Aptitude test for testers

Version 1.1

Answers

Introduction

The tester's aptitude test has been compiled to assist the test manager/team leader in the recruiting of good quality testers. This test should be used in conjunction with other interviewing techniques.

Structure

The test comprises of 25 questions, each carrying different marks. The questions have been designed to test a broad knowledge of testing from scenario testing to specific questions on testing tools.

Marking

The test should be completed in 1.5hours. However if it takes the candidates longer then penalty points can be deducted (1 point for every extra minute for example). If the candidates take less time then they can be awarded extra points.

The total number of points given from the test is 150 – this can be translated into a percentage and you might want to consider having a sliding scale for the potential testers:

- Score less than 50% - **Fail**
- Score 50% to 65% - **Trainee Tester**
- Score 65% to 80% - **Tester**
- Score more than 80% - **Senior Tester**

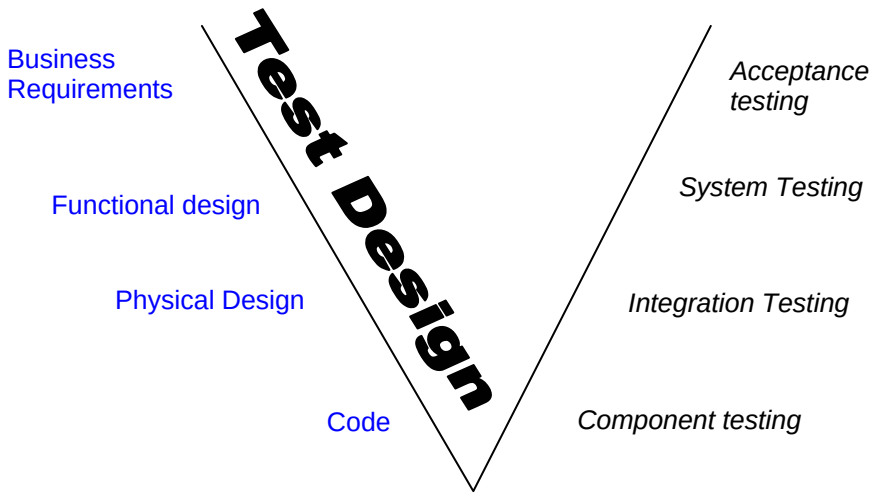
Acknowledgements

The first draft of the Aptitude Test was issued in June 2001. The following people were kind enough to suggest alternative questions and comment on the content.

Appreciation for the first version goes to:
Dave Norman, Tim Edmonds, Richard Durham, Barbara Eastman, Kris Pears, Steve Walker and Christine Hunt.

No.	Answer	Score
1.	They are both accurate! The purpose of testing is to find faults AND ensure it meets the users needs (fit for purpose).	**2**
2.	It depends on how good your tests were and what they were testing. To have justified confidence in the software we must have confidence in our tests, data and environment.	**2**
3.	Talk to users, developers and analysts to understand what the system is supposed to do. Document this understanding and get it reviewed and use this as a substitute for the Requirements/Design documentation. Talk with testers who have tested the system previously Read whatever is available and clarify assumptions	**4**
4.	The tester should first establish whether the reason is because of a test fault (i.e. they have made a mistake) or whether it is an environment fault. If neither of these are true then they should then check to see whether this fault has already been raised. If not then either raise the fault or more preferable – talk to the development group to check the fault out.	**3**
5.	They are as important as each other. However testers need to have a different mindset to developers and therefore should actively look for potential faults. If we only concentrate on positive tests (show that the system does what it should do) then we will potentially experience problems when the system goes live. If we only concentrate on negative tests (showing the system doesn't do what it shouldn't) then again we could potentially miss significant faults. However if we look primarily at breaking the system then we may find lots of faults (the what if scenarios) but we may not establish if the system is going to meet the users needs and requirements. A balance is needed with all three approaches.	**2**
6.	A good test is one that can potentially find a fault in the system. If this test does not find a fault then it will give us a certain amount of confidence. Tests must also be efficient – we should not have tests which all do the same thing.	**2**

