



COMP 1230 – Programming 2

Assignment #8 Generics

Vu Nguyen

T00612390

Nov, 19

Problem:

The purpose of this problem is to practice using a generic Jar class.

Create a generic class, called Jar, with a type parameter that simulates drawing an item at random out of a Jar. For example the Jar might contain Strings representing names written on a slip of paper, or the Jar might contain integers representing a random drawing for a lottery. Include the following methods in your generic class, along with any other methods you'd like:

- an add() method that allows the user to add **one** object of the specified type
- an isEmpty() method (returns true if the Jar is empty, otherwise returns false)
- a drawItem() method that randomly selects an object from the Jar and returns it. Return null if the Jar is empty. Do not delete the selected item. (see Note below)
- a toString() method (returns a String containing the Jar's contents)

Requirements:

Your Jar class will need an array of size 10 to hold the objects in the Jar, and a count variable to maintain a count of how many objects are actually in the Jar.

In the driver file that tests your class create 2 Jars, one with the names of 6 of your friends, the other with numbers between 2 and 8 inclusive representing the number of hours you will spend partying tonight with 3 of your friends.

Use the add() method to populate the 2 Jars, and the drawItem() method for each Jar to determine i) which 3 friends you will invite out to party with and ii) how many hours of partying you and your friends will do.

Test Cases:

Test all methods- add, drawItem, toString, isEmpty

Test Ctor and make sure it's doing what a Generic type should do

Test for random in names and hours drawing

```

1  /*
2  * Student name: Vu Nguyen
3  * Student Number: T00612390
4  * Due Date: Nov 18
5  * Program Description: This program is to simulate the action of drawing out items
   from a hat
6  *           ie: in this case its being used to create a "jar" filled with names and a "
   jar"
7  *           filled with durations of time of which you will spend with these names.
8  *
9  */
10 import java.util.Random;
11 import java.util.Arrays;
12
13 /**
14  * The type Jar.
15  *
16  * @param <T> the type parameter
17  */
18 public class Jar<T>
19 {
20
21     private T list[];
22     private int count;
23
24     /**
25      * Instantiates a new Jar.
26      */
27     //creates 10 empty Jar spaces
28     public Jar()
29     {
30         list = (T[])new Object[10];
31         count=0;
32     }
33
34
35     /**
36      * Add.
37      *
38      * @param items the items
39      */
40     //add--adds items to list if space permits
41     public void add(T items)
42     {
43         //checks if list has space to add more
44         if (list.length > count)
45         {
46
47             list[count] = items;
48             count++;
49
50         }

```

```

51     }
52
53     /**
54      * Is empty boolean.
55      *
56      * @return the boolean
57      */
58     //isEmpty checks if the array is empty
59     public boolean isEmpty()
60     {
61         return (count == 0);
62     }
63
64
65     /**
66      * Draw item t.
67      *
68      * @return the t
69      */
70     public T drawItem()
71     {
72         //checks if array is populated or not
73         if (isEmpty())
74         {
75             return null;
76         }
77         Random random = new Random();
78         int i = random.nextInt(count);
79         //returns every item in list
80         return list[i];
81     }
82
83 }
84
85 @Override
86 public String toString()
87 {
88     T items[] = (T[]) new Object[count];
89
90     for (int i = 0; i < count; i++)
91     {
92         items[i] = list[i];
93         return Arrays.toString(items);
94     }
95 }
96
97
98
99 }

```

```

1
2  /*
3   * Student name: Vu Nguyen
4   * Student Number: T00612390
5   * Due Date: Nov 18
6   * Program Description: This file is the driver program for Jar
7   *
8   */
9  public class JarDriver
10 {
11     /**
12      * The entry point of application.
13      *
14      * @param args the input arguments
15      */
16     public static void main(String[] args)
17     {
18         //creates Jar for friends and hour
19         Jar<String> friendsJar = new Jar<String>();
20
21         Jar<Integer> hoursJar = new Jar<Integer>();
22
23         //sees if isEmpty is working for both Jars
24         System.out.println("Checking if friendsJar is empty: " + friendsJar.isEmpty());
25         System.out.println("Checking if hoursJar is empty: " + hoursJar.isEmpty());
26
27         // add elements to friendJar
28         friendsJar.add("Karen");
29         friendsJar.add("Mike");
30         friendsJar.add("Pike");
31         friendsJar.add("Susan");
32         friendsJar.add("Bob");
33         friendsJar.add("Dell");
34
35         // add elements to hoursJar
36         hoursJar.add(2);
37         hoursJar.add(3);
38         hoursJar.add(4);
39         hoursJar.add(5);
40         hoursJar.add(6);
41         hoursJar.add(7);
42         hoursJar.add(8);
43
44         System.out.println("itms have been added, Checking if hoursJar is empty: " +
45             hoursJar.isEmpty());
46
47         // draw 3 items from friendsJar
48         System.out.println("Picking 3 friends from Jar: " + friendsJar.drawItem() + " " +
49             friendsJar.drawItem() + " " + friendsJar.drawItem());
50
51         // draw one item from hoursJar
52         System.out.println("Picking time from Jar: " + hoursJar.drawItem());

```

```
52
53     // display the items in friendsJar, hoursJar
54     System.out.println("\nPrinting friendsJar without toString : \n" +friendsJar);
55     System.out.println("Printing hoursJar without toString : \n" +hoursJar);
56
57     System.out.println("testing toString for friendsJar " + friendsJar.toString());
58     System.out.println("testing toString for hoursJar " + hoursJar.toString());
59 }
60
61
62 }
63
```

```
Checking if friendsJar is empty: true
Checking if hoursJar is empty: true
Items have been added, Checking if hoursJar is empty: false
Picking 3 friends from Jar: Mike Dell Karen
Picking time from Jar: 3

Printing friendsJar without toString :
[Karen, Mike, Pike, Susan, Bob, Dell]
Printing hoursJar without toString :
[2, 3, 4, 5, 6, 7, 8]
testing toString for friendsJar [Karen, Mike, Pike, Susan, Bob, Dell]
testing toString for hoursJar [2, 3, 4, 5, 6, 7, 8]
```



```
Checking if friendsJar is empty: true
Checking if hoursJar is empty: true
items have been added, Checking if hoursJar is empty: false
Picking 3 friends from Jar: Karen Pike Susan
Picking time from Jar: 6

Printing friendsJar without toString :
[Karen, Mike, Pike, Susan, Bob, Dell]
Printing hoursJar without toString :
[2, 3, 4, 5, 6, 7, 8]
testing toString for friendsJar [Karen, Mike, Pike, Susan, Bob, Dell]
testing toString for hoursJar [2, 3, 4, 5, 6, 7, 8]
```

```
Checking if friendsJar is empty: true
Checking if hoursJar is empty: true
Items have been added, Checking if hoursJar is empty: false
Picking 3 friends from Jar: Bob Karen Mike
Picking time from Jar: 3

Printing friendsJar without toString :
[Karen, Mike, Pike, Susan, Bob, Dell]
Printing hoursJar without toString :
[2, 3, 4, 5, 6, 7, 8]
testing toString for friendsJar [Karen, Mike, Pike, Susan, Bob, Dell]
testing toString for hoursJar [2, 3, 4, 5, 6, 7, 8]
```