Java Programming

2-4: Collections – Part 1

Practice Activities

Lesson Objectives:

- Create a collection without using generics
- Create a collection using generics
- Implement an ArrayList
- Implement a Set

Vocabulary:

Identify the vocabulary word for each definition below.

	A set similar to an ArrayList without any specific ordering.
	An ordered Collection that may contain duplicates.
	An interface used to define a group of objects. This includes lists and sets.
	A list that is very similar to an array.
	A Collection of elements that does not contain any duplicates.

Try It/Solve It:

1. What is the difference between a set and a list?

2. You decide you want to roll 2 dice and see what the frequency is of each possible number combination. Would you use a Set collection to do this? State your reason(s).

3. Using a collection create a variable that will store a list of countries (Strings). Your collection should not store duplicates, and order is not important. Test your code by adding 6 countries, one of which is a duplicate.

4. Would the following Collection.sort() statements both work? Explain your answer.

```
HashSet<String> countriesSet = new HashSet<String>();

Collections.sort(countriesSet);

ArrayList<String> countriesList = new ArrayList();

Collections.sort(countriesList);
```

5. Below is a user implementation of a Stack using arrays.
   - push adds an item to the Stack
   - pop removes an item from the stack
   - isEmpty return a Boolean value of true if the Stack is empty

   Convert this to a generic implementation using an ArrayList.

```java
public class ArrayStack {
    private int maxsize;
    private int top;
    private int[] items;

    public ArrayStack(int maxsize) {
        if (maxsize <= 0)
            throw new ArrayStackException(
                        "Stack size must be positive");
        items = new int[maxsize];
        this.maxsize = maxsize;
        top = 0;
    }

    public void push(int item) {
        if (top == items.length)
            throw new ArrayStackException("Overflow Error");
        items[top] = item;
        top++;
    }


    public int pop() {
        if (isEmpty())
            throw new ArrayStackException("Underflow Error");
        return items[--top];
    }

    public boolean isEmpty() {
        return (top == 0);
    }

    public static class ArrayStackException extends RuntimeException {
        public ArrayStackException(String message) {
            super(message);
        }
    }


    public static void main(String[] args) {
        ArrayStack stack = new ArrayStack(3);
        stack.push(1);
        stack.push(2);
        stack.push(3);
        //stack.push(4);  //overflow error
        System.out.println(stack.pop());
        System.out.println(stack.pop());
        System.out.println(stack.pop());
```

```
                    }
            }
```