Software Development

Mini Assignment 1
MARKING SHEET

This document begins by summarizing important grading issues. Then it provides some guidance to what a correct solution looks like. I am not providing a complete solution.

IMPORTANT MARKING ELEMENTS:

- Practices the software techniques covered in lecture #2
- Practice some software techniques from COMP 250.
- Students may be aware of other software techniques. They must not use them.
- Algorithms should have the following qualities: optimality in speed (Big Oh) and memory (bytes), simplicity of solution, correctness, robustness, easy to read code, and commenting as documentation.
- When grading, it is important to look at lecture 3, to see what I am not expecting them to have in this assignment.

WHAT TO HAND IN

- PinMain.java
- PinTest.java
- Pin.java

HOW IT WILL BE GRADED

For your assignment to be graded your program (a) must run, (b) did not use tools, and (c) followed the assignment instructions. You are doing this assignment on your own.

- +5 - Optimality (Big Oh)
- +5 - Optimality (memory)
- +5 - Simplicity of solution
- +5 - Correctness (PinTest.java)
- +5 - Well written code that is easy to read (see lecture 2 for examples)
- +5 - Comments as documentation (see lecture 2 for examples)

PARTIAL SOLUTIONS

To answer this question correctly the students must decide (1) the optimal data structure, (2) the optimal algorithm, (3) following proper rules for constructing a Pin class, (4) using Pin properly from Main program and Test program, (5) Proper documentation and writing style.

(1) Optimal data structure

Candidate data structures

Int current is 4 bytes
Int digit1, digit2 is 8 bytes
Int current[2] is 12 bytes
Char digit1, digit2 is 2 bytes
Char current[2] is 6 bytes
String current is Total 10 bytes for ptr, size, chars
LinkedList current is Each node 8 bytes, head 4 bytes, Total 20 bytes.

Constraints on data structures:
Characters can be read in, therefore, using an integer solution means additional data structures when reading from keyboard.

Conclusion:
Even though using integer as our data structure results in one of the smaller sized data structures, the additional requirement of accepting characters in the input forces the use of another data structure for reading user input. If would be better to just use one data structure that could handle all cases.

Linked lists are too expensive and we don't need the capabilities it provides.

Best solution is char or String. Char is the most optimal but not the most flexible since the user input is not standardized.

Recommendation: character array or String. These can be sufficiently large to handle strange user input. Since they are both based on characters it accepts any kind of input. The character array solution is the most optimal. The String solution is easier to program given the Java string libraries. The string libraries also add more source code memory to the final solution.

If the student did the solution either way give them full points but give one bonus point for using array or characters.

(2) Optimal algorithm

The algorithms in this assignment are not very complicated. Everything should be O(n) or faster in Pin and in Main. The Test program does not need to be optimal.

(3) Following proper rules for constructing a Pin class

A properly constructed Pin class will reflect on how they are forced to write code in the Main and Test programs. Look at these other two classes as a validation that they constructed the Pin class properly. For example: all the methods they created for Pin are being used in Main and Test; the Pin data structure is either private or was public but not accessed from outside of the Pin class. The data

structure, if you removed the Pin class and put it into your own TA class, always protects the Pin data structure so that you can never put an incorrect value in that variable, where correctness is defined in the assignment question.

(4) Using Pin properly

(A) Even though the Main program only runs once, Pin's data structure must change at the end when the user enters the information correctly. If you run the program a second time, it will start from 00 once more.

(B) Test must exhaustively validate every good value, exhaustively validate every border value (each side of the border, or if there is an actual border number then just that value), you also need to validate incorrect values. However, if values are infinite the student can either (i) select 3 examples and test those, or (ii) use a loop and a random number generator to generate n bad values and test those, where n is much bigger than 3 but very much less than infinity.

(C) The internal variables of Pin should not be directly manipulated from Main or Test.

(5) Proper documentation and writing style

See lecture 2 and 3 for examples. The students need to follow those examples. Please take note that students can reduce the quantity of comments when they use good identifiers and good code formatting, but that cannot eliminate commenting.

Ask yourself this question when grading the student's comments: will this help a new developer when they come a month later to understand the code?

Do not forget the goals from the top of this marking sheet. They need to be factored in when you evaluate each assignment.