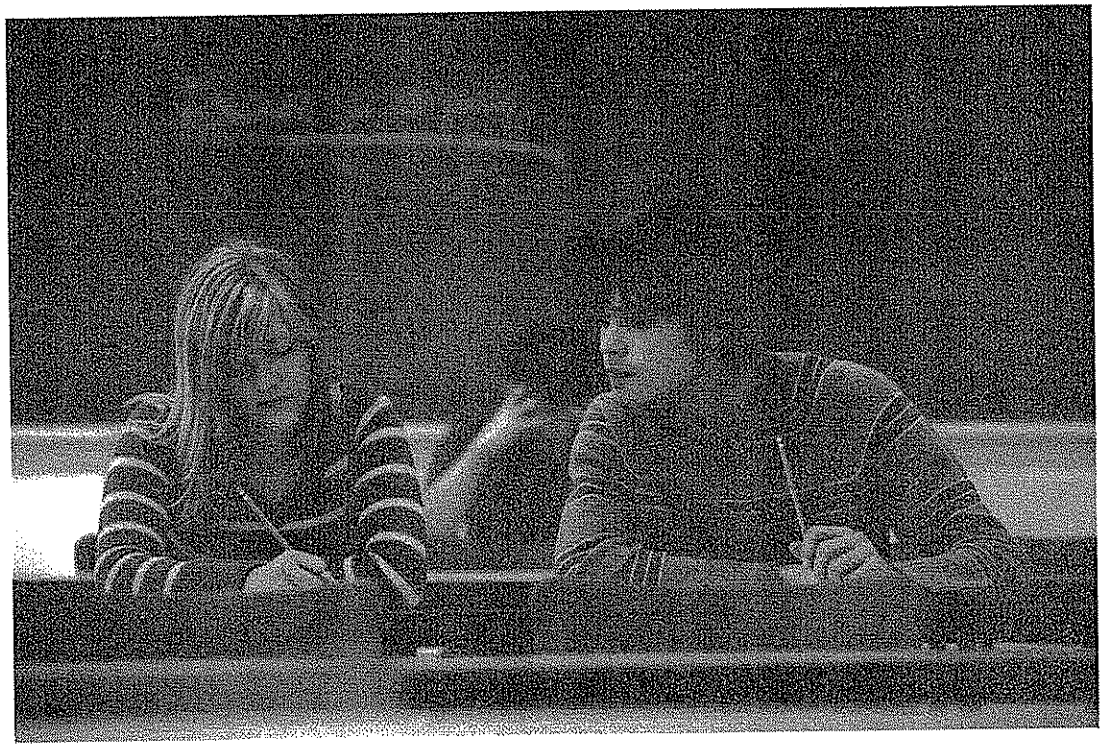
# Försättsblad till skriftlig tentamen vid Linköpings Universitet

(fylls i av ansvarig)

| | |
|---|---|
| **Datum för tentamen** | *2012-08-15* |
| **Sal** | ***TER 2*** |
| **Tid** | *8-12* |
| **Kurskod** | *TDDD04* |
| **Provkod** | |
| **Kursnamn/benämning** | *Programvarutestning (Software Testing)* |
| **Institution** | *IDA* |
| **Antal uppgifter som ingår i tentamen** | *12* |
| **Antal sidor på tentamen (inkl. försättsbladet)** | *9* |
| **Jour/Kursansvarig** | *Peter Bunus: peter.bunus@liu.se* |
| **Telefon under skrivtid** | *0703-496758* |
| **Besöker salen ca kl.** | *kl 10:00* |
| **Kursadministratör (namn + tfnnr + mailadress)** | *Gunilla Mellheden*<br>*013 282297 gunilla.mellheden@liu.se* |
| **Tillåtna hjälpmedel** | *Inga, endast skrivmaterial*<br>*No aids are allowed* |
| **Övrigt (exempel när resultat kan ses på webben, betygsgränser, visning, övriga salar tentan går i m.m.)** | |
| **Vilken typ av papper ska användas, rutigt eller linjerat** | |
| **Antal exemplar i påsen** | |

| | |
|---|---|
| Report No | TENTA_TDDD04 |
| Organization | Linköping University, Sweden |
| Last Modified | 11 August 2012 |
| Modified by | Peter Bunus (peter.bunus@liu.se) |

# Tentamen TDDD04 (Programvarutestning)
# Examination TDDD04 (Software Testing)



Linköping University, Department of Computer and Information Science
SE 581-83 Linköping Sweden

pelab

# Tentamen: TDDD04 – Programvarutestning

## Examinator: Peter Bunus

## Information

Poängavdrag kommer att göras om punkterna nedan inte åtföljs!

