KCP QA Quiz

MAX TIME 24 HOURS.  PLEASE KEEP TRACK OF YOUR OWN TIME.

START TIME: 6/2/2011 5:00pm

END TIME: 6/3/2011 10:00am

**Directions:**

Should you have any questions about anything you are being asked in this written quiz then just make an assumption, write down what your assumption was, and answer the question.  It is important to us that you do not ask us for any guidance after you have started this process.  If you are not a strong coder then do your best and focus on the test part of the coding question.  The last question is testing only.

|  |
| --- |
| Write code in any language that given an array of integers returns an array of all the duplicated integers.  Write down any assumptions rather than asking questions.  How would you test your program? |
| See attached java implementation DuplicateDetector.java and DuplicateDetectorTest.java.  I realize using a dynamically typed language would have been way easier. In Python or Ruby you could use list comprehension, or a hash iterator, and in PHP array\_unique() would do the trick. But I had fun messing with all the Java type troubles and didn’t want to throw all my extra work away. Type safety makes the number of test scenarios a lot smaller too because you don’t have to worry about casting from strings, floats, etc.  I used a TestNG data provider, and was going to have it read from a spreadsheet but didn’t have the time to implement it. I could have used Fitnesse but it’s more difficult to work into continuous integration.  Assumptions:  If there are multiple duplicates, it only returns the number once.  It returns the matches unsorted (in the order found.) |

|  |
| --- |
| Write the function Boolean isPalindrome(String input) {}.  A palindrome can be read forwards or backwards and it has the same spelling.  Example:  “radar” = true, “blue” = false  How would you test it? |
| See attached PalindromeDetector.java and PalindromeDetectorTest.java  This test is structured similarly to the first one using TestNG and data providers. I created 4 categories of tests:   1. Valid palindromes 2. Invalid palindromes 3. Tricky cases that depend on assumptions in the implementation 4. Checking invalid types – which isn’t strictly necessary except the data provider is untyped   isPalindrome() simply checks: string.equals(reverse(string) I created three implementations of a private function reverse(). One does it the “hard” way  Assumptions:  It does a simple string comparison. Numbers, punctuation, etc. included  It can optionally ignore capitalization and spacing  Input is a valid String or null (using data provider requires check) |

Last Question:  Imagine Amazon making a web page where you can SMS/Text Message people from a web page on the Amazon.com site.  You might use this if you found a good deal or have a gift idea.  What is your test plan and please give us specific examples of how you would test it.  For example don’t just say “stress test it”.  You need to give us specifics on how YOU would stress test it.

|  |
| --- |
| **Amazon SMS Test Plan**  **Test Time Frame:**  2 weeks  **Areas of Test:**  Functionality of sending SMS message  Input validation  Authorization & usage constraints  Presentation & Layout  Platforms  Concurrency & Scalability  **Scope:**  Given limited time, scope will be reduced. All areas of test will be exercised, but priority will be given to major functionality and authorization and usage constraints. Since this is an initial release, presentation and scalability will be less important. A simple layout reduces the risk of presentation problems. A limited release will alleviate scalability issues until further operational testing can be performed.  **Test Cases:**  **Functionality**  User can send SMS  Recipient receives SMS  User can specify a product in the SMS  Link to product specified takes recipient to the correct product  Link to product tracks that it was sent by SMS from user  **Input Validation**  Input must be a string  Input cannot be more than 140 characters  Input can contain a single product link  Product link must be to an amazon product amzn.com/  No other links are allowed  **Authorization & Usage Constraints**  A user must be logged in to Amazon to use the SMS service  A user can only send 1 SMS per minute  A user can send a maximum of 5 SMS per day  A user should only be able to send SMS from 1 IP address within 1 minute  **Presentation & Layout**  Presentation should match attached screenshots  Test Major browsers (FF 3.6, IE 7, Safari) and devices (Windows, Mac, iPhone, Android)  Layout should be the same on all platforms  Mobile device layout should fit on screen  **Concurrency & Scalability**  System should be able to send up to 10000 messages per minute  A message should send or fail within 30 seconds  Every message that has not sent or failed within 1 minute will be cleared  **Test Strategy**  All testing will be performed manually for the initial release except concurrency and scalability tests. A utility script (using JMeter) will be provided that spawns 1000 threads on 10 computers and sends 10000 messages simultaneously.  There will be bug fix releases at the end of each week. A limited regression test will be run after each release that takes no more than 2 hours. |