



Department of Computer Sciences

## **Examination paper for TDT 4242 Advanced Software Engineering**

**Academic contact during examination:** Jingyue Li

**Phone:** 9189 7446

**Examination date:** May 26. 2017

**Examination time:** 09:00 AM to 1:00 PM

**Permitted examination support material:** Code C – Pocket calculators allowed

**Other information:** Exam developed by Jingyue Li and checked by John Krogstie

**Language:** English

**Number of pages:** 7 (including this page and the Appendix 1)

**Checked by:**

---

Date

Signature

## Introduction

In this exam, you can score a maximum of 50 points. The remaining 50 points for the semester comes from the compulsory exercises.

If you feel that any of the problems require information that you do not find in the text, then you should

- Document the necessary assumptions
- Explain why you need them

Your answers should be brief and to the point.

## Task 1 – Requirements engineering (14 points)

The case is a snow clearing robot. See Appendix 1 for more details

### Task 1.1 Requirement quality (6 points)

Try to categorize the textual requirements in Appendix 1 according to the requirements quality metrics: *ambiguity*, *inconsistency*, *forward referencing*, *opacity*, and *noise*. (4 points). If one requirement has several quality issues, list and explain all of them.

Then, try to fix the requirements quality issues of each requirement in the snow clearing robot in Appendix 1 and write down the improved requirements. (2 points)

(Note: There is no single correct answer of improved requirements of the snow clearing robot. You can just formulate improved requirements based on what you think a snow clearing robot should be. The key point here is to show you know how to create high quality requirements).

### Task 1.2 GORE (4 points)

Based on your improved requirements, use the informal temporal patterns below to describe two upper levels of goal decomposition for the snow clearing robot case in Appendix 1.

Achieve [TargetCondition]

Cease[TargetCondition]

Maintain[GoodCondition]

Avoid[BadCondition]

Improve[TargetCondition]

Increase[TargetQuantity]

Reduce[TargetQuantity]

Maximise[ObjectiveFunction]

Minimise[ObjectiveFunction]

## Task 1.3 Requirement prioritization (4 points)

Prioritize the following high-level functional requirements of an **Intelligent Traffic Control System (ITCS)** using the cumulative voting method (2 points) and binary priority list method (2 points).

(Note: There is no single correct answer of ranking of the requirements. The key point is to show that you know how to use the two methods to prioritize requirements)

Req1. The ITCS system should be able to log and store traffic data collected through sensors into a server.

Req2: The ITCS system should be able to manage the traffic light to prioritize emergency vehicle.

Req3: The ITCS system should be able to manage the traffic light to decrease the amount of traffic congestions.

Req4: The ITCS system should be able to manage the traffic light to minimize the pollution produced by road traffic.

Req5: The ITCS system should be able to manage the traffic light to minimize pedestrians' waiting time when passing crosses.

Req6: The ITCS system should be able to analyze the traffic data and make a report.

## Task 2 – Testing (15 points)

### Task 2.1 Domain testing (3 points)

An online shopping web application has three variables (Availability, Payment method, and Delivery method). The possible values of the variables are as follows.

- Availability: Available (AVA), Not In Stock (NIS), DIScontinued (DIS)
- Payment method: Credit Card (CC), Gift Voucher (GV)
- Delivery method: Mail (MA), UPS (UPS), Fedex (FE)

Your task is to write all the 2-way combinatorial test cases (Abbreviations of each variable value can be used)

### Task 2.2 Regression testing (6 points)

Suppose we have a code snippet as follows. We initially have a test set T with 4 test cases to test the code. The 4 test cases in the test set T are:

