 17 3 The Green Knight
T 20
Q

```
1  public interface Interface
2  {
3      public void enqueue(Object x);
4      public Object dequeue();
5      public Object getFront();
6      public int size();
7      public boolean isEmpty();
8      public boolean isFull();
9      public void makeEmpty();
10 }
```

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

class LaFood
{
    public static void main(String [] args) throws IOException
    {
        int action=1;
        int waittime[]= new int[10];
        int waitcount=0;
        System.out.println("\n*** Welcome to LaFood Restaurant Simulator ***");
        System.out.println("Enter data file name: ");
        Scanner input=new Scanner(System.in);
        String file=input.next();
        input.close();
        Scanner reader=new Scanner(new FileReader(file));
        Queue X=new Queue();


        while(action!=0)
        {
            X.enqueue(new List(reader));
            String a=((List)X.info()).getEvent();
            if(a.equals("A"))
            {
                int b=((List)X.info()).getTime();
                int c=((List)X.info()).getSize();
                String d=((List)X.info()).getName();
                System.out.println("Please wait at the bar, party"+d+" of "+c+"
people. (time="+b+")");
            }
            else if(a.equals("T"))
            {
                int weehoo=0;
                int e=((List)X.info()).getTime();
                while(weehoo==0)
                {
                    Object holder=((List)X.dequeue());
                    int f=((List)holder).getTime();
                    String h=((List)holder).getName();
                    if(h==null)
                    {}
                    else
                    {
                        waittime[waitcount]=e-f;
                        waitcount++;
                        System.out.println("Tabel for "+h+"! (time="+e+")");
                        weehoo++;
                    }
                }
            }
            if(a.equals("Q"))
            {
                action=0;
            }
        }
        System.out.println("** Simulation Terminated **\n");

        int hold=0;
        for(int i=0;i<waitcount;i++)
        {
```

```
61              hold=hold+waittime[i];
62          }
63          double waitav=hold/waitcount;
64          System.out.println("The average waiting time was: "+waitav);
65          System.out.println("The following parties were never seated:");
66          action=1;
67          while(action!=0)
68          {
69              List g=(List)X.dequeue();
70              if(g.getEvent().equals("A"))
71              {
72                  g.print();
73              }
74              else if(g.getEvent().equals("Q"))
75              {
76                  action=0;
77              }
78              else{}
79          }
80          System.out.println("\n\n");
81      }
82  }
```

```java
import java.util.*;
public class List
{
    //data members(variables)
    String event;
    int time;
    int size;
    String name;

    public static int count=0;

    //constructor
    public List(Scanner reader)
    {
        event=reader.next();
        if(event.equals("Q"))
        {

        }
        else if(event.equals("T"))
        {
            time=reader.nextInt();
            reader.nextLine();
        }
        else
        {
            time=reader.nextInt();
            size=reader.nextInt();
            name=reader.nextLine();
        }
    }

    public void print()
    {
        if(event.equals("A"))
        {
            System.out.println("party "+name+" of "+size+" people");
        }
        else{}
    }

    public String getEvent() {return event;}
    public int getTime()
    {
        if(event.equals("Q"))
        {
            return -1;
        }
        return time;
    }
    public int getSize()
    {
        if(event.equals("A"))
        {return size;}
        return -1;
    }
    public String getName()
    {
        if(event.equals("A"))
        {return name;}
        return null;
```

```
62      }
63  }
```

```
1  public class Queue implements Interface
2  {
3      Object [] waitlist;
4      int front, back, count;
5
6      public Queue(int maxsize)
7      {
8          waitlist= new Object[maxsize];
9          front=0;
10         back=maxsize-1;
11         count=0;
12     }
13
14     public Queue()
15     {
16         this(50);
17     }
18
19     public void enqueue(Object x)
20     {
21         if(isFull()) return;
22         back++;
23         if(back>=waitlist.length) back=0;
24         waitlist[back]=x;
25         count++;
26     }
27
28     public Object dequeue()
29     {
30         if(isEmpty()) return null;
31         int oldfront=front;
32         front++;
33         if(front>=waitlist.length) front=0;
34         count=count-1;
35         return waitlist[oldfront];
36     }
37
38     public Object info()
39     {
40         if(isEmpty()) return null;
41         return waitlist[back];
42     }
43
44     public Object getFront()
45     {
46         return isEmpty()?null:waitlist[front];
47     }
48     public int size() {return count;}
49     public boolean isEmpty() {return count<=0;}
50     public boolean isFull() {return count>= waitlist.length;}
51     public void makeEmpty()
52     {
53         front=0;
54         back=waitlist.length-1;
55         count=0;
56     }
57
58 }
```