No.	Answer	Score
7.	Do you have a test case: 1. for a valid scalene triangle? 2. for a valid equilateral triangle? 3. for a valid isosceles triangle? 4. for each of the three permutations of two equal sides in valid isosceles triangles? 5. in which one side has a length of zero? 6. in which one side has a negative length? 7. in which the sum of the length of two sides is equal to the length of the third? 8. for each of the three permutations of case 7? 9. in which the sum of the length of two sides is less than the length of the third? 10. for each of the three permutations of case 9? 11. in which all side lengths are zero? 12. which uses non-integer input values? 13. which uses the wrong number of input values? 14. did all your test cases specify the expected output? Myers states that experienced professional programmers score on average 7.8 out of the first 14 questions. Extra points can be given for further tests such as performance, reliability and configuration	**17**
8.	This is not a serious problem. The message is being printed. The best solution would be (a) or (d) – it is essential that faults be raised as soon as possible so that Development can fix them. However this is dependent on the severity and priority of the fault. This fault is not stopping any further testing on this script – it might be that other similar problems occur with other messages and this extra information might assist development with further investigation	**3**
9.	The answer is (d) – it might be our environment or it could have been fixed by some other fault fix in the new version.	**3**
10.	First we should investigate the faults – is it because we had run our tests wrongly, or that we were running the tests on the wrong environment? Assuming that it is because the software has regressed – then we must establish the nature of the faults and severity of the faults. It is probably inefficient to run any further tests at this stage. We should work with development in getting a new version of the software with the faults fixed and re-tested before running test set 2.	**6**

No.	Answer	Score
11.	 The key aspect here is that testing should happen throughout the Development Lifecycle. Also designing of the test cases should happen as soon as possible.	**6**
12.	Component Testing Lowest level of testing, detail, finding faults, performed by the developers Component Integration Combining components, testing interfaces, performed by developers, various types of integration (top-down, functional, bottom up and big bang). Business scenarios and non-functional aspects if possible. System Testing (functional and non-function) Testing the system as a whole. Testing requirements and business processes. Also testing non-functional aspects such as Performance, usability etc. System Integration Testing the system with other systems and networks Acceptance Testing Testing by users/customers to gain confidence that the system is going to support the business as well as meet their requirements.	**10**
13.	Regression Testing: Running tests to ensure that the software has not regressed in anyway as a result of changes to the software and/or environment. Regression testing is running passed tests again to ensure that they still pass. Re-Testing This is running a test again that had found a fault to check that the fault has been fixed correctly. Re-testing is running a failed test again to ensure that it now passes.	**4**

No.	Answer	Score
14.	Assuming there are 7hours per working day. This task would take you: 600x10 = 6000 minutes = 100 hours = 14.286 days There are a number of options that could be considered: ❑ Work overtime (this should not be considered as a first resort) ❑ Ask for more staff to help (again this may not be the best approach, particularly if you need to spend time training and mentoring the new staff) WE SHOULD: ❑ Re-prioritise our tests and run the most important tests first ❑ Assuming that not all the 600 tests would have been run within this time, risk assessment need to be made as to the consequences of not running the extra tests. ❑ After this initial week and the system is implemented there is no reason why the extra tests could not be run (assuming that you are given the time)	**5**
15.	Testing tools are very important to assist the tester in their work. Using tools can also potentially make the tester more efficient in their work – they are able to run more tests (using regression testing for example). Or they can quickly compare 3 reports (comparison tool). The tools in themselves however do not make good testers and also should not be considered if the test process is in 'chaos'.	**4**
16.	❑ Requirement Testing Tools ❑ Test Design Tools ❑ Test Data Preparation Tools ❑ Regression Testing tools ❑ Debug Tools ❑ Dynamic Analysis Tools ❑ Coverage Measurement Tools ❑ Static Analysis Tools ❑ Performance Testing Tools ❑ Test Management Tools ❑ Network monitoring tools ❑ Test Harness or Simulation tools The importance of this question is to see if the candidate has any knowledge about tools. We do not want the names of tools but want to know if the candidate can distinguish between the types of tool.	**9**
17.	Any of the following: BS 7925-1 (Glossary of testing terms), BS7925-2 (Component Testing), ISO9000 and ISO9001 (Quality standards), IEEE829 (Test Documentation), IEEE1028 (Reviews), IEEE1044 (Incidents)	**2**

