```
import java.util.Deque;
import java.util.LinkedList;

/**
 * This is my code!  Its goal is to create a seating plan for
 *    people in a cruise
 * CS 312 - Assignment 8
 * @author Mari Sisco
 */

public class DiningHall
{
  protected Deque<Table> tables;

  // create method in ood
  public DiningHall()
  {
    this.tables = new LinkedList<>();
  }

  public void addTable(Table t)
  {
    tables.add(t);
  }

  public Table lookupTable(Person p)
  {
    for(Table t: tables)
    {
       if(t.lookupPerson(p))
         return t;
    }
    return null;
  }

  public void placeGroup(Group g)
  {
    Table t = findTableAgeRange(g.averageAge(), g.size());
    if (t == null)
    {
      t = findTableEmpty(g.size());
    }
    if (t == null)
    {
      t = findTableWithFreeSeats(g.size());
    }
    if (t == null)
    {
      return;
    }

    for(Person p : g.people)
    {
      t.addPerson(p);
    }
  }

  //toString of seating chart
  public String seatingChart()
  {
    String s = "";
    for(Table t: tables)
      s += t + "\n";
    return s;
  }

  private boolean ageMatch(Table t, int targetAge, int seatsNeeded)
  {
    boolean x = false;
```



```
    if(t.averageAge() <= (targetAge + 5) && t.averageAge() >= (targetAge - 5))
      x = true;

    return !t.isEmpty() && t.spaceFor(seatsNeeded) && x;
  }

  private boolean zeroOccupants(Table t, int seatsNeeded)
  {
    return t.isEmpty() && t.spaceFor(seatsNeeded);
  }

  private boolean freeSeats(Table t, int seatsNeeded)
  {
    return t.spaceFor(seatsNeeded);
  }

  public Table findTableAgeRange(int targetAge, int seatsNeeded)
  {
    for(Table t: tables)
    {
      if(ageMatch(t, targetAge, seatsNeeded))
        return t;
    }
    return null;
  }

  public Table findTableEmpty(int seatsNeeded)
  {
    for(Table t: tables)
    {
       if(zeroOccupants(t, seatsNeeded))
         return t;
    }
    return null;
  }

  public Table findTableWithFreeSeats(int seatsNeeded)
  {
    for(Table t: tables)
    {
      if(freeSeats(t, seatsNeeded))
        return t;
    }
    return null;
  }
}
```

```
/**
 * This is my code!  Its goal is to create a table based on groups
 *   and average ages
 * CS 312 - Assignment 8
 * @author Mari Sisco
 */
import java.util.Deque;
import java.util.ArrayDeque;
import java.util.LinkedList;

public class Table
{
    protected Deque<Person> occupants;
    protected int seatsAvailable;
    protected String tableName;

    // constructor is the create method
    public Table(int seats, String name)
    {
        occupants = new ArrayDeque<>();
        seatsAvailable = seats;
        tableName = name;
    }
    public void addPerson(Person p)
    {
        occupants.add(p);
        seatsAvailable--;
    }

    public int averageAge()
    {
        int sum = 0;
        for( Person p: occupants)
        {
            sum += p.age;
        }
    if (occupants.size() == 0)
      return 0;
        return sum/(occupants.size());
    }

    // display method
    public String toString()
    {
        String s = tableName + ":(average age " + averageAge() + ")";
        for(Person p: occupants)
        {
            s += p + " ";
        }
        return s;
    }

    public boolean isEmpty()
    {
        return this.occupants.size() == 0;
    }

    public boolean lookupPerson(Person lost)
    {
        for(Person p: occupants)
        {
            if (p.compareTo(lost) == 0)
                return true;
        }
        return false;
    }

    public boolean spaceFor(int need)
    {
```



```
        return seatsAvailable >= need;
    }

}
```

```
/**
 * This is my code!  Its goal is to create a group of people
 * CS 312 - Assignment 8
 * @author Mari Sisco
 */


import java.util.Deque;
import java.util.ArrayDeque;

public class Group
{
    protected Deque<Person> people;

    public Group()
    {
        people = new ArrayDeque<Person>();

    }

    public void addPerson(Person p)
    {
        people.add(p);
    }

    public int averageAge()
    {
        int sum = 0;
        for(Person p: people)
            sum = sum + p.age;
        return sum/size();
    }

    public int size()
    {
        return people.size();
    }

    public String toString()
    {
        String s = "(average age " + averageAge() + ")";

        for(Person p: people)
            s += "-> " + p + "\n";

        return s;
    }
}
```



```
/**
 * This is my code!  Its goal is to create a person, with a name
 *    and age
 * CS 312 - Assignment 8
 * @author Mari Sisco
 */

public class Person
{
    protected int age;
    protected String name;

    public Person(String name, int age)
    {
        this.name = name;
        this.age = age;
    }

    //display method
    public String toString()
    {
        return name +"(age " + age + ")";
    }

    public int compareTo(Person otherGuy)
    {
        return name.compareTo(otherGuy.name);
    }
}
```