ISM3230 In-class lab Week 8, Module 6 – Working with Arrays Fall 2015

FOCUS TOPICS

- Arrays: storing multiple values of the same type

TASK

Your task is to take set of potential mountains, and determine whether they meet the minimum height requirement of 1,000 feet (based on http://science.nationalgeographic.com/science/earth/surface-of-the-earth/mountains-article/).

As input, ask the user to enter the number of potential mountains to be checked. Then prompt the user to enter the name and height of each potential mountain in turn.

As output, report the original potential mountain and its height, and whether or not you found it to be an actual mountain.

Follow the instructions below carefully, and be sure to use the structures specified there. Specifically, the instructions below ask you to create and use three different arrays for this task. It is possible to avoid one of the arrays; however, we are practicing using arrays in today's lab, so please be sure to use all the arrays specified. **If you do not use all three arrays in your logic, please do not expect full credit for this lab.**

INSTRUCTIONS

1. Create a new NetBeans project.
2. Create a constant to represent the minimum mountain size.
3. Prompt the user to enter a number of mountains to be checked. Read the input from the keyboard, and store it in an appropriately typed variable.
4. Create three arrays sized using the user-entered number of potential mountains to be checked:
    - An array of Strings to store the potential mountain names
    - An array of integers to store the potential mountain heights
    - An array of booleans to store the decision (whether the potential mountain is in fact a mountain).
5. Initialize the arrays, such that:
    - Each member of the String array is an empty string (“”)
    - Each member of the integer array is 0
    - Each member of the boolean array is false
6. Gather user input to fill the arrays (potential mountain name and height for each of the potential mountains).
7. Test each potential mountain to determine whether it meets the minimum height requirement, and store the result in the Boolean array.
8. Following the sample output below, report each potential mountain's name and height, and whether the potential mountain meets the minimum height requirement.

9. When your code is working, upload your .java file to the Week 8, Module 6 in-class lab dropbox.

SAMPLE OUTPUT

```
How many potential mountains do you have?
2
Enter the name of potential mountain 1:
Washington
Enter the height of potential mountain 1:
9889
Enter the name of potential mountain 2:
Lincoln
Enter the height of potential mountain 2:
342
Mount Washington, at 9889 feet, meets the minimum height to qualify as a mountain
Lincoln Hill, at  342 feet, does not meet the minimum height to qualify as a mountain
BUILD SUCCESSFUL (total time: 19 seconds)
```

```
How many potential mountains do you have?
3
Enter the name of potential mountain 1:
Grant
Enter the height of potential mountain 1:
541
Enter the name of potential mountain 2:
Lee
Enter the height of potential mountain 2:
872
Enter the name of potential mountain 3:
Jackson
Enter the height of potential mountain 3:
955
Grant Hill, at  541 feet, does not meet the minimum height to qualify as a mountain
Lee Hill, at  872 feet, does not meet the minimum height to qualify as a mountain
Jackson Hill, at  955 feet, does not meet the minimum height to qualify as a mountain
BUILD SUCCESSFUL (total time: 1 minute 22 seconds)
```

```
How many potential mountains do you have?
2
Enter the name of potential mountain 1:
Snow
Enter the height of potential mountain 1:
5678
Enter the name of potential mountain 2:
Rain
Enter the height of potential mountain 2:
9982
Mount Snow, at 5678 feet, meets the minimum height to qualify as a mountain
Mount Rain, at 9982 feet, meets the minimum height to qualify as a mountain
BUILD SUCCESSFUL (total time: 20 seconds)
```