t1:  $\langle x = 1, y = 2 \rangle$

t2 :  $\langle x = 1, y = 3 \rangle$

t3 :  $\langle x = 3, y = 1 \rangle$

t4:  $\langle x = 2, y = 2 \rangle$

If the line “return b\*b” in function f1 is changed to “return a\*b”, your task is to choose “safe re-testable subset” from T using graph-walk approach with CFG (Control Flow Graph) and explain in details how you choose the subset.

```

int M (int x, int y){
    int z;
    if (x>=y)
        z = f1 (x, y);
    else
        z = f2(x, y);
    return z;
}

int f1( int a, int b){
    if ((a-1) == b)
        return a-1;
    else
        return b*b;
}

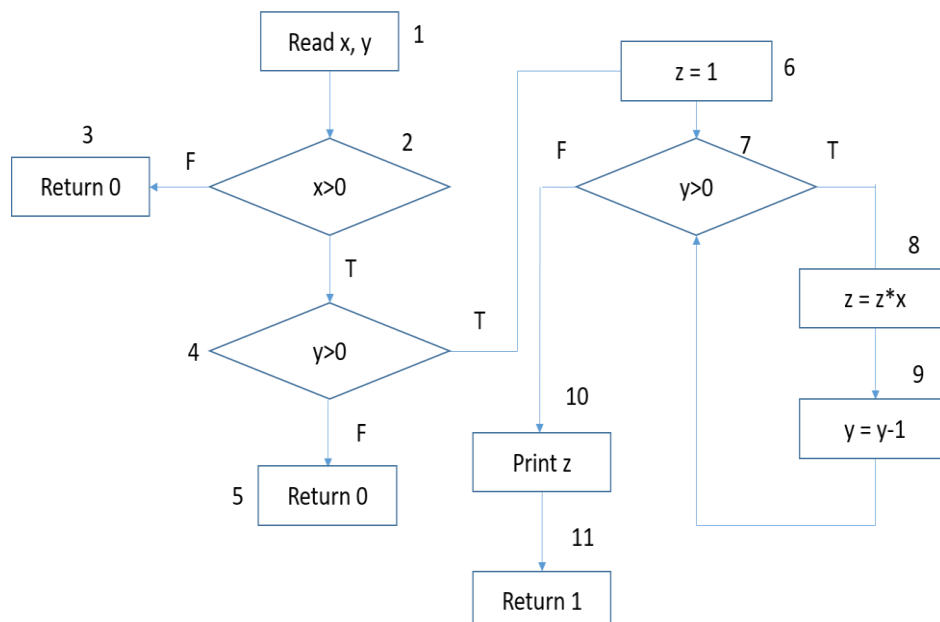
int f2( int a, int b){
    if (a == (b-2))
        return b-1;
    else
        return a-1;
}

```

**Code snippet**

## Task 2.3 Dataflow testing and test coverage (6 points)

For the following CFG (Control Flow Graph), find out paths to be covered for variables x, y, z, if All Uses (AU) is the test strategy to be applied.



## **Task 3 – Advanced topics (15 points)**

### **Task 3.1 (3 points)**

Explain the three high level categories of OSS license types and give one example license in each category.

### **Task 3.2 (3 points)**

Explain what “operating system-centric” software ecosystem is (1 point) and list its success factors (1 point) and challenges (1 point).

### **Task 3.3 (4 points)**

Explain SLA (Service level agreement) and its content (1 points), OLA (Operations level agreement) and its content (1 points), and the relationship between SLA and OLA (2 points).

### **Task 3.4 (5 points)**

Explain why systematic reading techniques outperforms unsystematic reading techniques (4 points), and list and explain one systematic reading technique (1 point).

## **Task 4 – Multiple choice (6 points, one point for each question. You get the one point if all correct answers are chosen)**

- 1) What are the desirable properties of the component-based architecture?
  - a) Parameterizable components
  - b) Components should not be customizable
  - c) Supports component development in multiple programming languages
  - d) Allows easy distribution of components from seller to buyer
- 2) Which are the challenges of service oriented software engineering?
  - a) Dynamic software evolution
  - b) Single point of failure
  - c) Complex negotiation process
  - d) Hard to understand when automated
- 3) What are benefits of TDD (Test-Driven Development)?
  - a) TDD is always applicable
  - b) TDD can lead to higher code coverage
  - c) TDD will guarantee a shorter development time
  - d) TDD helps simplify debugging

- 4) What are differences between Model-Based Testing (MBT) and traditional manual testing?
  - a) For MBT, test cases are tightly coupled to the model
  - b) For MBT, test cases are generated automatically from models
  - c) For MBT, there is a high cost in early phases of the project
  - d) For MBT, test oracles still need to be input manually
- 5) What can cause project cost/effort derives from its original cost/effort estimation?
  - a) Pressure from managers to give low bid to get the project
  - b) Over-optimistic estimates
  - c) Frequent major requirement changes
  - d) Major changes in design and implementation
- 6) Which are advantages of scenario-based testing?
  - a) Easy to reuse
  - b) It is a user focused testing
  - c) Does not require working features before testing
  - d) Can expose requirement related issues

## Appendix 1 – Snow clearing robot



(Note: This picture is just to illustrate what a snow clearing robot is. This robot is not necessary linked to the requirements below)

### Requirements:

Req1: The robot is switched on and off with a button on the instrument panel.

Req2: When the robot identifies that the snow on the ground around itself reaches a certain depth, the robot should automatically start and move to clear the snow.

Req3: When the user presses the remote-control key, the user can take over and have a full control of the robot to be able to move it forward, move it backward, make it turn left, and make it turn right. However, the robot can still make its own decision to move or not.

Req4: When the air temperature is below 0 degree Celsius, the robot should automatically go to the charge site to charge its battery.

Req5: The robot should be able to identify obstacles and avoid collision with the obstacles.

Req6: The robot should be able to turn to a safe mode when it is in an unsafe environment.

Req7: The robot should maintain a safe speed when clearing the snow.

Req8. The robot should be able to blink the light to another snow clearing robot to say hello when they meet.