Please find below the QA Engineer Prescreening Assessment as per our conversation.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

 Note: Following is the list of preferred tools set but feel free to pick anything else that you are more comfortable with.

- PHP, Ruby

- Selenium 2

- appium

* Since I don’t have much experience with PHP or Ruby I will be using JAVA for programming language, Selenium+JAVA+PageObjectModel for Web automation, Appium+JAVA for Mobile App Automation and Maven Tool as a dependency managment tool inorder to provide answers below.

Q1: Create a class to perform simple arithmetic and write unit tests for it.

🡪 Qns1.rar .Here it contains Arithmetic.java class with a single function to add 2 positive integers. All the tests are in ArithmeticTest.java.

Q2: Why is test code coverage a good metric and what are its shortcomings?

To elaborate use the class and tests created in Q1.

Ans: Test code covagage is a good metric because it measure the degree or amount of code execution during the Test run. But having said that, there are some shortcommings. Let me explain:

* In case of code coverage for Qno 1, I kept 1 function to make it more clear. Only 1 function is in place that is supposed to add 2 positive integer numbers. In case of code coverage, it looks as if only one test seems fine. That is, test called positiveNumber() seems sufficient enough to test the method because its passing 2 positive values (2 and 5) whose result should be 7. This testcase called positiveNumber() can execute addPositiveNumbers() method. But its not sufficient.
* Reason is because positive number could be any number starting from 1 to infinity in the positive side of number system. So how much testcases are required? In My view only 3. How ? Let me explain.
  + First one is add 2 positive single digit number. This test case is obvious. Everyone does it.
  + Second one try adding 2 single digit **negative numbers**. Its a precautionary/Negative test. In case if there occur a code change and somebody forgets to implement the negative input handling clause.
  + Third one is boundary value testing. This is tricky test case because we have to make sure that we exploit the limitation of term “positive integer” in java. For every integer value in java, its positive limit is 2147483647 and not more. If we try adding 2 number at its limit then the outcome must be 2\*2147483647 right? But no. Because of limitation of integer value a smarter way of validating the limits must also be consider.
  + People might think that there might be the need of adding another testcase that is less than 2147483647 value (lets say 2147483638 and 10). It could generate 2147483648 right? But its really not necessary because it will now generate garbage negative value if value exceeds 2147483647.

Q3: How do you think testing in agile context is different then testing in traditional software development processes?

Ans: From my experience, Its different because in Agile I get to test early as compared to traditional where I usually get to test towards the end of the release. Team members are busy discussing about how to test rather than how to devlop in the begining of the iteration. Development seems more test driven and As a QA, I am more to automate things rather that manually execute. However, its not feasible to automate in some sceaniors.

One drawback I find is that executing regression suit in Agile is abit diffcult. We do some regression towards the end of the iteration but more on the feature level and less on the product level. In traditioal sdlc process, we always save some time to do regression testing in both feature level aas well as product level (end to end detail testing of the software).

Q4: Open [propertyfinder.ae](http://propertyfinder.ae/) in your browser and take a look around Now open [propertyfinder.qa](http://procpertyfinder.qa/) and do the same.

Now consider the following scenario and automate it using Selenium or similar tool.

"As a user I want the list of the least expensive apartment to rent in marina which has at least two bedrooms"

-- Additionally open the last item in the result set and verify it has two bedrooms

-- Bonus points: Refactor the above script to be data driven?

Ans: PropertySearch.rar

Here I have followed PageObjectModel . It contains a test file called PropertyFinderSearchTest.java file which contains 3 tests.

* First test is just finding the list of the least expensive apartment to rent in marina which has at least two bedrooms.
* Second test does the same as first but now it opens the last Page>>last item and verify that it has 2 bedrooms.
* Third test makes it data driven. I have used a properties file called dataDriven.properties with comma separated parameters.

 Q5: Which is your favorite open source Android/iOS apps? Automate two of its functional tests.

Ans: I love Playing games and I am pretty sure they are not Open Source. I recently started Appium so I have choosen GMAIL app for the Automation.

AppiumGmail.rar is the file, Here I have 2 automated tests. One is to switch account and switch it back, and another is to send an email.