Software Testing — 1DL610

Justin Pearson

January 2018

Cover Sheet

Problem no.	Solution provided	Max	Your points
1		10	
2		9	
3		12	
4		10	
5		8	
6		9	
	Total:	58	

Anonymous Exam Code . : . . . .

Exam Rubric

All answers to be written in English or Swedish. A mark of 50% is required for a 3, the marks 4 and 5 are evenly distributed in the interval from 50% to 100%. I will not be able to come to the exam to answer questions, if you have trouble understanding a question then please state clearly any assumptions that you have to make to answer the question.

Hjälpmedel:

Pen,pencil, ruler,rubber, dictionary, calculator.

1. General Questions

   (a) What is unit testing? Give an example, and describe what phase of the software development life cycle it is appropriate for. (**points 2**)

   (b) Describe the difference between black-box and white-box testing. (**points 2**)

   (c) Describe how testing strategies would differ in integration testing and unit testing. (**points 2**)

   (d) Dijkstra famously said: "You can test for the presence of bugs, not their absence". With reference to Turing's halting problem explain what Dijkstra meant. For the practical software engineer testing is still useful. Explain why you should still do testing even if you believe Dijkstra's pronouncement. (**points 4**)

2. The RIP model.

   (a) Describe the distinction between a *Software fault* and a *software failure*. Give examples to illustrate the distinction. (**points 2**).

   (b) Explain the RIP (Reachability, Infection and Propagation) model and its relation to software faults and failures. (**points 3**)

   (c) Imagine that you are testing a function, and you require a test case that makes a loop execute exactly 0 times.Further, the counter in the loop is not one of the parameters of the function. Explain how you would go about deriving a test case and how this relates to the RIP model. (**points 4**).

3. Consider the following fragment of C(like) code:

```
int foobar(int x) {
  int z = 0;
  while(z<15) {
    for(int i=0; i<x; i++) {
       if( i % 2 == 0) {
         z = z + i;
       }
        else {
          z = z + 3;
       }
    }
    }
    return(z);
}
```

Note that % the remainder after division operator.

(a) Draw a Structural Graph that represents the above piece of source code. You must indicate how each node and edge corresponds to the code.(**points 3**).

(b) For you graph, enumerate a complete set of test paths that guarantee edge coverage. (**points 2**).

(c) Construct a set of test cases that covers all feasible test paths that guarantee edge coverage. If the path is not feasible, then say so. (**points 2**)

(d) Enumerate all the prime paths of the graph you have constructed. You must show how your prime paths are constructed. (**points 2**)

(e) Construct a complete set of test paths that cover all the prime paths. (**points 2**)

(f) For each of the test paths above give a test case for the path. If the path is not feasible, then say so. (**points 2**)

4. Logic Coverage. Given that the variables `a,b` are integers, and that `P,Q` are boolean variables, consider the following expression:

```
(P || (a < 5) ) || (  (a < b) ||  ( b<5 && Q) )
```

Note that `&&` is logical conjunction (and) and `||` is logical disjunction (or).

(a) Define *clause coverage* (**points 2**) and for the above expression derive a set of test cases that give clause coverage. (**points 2**)

(b) Define what it means for a clause to determine a predicate. (**points 2**)

(c) There are many variations on how to define *active clause coverage*. Give two definitions and explain the differences. (**points 4**).

5. Input Space Partitioning and modelling.

   (a) For an input domain define what a partition of a domain is. (**points 2**)

   (b) When deriving test cases, why should the blocks in a partition be disjoint? (**points 2**)

   (c) Consider a function

   `MoreThanOnce(L,Elem)`

   Where `L` a list or an array and `Elem` is some object. The function returns `true` if the object `Elem` appears two or more times in the list `L`.

      i. First give some partitions of the input parameters using an interface based project. (**points 2**)

      ii. Now Give some partitions of the input parameters using a function based approach. (**points 2**)

6. General Practical Questions on Testing.

   (a) What is a test plan? Why is one necessary or useful? What sort of things go into a test plan? (**points 4**) (I don't expect you to give a complete description of test plans, but I do expect you to include some of the important items that should go into a test plan.)

   (b) It is important to keep a good set of tests cases for regression testing, but as the project ages the regression suite can become too large. As a test engineer explain what criteria you would use to choose which test cases remain and which test cases can be taken away. (**points 3**)

   (c) Even during requirements gathering there is an opportunity for a test engineer to contribute. Give some examples of how a test engineer could contribute. (**points 2**)