Programming Assignment 3: OrchardStroll

Dr. Sharma Thankachan

COP 3503 – Fall 2018

Due Sunday, October 21st, at 11:59pm

1. Deliverable:

A single source file named *OrchardStroll.java* written in Java.

2. Objective:

The objective of this assignment is to test your understanding of dynamic programming and memoization. It includes some pre-written code that provides an example of clean, object-oriented Java.

3. Description:

4. Requirements:

You must write your assignment in the provided *OrchardStroll.java* source file. Do not edit the method signature that is provided. You may, however, write additional methods as you see fit. I suggest that any additional methods you add be static, and that you avoid using static class variables (unless you really, *really* know what you're doing with them).

```
public static int determineLargestHaul(Tree[] trees)
```

This method must determine the largest gross weight of fruit that can be collected from the provided array of trees using only 2 baskets. This method should then return that value as an integer. Your solution must employ dynamic programming, and it must be iterative, not recursive. See section 3 for details on how to collect fruit from this tree array.

NOTE: If you submit a recursive solution that does not employ dynamic programming, you will time out on all the grading test cases.

NOTE: If you submit a dynamic programming solution that uses recursion, your stack will overflow on all the grading test cases.

NOTE: You are free to write additional methods and make method calls in your solution. You do not need to worry about overflowing with a handful of nested method calls. The recursion overflow will only occur if you make 100's of recursive calls... (the grading test cases will be huge).

5. Input Specification:

The *determineLargestHaul()* function will receive an array of Tree objects with a length in the range [1, ∞). The array will **not** contain any null trees. Every tree will have a non-null type and will have a weight in the range [1, 1,000,000]. The different types of trees available can be found in an enumeration within the Tree class. Finally, the final answer for this function will **never** be so large that it overflows.

6. Testing:

To test, you will want to run the provided test script in a Linux environment. Your assignment will be graded on Eustis, and if your assignment does not work on Eustis then it **does not work.** For the script to function, you will need to set up your directory (folder) to look like this:

```
TestSuite/
        Test.sh
        Tree.java
        OrchardStroll.java
        OrchardStrollTester.java
```

To run the test script, use this command:

```
$ bash Test.sh
```

NOTE: The "$" character is not part of the command. It is a symbol frequently used to indicate the beginning of a terminal command.

7. Grading Criteria:

There are 10 tests for this assignment and each test is weighed equally. The final assignment score will be out of 100. Here is how the tests are divided:

10/10 *determineLargestHaul()* unit tests

8. Final Notes:

Comments will not be graded generally, however I reserve the right to penalize assignments that include no comments, or whose comments are horrible. In the real world, comments are extremely useful and expected from developers, so get used to writing them. I also reserve the right to penalize assignments with atrocious/inconsistent style (or lack thereof). This does not mean that you will lose points for using one particular style over another—it only means that you need to apply your style consistently.

This assignment must be done individually. You cannot review or use code from peers outside of your group, but you may have high-level discussions about this assignment with any of your peers.

Examples of **acceptable** peer questions about this assignment look like this:

- How do you convert a number in base-10 to base-60 on paper?
- What data structures are most useful when implementing radix sort?
- Can you show me how to use Horner's Rule to calculate the base-10 version of a base-60 number on paper?

Examples of **unacceptable** peer questions about this assignment look like this:

- What does the code look like to convert a number from base-10 to base-60?
- I have an off-by-one error in my radix sort code, can you help me find the bug?
- Can you show me how to code up Horner's Rule?

You MUST do all of the following:

- Include your name(s) and NID(s) in a comment at the top of your submission.
- Ensure that your submission compiles and runs on Eustis.
- Avoid including your single source file in a package.
- Name your submission *BabylonianSort.java.*
- Submit the assignment before the deadline.
- Write the assignment in Java.

If you do not follow the guidelines above you will receive a 0.