1) Använd endast framsidan (delfrågor kan vara på samma sida).
2) Sortera inlämnade svar med avseende på uppgiftsnummer i stigande ordning.
3) Svaren får vara på svenska eller engelska.
4) Dina svar skall tydligt visa lösningsmetod. Enbart rätt svar kommer inte att ge poäng. I det fall du är osäker på frågeställning, skriv ner din tolkning och lös uppgiften utifrån din tolkning.

## Betygsgränser

| | | |
|---|---|---|
| [0..55) | poäng | Betyg U |
| [55..70) | poäng | Betyg 3 |
| [70..85) | poäng | Betyg 4 |
| [85..100] | poäng | Betyg 5 |

Lycka till!

# Examination: TDDD04 – Software Testing

## Examiner: Peter Bunus

## Information

Please also observe the following; otherwise it might lead to subtraction of points:

1) Use only the front side of the sheets.
2) Sort the solution sheets according to the task number.
3) Answers may be in English or Swedish.
4) Your answers should clearly show solution methods, reasons, and arguments. Short answers should be briefly motivated. If you have to a make an assumption about a question, write down the assumptions you make.

## Grading

To pass the exam you have to do at least 55 points from 100 possible.

| | | |
|---|---|---|
| [0..55) | points | Grade Fx |
| [55..70) | points | Grade C |
| [70..85) | points | Grade B |
| [85..100] | points | Grade A |

Good Luck!

Bonne chance!

Viel Glück!

Sėkmės!

祝你好運

祝福

1. A program validates a numeric field as follows: values less than 10 are rejected, values between 10 and 21 are accepted, values greater than or equal to 22 are rejected. Which of the following input values cover all of the equivalence partitions? (10p)

   a) 10,11,21
   b) 3,20,21
   c) 3,10,22
   d) 10,21,22

   You should motivate your answer. An answer without a proper motivation will lead to point subtraction.

2. Define Boundary Value Testing. Give an example (10p)

3. Which of the following requirements is testable? Please briefly explain your answer (5p).

a) The system shall be user friendly.
b) The safety-critical parts of the system shall contain 0 faults.
c) The response time shall be less than one second for the specified design load.
d) The system shall be built to be portable.

4. Analyze the following highly simplified procedure:

   Ask: "What type of ticket do you require, single or return?"
   IF the customer wants 'return'
   Ask: "What rate, Standard or Cheap-day?"
   IF the customer replies 'Cheap-day'
   Say: "That will be €11:20"
   ELSE
   Say: "That will be €19:50"
   ENDIF
   ELSE
   Say: "That will be €9:75"
   ENDIF

   Now decide the minimum number of tests that are needed to ensure that all the questions have been asked, all combinations have occurred and all replies given. Please give a short explanation to your answer (10p).

   a) 3
   b) 4
   c) 5
   d) 6

5. In prioritizing what to test, the most important objective is to (5p):

a) find as many faults as possible.
b) test high risk areas.
c) obtain good test coverage.
d) test whatever is easiest to test.

You should motivate your answer.

6. Which of the following should NOT normally be an objective for a test? (5p)

a) To find faults in the software.
b) To assess whether the software is ready for release.
c) To demonstrate that the software doesn't work.
d) To prove that the software is correct.

7. An important benefit of code inspections is that they: (5p)

a) enable the code to be tested before the execution environment is ready.
b) can be performed by the person who wrote the code.
c) can be performed by inexperienced staff.
d) are cheap to perform.