No.	Answer	Score
18.	**How would you approach these requirements:** a) The system must be user-friendly What do we mean by 'user-friendly'? Questions to ask: ❑ **Friendly to whom?** ❑ **Who are the users?** Test approaches: ❑ **Talk to the users** ❑ **Document assumptions** ❑ **Compile test scenarios for people who have not seen the system** ❑ **Document tests and review these with the users** b) The system must be easy to install What do we mean by 'easy? Questions to ask: ❑ **For whom?** ❑ **Is there any installation documentation to follow?** Test approaches: ❑ **Follow installation documentation (if there is any)** ❑ **Allow tests to be run by an inexperienced user to see how easy it is** ❑ **Document tests and review these with the users** c) The following response times are to be achieved with the new system: • Initial loading of the web application must be achieved within 3 seconds • Updating of the information on the web page must be no more than 5 seconds Once more we need to ask some probing questions surrounding this requirement: ❑ **What happens if we don't meet the times?** ❑ **Would a range of values be better?** ❑ **What is happening on the network?** ❑ **Are these average times or are they 'peak' times?** ❑ **What is involved in updating – how much information?** **In attempting to test this requirement we would document the exact criteria for the test and the simplest way would be to time a number of tests and supply the average.** **With all these 3 requirements, what we are looking for is to see whether the potential tester will challenge the requirements of whether they would just accept them and try to test to the best of their ability**	**9**
19.	❑ There are faults in the software ❑ Failures in live operation can be expensive ❑ Sometime a 'legal' or contractual requirement ❑ To asses the quality of the software ❑ To preserve the quality of the software ❑ To help achieve quality software (by finding and removing the faults)	**4**

No.	Answer	Score
20.	Positive/Valid Tests Input — Expected Result 0 — Operator 201 — Room 201 (valid boundary) 405 — Room 405 (valid partition) 500 — Room 500 (valid boundary) 7 — Room Service 8 — Reception 9 — Outside line Negative/In-valid Tests Input — Expected Result 1 — Error 6 — Error 200 — Error (invalid boundary) 501 — Error (invalid boundary) 550 — Error (invalid partition) Any other button — Error Destructive Tests What if I accidentally hit multiple buttons? What if I entered an 0800 number without first getting an outside line (operator should answer)	**15**
21.	Static Testing is non-execution of the code. Techniques include; reviews, inspections, walkthroughs, individual techniques such as desk checking, data-stepping and proofreading. There is also static analysis (data flow and control flow analysis)	**4**

No.	Answer	Score
22.	• ask the customer to prioritise the requirements • ask the customer to prioritise the tests • what is most critical to the customer's business • test where a failure would be most severe • test where failures would be most visible • test where failures are most likely • areas changed most often • areas with most problems in the past • most complex areas, or technically critical	**10**
23.	Key points: 1. Different programmers wrote A and B 2. Complexity level of the programs are the same 3. Size of the programs are the same 4. Tester is the same for testing A and B 5. Number of tests run on both programs is the same 6. Number of bugs is higher in program B Program B seems to have far more faults therefore we would be inclined to spend the further week testing Program B, as there is likely to be more bugs to find. We may also not be very confident at this point with Program B therefore we need to see our confidence increased.	**4**
24.	1. Invalid Card – reject card and exit 2. Valid Card and Invalid PIN – error message 'invalid pin…' (then enter valid pin) 3. Valid Card and Invalid PIN – error message 'invalid pin…' (then enter another 2 invalid Pins) 4. Valid Card, Valid Pin & Cancel (correct length pin) 5. Valid Card, Valid Pin in a large number – but the pin number contains more than the maximum number – should error 6. Valid Card, Valid Pin & Cash Withdraw without receipt 7. Valid Card, Valid Pin & Cash Withdraw with receipt 8. Valid Card, Valid Pin & Balance enquiry 9. Valid Card, Valid Pin & Statement Request 10. Destructive tests include: • Putting in 2 cards • Putting correct pin, but adding an extra number to make invalid Assumptions: 1. Can insert up to 3 invalid pins and machine retains card 2. Can only select one transaction and then have to re-insert card 3. Pressing cancel will return card	**10**

No.	Answer	Score
25.	Potential Problems/Omissions ❑ No date on log as to when raised ❑ No keywords (i.e. screen) so that searches can be performed preventing duplication of fault logs ❑ No status of the log (opened/fixed/closed/cleared etc.) ❑ No owner of the log. ❑ Has priority – but no severity (i.e. risk to the customer) ❑ No version number of the system being tested – it is very likely that the testers are on a different version to development and that it was a fault but has been inadvertently fixed on this latest software ❑ Query the priority of this log (should it be a 3?) ❑ No actual error message on the log – this may give some clue to the developer about the nature of the fault ❑ Response seems to be leading to a dialogue – if we are not careful this fault will never be fixed! Tester should talk to the developer rather than sending another message via the fault log. ❑ The response by the developer points to another part of the system (security) – this may be an indication of developers trying to quickly close the issue without performing sufficient investigation. It could however be because the tester has not spent enough time documenting the problem.	**10**