**Familiarity Review Template**

**Name:** Lynn Dimick

**Date:** 22 Feb 2020

**Week:** 07

**Coding Topic: Java Collections**

**Description of Understanding:** I have really struggled with this topic. I have a difficult time finding applicable examples for some of these concepts.

In the first example, I created an array and added 7 members to it. These represent test scores. I listed the elements, and then I removed the 2nd element in the list (element[1]) using the remove method, and then redisplayed the contents of the array to show that the array had been reduced in content by one element.

For the second collection I demonstrated the use of the list collection. I created a list that contained the elements of 3, 9, 1, 4, 7, 2, 5, 3, 8, 9, 1, 3, 8, 6. Note that there are more than one element that matches another such as 1 and 3. I created a new HashSet and displayed its contents. The HashSet now shows just the unique elements and has removed any duplications.

For the third collection I created a TreeMap. The purpose of this demonstration was to show that you can use a key value, that correlates to a data point, and return the data point contents. IN this case I populated the Tree with several programming languages and their common file extensions. The first step was to validate that .java is a valid key in the tree. Once the key presence is verified, I removed the name JAVA from the tree and displayed the contents of the tree.

I have actually used this type of technology in the past with an automotive parts inventory management system. We used the part numbers to match up the description which was then used to print out the description on the Point of Sale invoice/receipt.

For the next collection I created a linked list. I then added three names by the use of the add function. I then removed one name by the use of the remove function and then displayed the contents.

For the final collection demonstration, I found that the tree was the most difficult to comprehend. However, because I have used BTRIEVE in a previous life I had a working knowledge, which helped. In this example I created a root node with the root value and the left and right node values. I inserted the first node value. When a second value was inserted the code checks to see if the value is lower than the current left node. The first check is to see if the new value is less than the root value. If the value in the left node is not null, and the new value is less than the root value, it places the value into the left node. Otherwise it checks the right node value.

When it arrives at the next level node it goes through the check process again and will create a new node depending upon the value in relation to the existing node.

My demo code shows each insertion and then displays the values in a sorted order. The original inserted order was 3,76,8,7,6,7,13,8.

The expected output for all demo’s is:

Display the List contents

Grade #0 == 90

Grade #1 == 100

Grade #2 == 80

Grade #3 == 85

Grade #4 == 75

Grade #5 == 80

Grade #6 == 90

Remove the 2nd element (Index[1])

Element 1 == 100

Display the List contents

Grade #0 == 90

Grade #1 == 80

Grade #2 == 85

Grade #3 == 75

Grade #4 == 80

Grade #5 == 90

Now Demo the Set. Create the list to contain:

[3, 9, 1, 4, 7, 2, 5, 3, 8, 9, 1, 3, 8, 6]

The SET now contains the unique members

[1, 2, 3, 4, 5, 6, 7, 8, 9]

{.c=C, .cpp=C++, .cs=C#, .java=Java, .php=PHP, .pl=Perl, .xml=XML}

.java: Java

.c: C

We have set up a map with some programming languages in the map

We see if .java is a valid key in our map. If it is we will delete it.

.java is a valid key

Remove .java from the map

Removed value: Java

{.c=C, .cpp=C++, .cs=C#, .php=PHP, .pl=Perl, .xml=XML}

Showing the content of the queue

[Mike, Melissa, Scott]

Showing the added name in the queue

[Mike, Melissa, Scott, Christy]

Now we will remove the first item in the queue

Next customer is: Mike

The queue now contains:

[Melissa, Scott, Christy]

Binary Tree Example

Building tree with root value 7

Inserted 3 to left of 7

Inserted 76 to right of 7

Inserted 8 to left of 76

Inserted 6 to right of 3

Inserted 13 to right of 8

Inserted 9 to left of 13

Traversing tree in order

3 6 7 8 9 13 76

**Teaching Video:**

**Starting at:**

**Also Integrated with:** Collections Example System Test

|  |  |  |  |
| --- | --- | --- | --- |
| **File** | **Git Link** | **What should I be looking for?** | **Sandbox or Your code?** |
| [CollectionBasics.java](https://github.com/Dimick-Lynn/CIT360/blob/master/collectionsdemo/CollectionBasics.java)  [Tree.java](https://github.com/Dimick-Lynn/CIT360/blob/master/collectionsdemo/Tree.java) | https://github.com/Dimick-Lynn/CIT360/blob/master/collectionsdemo | See output above | Mine |
| [Collections Example System Test.xlsx](https://github.com/Dimick-Lynn/CIT360/blob/master/Collections%20Example%20System%20Test.xlsx) | https://github.com/Dimick-Lynn/CIT360/blob/master/Collections%20Example%20System%20Test.xlsx | Results of System Test for the code above demonstrating Collections | Mine |
|  |  |  |  |

**Coding Topic:** Systems Test

**Description of Understanding:** The System test is a documented process of the steps and expected results for testing an application. The results are to be recorded and variances should be identified but not remediated at this point. In this case I prepared the tests for verifying the functioning of my Collections Demos.

**Teaching Video:**

**Starting at:**

**Also Integrated with: NA**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **File** | **Git Link** | | **What should I be looking for?** | | **Sandbox or Your code?** | |
| [Collections Example System Test.xlsx](https://github.com/Dimick-Lynn/CIT360/blob/master/Collections%20Example%20System%20Test.xlsx) | | <https://github.com/Dimick-Lynn/CIT360/blob/master/Collections%20Example%20System%20Test.xlsx> | | Results of System Test for the code above demonstrating Collections | | Mine |

**Coding Topic:** State Diagrams

**Description of Understanding:** This is the presentation that I gave in our group meeting. The key concept here is that the diagram shows the state or condition of something. It is not a flowchart. A system state could be the contents of a value, the position of a switch, a decision that was made, etc. A flow chart shows decision that must be made, a logical flow, etc. A state diagram is used to document a process or a device. For instance, in slide 5 of the deck it shows a printer. Each description is what the printer is doing and not what input the user has given. An input from the user might involve the steps needed to send a print job to the device. That input might include the source, the size of paper, the colors to use, the number of copies. In the state diagram we see that the printer is in standby mode until a print job is sent. Once that job is received we can begin the changes to the system that are needed such as changing the paper that is used, the colors, etc.

**Teaching Video:**

**Starting at:**

**Also Integrated with: NA**

|  |  |  |  |
| --- | --- | --- | --- |
| **File** | **Git Link** | **What should I be looking for?** | **Sandbox or Your code?** |
| System State Diagram.pptx | <https://github.com/Dimick-Lynn/CIT360/blob/master/System%20State%20Diagram.pptx> | PowerPoint Presentation | Mine |
|  |  |  |  |