Williams Sonoma - Software Engineer - Backend End - Description & Coding Challenge

Please send these instructions to candidates and the attached zip folder. Please ask candidates to return the challenge with a github link.

**Williams Sonoma Backend Challenge for Java Developer**

Please make sure that the resume is an accurate representation of skill set and technologies used. Anything on the resume is fair game and if the candidate cannot speak to the point, they will likely be disqualified.

Have a solid answer to “Why are you interested in a job at Williams Sonoma?”

**Now…the Challenge**

The Zipcode challenge is a real world problem you may run into at WSI. It lists products – size and weights – and based on shipping requirements you can/cant send out to certain zipcodes – so write an algorithm to tell which shipments can go where.

**Instructions**

* The point of this exercise is to show your thought process in how you solve problems and structure code (**this should be a pure Java file**!)
* All code must be written from scratch, PLEASE do not plagiarize – they will check if it is copied and if so, this will disqualify you
* Please check your code for syntax errors
* Use good Java development standards and object oriented principles.
  + Use spaces instead of tabs
  + JavaDoc must be present and proper
  + Use of proper access modifiers
  + Variable naming must make sense. Use of variables like x, y, z and so on is a big deal (Big no)
  + Use of interfaces and not implementation
  + General use of good coding standards
  + Have them run Checkstyle/Sonar checks (at least Google’s)
  + Look for usage of streams / lambdas in a non-functional way - these must not alter state.
* Writing Unit tests are a **must have** and we recommend thorough unit tests! Multiple classes/files really impress the managers. It's the little details that often disqualify candidates.
* Notate your code so that it looks clean and your thought process clearly shows through.

**Additional Tips**

* The code should able to take user input.
  + For example, input:
  + ([49679, 52015], [49800, 50000], [51500, 53479], [45012, 46937], [54012, 59607], [45500, 45590], [45999, 47900], [44000, 45000], [43012, 45950])
  + Merged Result:
  + [43012, 47900], [49679, 53479], [54012, 59607]
* Please add JUNIT unit tests for you code.
* Please add detailed Javadoc for each class and important methods.
* Please add detailed log for input / output in the main/test code.
* Please add well documented readme file
* Add some screenshots if it’s necessary.