8. Consider the following statements: (5p)

i. 100% statement coverage guarantees 100% branch coverage.
ii. 100% branch coverage guarantees 100% statement coverage.
iii. 100% branch coverage guarantees 100% decision coverage.
iv. 100% decision coverage guarantees 100% branch coverage.
v. 100% statement coverage guarantees 100% decision coverage.

a) ii is True; i, iii, iv & v are False
b) i & v are True; ii, iii & iv are False
c) ii & iii are True; i, iv & v are False
d) ii, iii & iv are True; i & v are False

9. Defects discovered by static analysis tools include: (5p)

i. Variables that are never used.
ii. Security vulnerabilities.
iii. Programming Standard Violations
iv. Uncalled functions and procedures

a) i , ii, iii, iv are correct
b) iii ,is correct i, ii, iv are incorrect.
c) i ,ii, iii and iv are incorrect
d) iv, ii is correct

10. Minimum Tests Required for Statement Coverage and Branch Coverage(10p)

```
Read P
Read Q
If p+q > 100 then
Print "Large"
End if
If p > 50 then
Print "pLarge"
End if
```

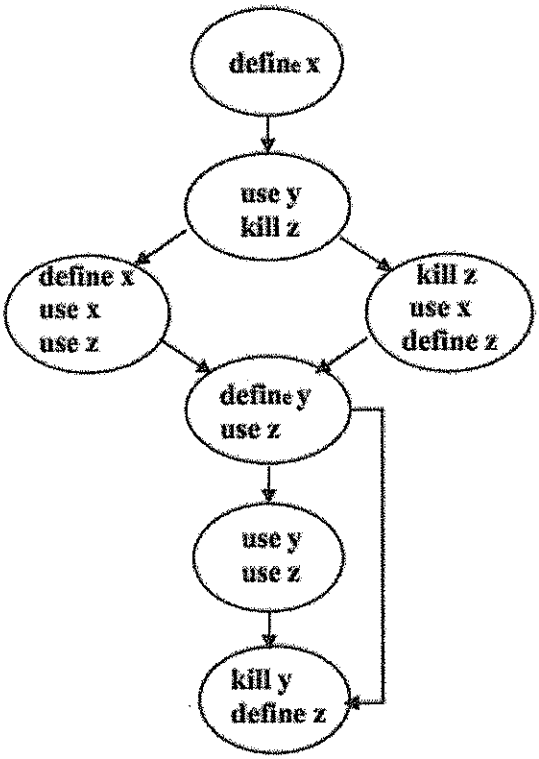
a) Statement coverage is 2, Branch Coverage is 2
b) Statement coverage is 3 and branch coverage is 2
c) Statement coverage is 1 and branch coverage is 2
d) Statement Coverage is 4 and Branch coverage is 2

Don't forget to briefly motivate your answer.

---

11. Early data flow analysis often is centered on a set of faults that are known as define/reference anomalies. Given the following notations and the control flow graph annotated with define-use-kill information, for each variable examine the define-use-kill patterns along the control flow graph and the kind of anomaly it could generate. (15 p)

- **d**: defined, created, initialized, etc.
- **k**: killed, undefined, released
- **u**: used for something
- **~d**: the variable does not exist, then it is defined
- **~u**: the variable does not exist, then it is used
- **~k**: the variable does not exist, then it is killed



12. Let us consider the following code:

```
PROCEDURE test( VAR b : INTEGER; c : INTEGER );
BEGIN
        b := c + 5;
END

BEGIN /* main routine of a nonsense program */
        x := 1;
        WHILE (x = 1) DO
                x := 2;
                test ( x, 1 );
                x := 3;
        OD;
        WHILE (x = 1) DO
                x := 4;
                x := 5;
                test ( x, 2 );
        OD;
        WHILE (x = 1) DO
                x := 6;
                IF (x = 7) THEN x := 8;
                ELSE test ( x, 3 );
                FI;
        OD;
END.
```

a) Draw a control flow graph (5p) to represent the following code:
b) Calculate the cyclomatic complexity of the control graph (5p)
c) Write down input values for test cases that satisfy McCabe's base path coverage (5p)