Försättsblad till skriftlig tentamen vid Linköpings Universitet

Datum för tentamen	2018-01-02
Sal	
Tid	8-12
Kurskod	TDDD04
Provkod	
Kursnamn/benämning	Programvarutestning
Institution	IDA
Antal uppgifter som ingår i tentamen	
Antal sidor på tentamen (inkl. försättsbladet)	6
Jour/Kursansvarig	Kristian Sandahl
Telefon under skrivtid	0706 68 19 57
Besöker salen ca kl.	10:30
Kursadministratör (namn + tfnnr + mailadress)	Anna Grabska Eklund
Tillåtna hjälpmedel	Ordbok

LiTH, Linköpings tekniska högskola
IDA, Institutionen för datavetenskap
Lena Buffoni

Written exam

TDDD04 Software Testing

2017-01-02

Permissible aids
Dictionary (printed, NOT electronic)

Teacher on duty
Kristian Sandahl, tel: +46 (0)706 68 19 5

Instructions and grading
You may answer in Swedish or English.

Your grade will depend on the total points you score on the exam. This is the grading scale:

Grade	3	4	5
Points required	50%	67%	83%

Important information: how your answers are assessed

Many questions indicate how your answers will be assessed. This is to provide some guidance on how to answer each question. Regardless of this it is important that you answer each question completely and correctly.

Several questions ask you to define test cases. In some cases you are asked to provide a minimal set of test cases. This means that you can't remove a single test case from the ones you list and still meet the requirements of the question. Points will be deducted if your set of test cases is not minimal. (Note that "minimal" is not the same as "smallest number"; even when it would be possible to satisfy requirements with a single test case, a set of two or three could still be minimal.)

You may find it necessary to make assumptions in order to solve some problems. In fact, your ability to recognize and adequately handle situations where assumptions are necessary (e.g. requirements are incomplete or unclear) will be assessed as part of the exam. If you make assumptions, ensure that you satisfy the following requirements:

- You have documented your assumptions clearly.
- You have explained (briefly) why it was necessary to make the assumption.

Whenever you make an assumption, stay as true to the original problem as possible.

You don't need to be verbose to get full points. A compact answer that hits all the important points is just as good – or better – than one that is long and wordy. Compact answers also happen to be quicker to write (and grade) than long ones.

Please double-check that you answer the entire question. In particular, if you don't give a justification or example when asked for one, a significant number of points will always be deducted.

1. **Definitions** (6p)
    a. W hat is **exhaustive black-box testing**? Should a project aim for exhaustive testing? Justify. (2p)
    b. Define use-case testing and give two elements of a use-case. (2p)
    c. What are MM-paths? At what level can this type of testing be applied. (2p)
2. **Control flow coverage** (10p)

```
a.  public int maxPositive(int a, int b, int c){
b.
c.                  if (a < b){
d.                          if (b > c & b>0)
e.                                  return b;
f.                          else if (c > 0)
g.                                  return c;
h.                          else
i.                                  return 0;
j.                  }
k.                  else{
l.                          if (a > c & a>0)
m.                                  return a;
n.                          else if (c > 0)
o.                                  return c;
p.                          else
q.                                  return 0;
r.                  }
s.  }
```

    a. For the code above draw a control flow graph (3p)
    b. Give a set of test-cases that provides decision coverage (2p)
    c. Give a set of test cases that provides modified condition/decision coverage (2p)
    d. Give two possible test criteria for loop coverage, are they always applicable? Justify. (3p)
3. **Easy points** (6p)

Answer TRUE or FALSE. You get 1p for each correct answer, 0p for no answer, and -1p for each incorrect answer. However, you cannot get negative points for this question.

    a. Testing cannot prove correctness
    b. Coverage is a good indicator of test suite effectiveness
    c. Domain-based testing is best suited to numerical values
    d. A fault in a program may lead to an error
    e. Black-box testing cannot be applied at system level
    f. Exploratory testing is dependent on the quality of the requirements
4. **Decision table** (10p)

You are asked to test the following functionality: A car insurance premium is calculated on the base price using the following rules:

- If a driver has more than 5 years of experience and no accidents he gets a discount of 10%
- If a driver has more than 10 years of experience and no accidents he gets a discount of 30%
- If the driver drives a red car he pays 10% more, even if he gets a discount
- If the driver has had an accident in the past 3 years he pays 30% more

    a. Draw a decision table for the problem above. If you make any assumptions, state them clearly. (6p)

b. Generate a set of test cases based on this decision table (4p)

5. **Path testing** (14p)

```
a.  /* returns the result of multiplying a by b */
b.      public int multiply(int a, int b){
c.          if (b==0)
d.              return 0;
e.
f.          if (b==1)
g.              return a-1;
h.
i.          b=b-1;
j.          return a * multiply(a, b);
k.
l.      }
```

   a. For the code above calculate the cyclomatic complexity and provide the set of basis paths. (6p)
   b. Based on these basis paths, provide a set of test-cases, trying to choose values meaningfully. (2p)
   c. Is the set of basis paths unique? Justify your answer. (2p)
   d. Describe how mutation testing works on the code below, by giving an example of two **different** mutations and say whether they are detected by your test suite or not. (4p)

6. **Test plan** (6p)
   a. Explain what a software defect taxonomy is and give **two** ways it can help with the testing process. (3p)
   b. Give three advantages of scripted testing over exploratory testing. (3p)

7. **System Testing** (10p)
   a. Give two advantages of graph-based integration over top down and bottom up integration. (2p)
   b. What are the criteria of a good model for testing? Explain (4p)
   c. Give two benefits and two limitations of model-based testing (4p)

8. Choice of test methods (6p)

For the following situations, describe the testing method you would choose and support your choice with concrete details from the test case. Only concrete explanations will get points.

   a. You need to develop a web application for online printing that will be deployed on all the computers at a company with a several operating systems and a large number of browsers.
   b. You need to develop a booking system for concert tickets. The system needs to have features such as booking a ticket, changing booking details, paying for a booking, queuing for booked tickets.
   c. You are asked to develop an application for a hospital triage system. It takes in a number of factors such as age, temperature, pain level and assigns a level of priority to the patient on a scale from one to five.

9. Coverage criteria (6p)

   a. Explain what data-based coverage criteria mean. Give an example of a testing method for achieving data coverage. (2p)

b. As a project manager, you need to decide when to stop testing. Describe two possible criteria you could use to decide to stop testing. For each of them give one limitation (4p)