NEWCASTLE UNIVERSITY

SEMESTER 1 2009/10

SOFTWARE VERIFICATION TECHNOLOGY

Time allowed – 1½ Hours

Instructions to candidates:

Answer ALL questions from Section A, ONE question from Section B and ONE question from Section C

Please attach "The model for annotation" sheet to your exam script booklet

Marks shown for subsections are indicative only

[Turn over

SECTION A.

Answer ALL questions in this section.

Question 1.

Explain the terms *total correctness* and *partial correctness* in the context of software verification [2 marks]

Question 2.

Explain the concepts *coupling* and *cohesion* in the context of software design [2 marks]

Question 3.

What is a *Kripke model* [2 marks]

Question 4.

Give definitions of *literal*, *clause-form* and *conjunctive normal form (CNF)*. [2 marks]

Question 5.

Define the term *Hoare triple* and the term *weakest pre-condition*. What is the difference between them? [2 marks]

SECTION B.

Answer ONE question from this section.

Question 6.

a) Describe two types of programming error (with examples) that may be revealed by *static analysis* of the code. [4 marks]

b) Explain how a Control Flow Graph (CFG) is used to model a piece of code. [2 marks]

c) Consider the code fragment below, in which `a, b, v, w, x` and `y` are positive integer variables. Line numbers are shown to the left for ease of reference.

```
 1.  read(a);
 2.  read(b);
 3.  if a > b then
 4.    read(x);
 5.    read(y);
 6.  endif
 7.  v := y;
 8.  if a-b > 0 then
 9.    w:=x;
10.  endif
```

Draw the CFG of the code above. [3 marks]

d) A static code analysis may be described as *Safe,* or *Precise.* What do these two terms mean, in terms of the results returned by the analysis? [3 marks]

e) Use the code sample and CFG from part c) to explain how a static analyser checks code. Explain why the process may not always produce a definite answer. [8 marks]

Question 7.

The implicit VDM function below is part of a model concerning the management of emergency calls for an Ambulance service.

The region covered by the service is divided into `Sectors`, each of which may be occupied by zero or more `Ambulances` at a given time. The mapping `AmbMap` represents this information. An Ambulance may or may not be available for callout. The mapping `Available` represents this information.

The function `getAmbulance` returns the identifier of an Ambulance that is available and present in the given Sector.

```
types
Sector = token;
AmbMap = map Sector to set of AmbulanceId;
Available = map AmbulanceId to bool;

functions
getAmbulance (loc: Sector, aMap: AmbMap) aId: AmbulanceId
pre loc in set dom aMap and
  exists a in set aMap(loc) & Available(a)
post aId in set aMap(loc) and Available(aId)
```

a) Using the `getAmbulance` function to illustrate your answer, describe the main features of the *design-by-contract* approach for software development. Include in your answer an explanation of the rights and obligations of the implementer and client of this function. [8 marks]

b) Consider a Java implementation of this model. Suggest concrete type representations or classes which might be used in place of these abstract types. [4 marks]

c) Suppose the Java implementor decided to implement the getAmbulance function so that it provided the identifier of the *nearest* available Ambulance in the given sector. Would such an implementation satisfy the contract? Explain your answer. [3 marks]

The JML specification (below) has been provided for a class which maintains information about books in a bookshop.

```
public class Book {
 private /*@ spec_public non_null @*/ String title;
 private /*@ spec_public @*/ int cost;

 //@ public invariant !title.equals("") && cost >= 0;

 //@ ensures cost == newprice;
 public void setCost(int newprice);

 //@ ensures \result == cost;
 public int getCost();

 //@ ensures t.equals(title) && c == cost;
 public Book(String t, int c);
}
```

d) Suggest any preconditions which should be added to the methods in this specification. [2 marks]

e) Consider a new method to be called *discount* which should record the new cost of the book to be reduced by 10 percent from its original price. Provide a JML specification (*ensures* and *requires* annotations, if required) for this method. [3 marks]

SECTION C.

Answer ONE question from this section.

Question 8.

Consider the following PROMELA model

```
#define top 3
#define floors 4
#define ground 0
chan opendoor[floors] = [0] of {byte};
chan closedoor[floors] = [0] of {byte};
chan opend[floors] = [0] of {byte};
chan closed[floors] = [0] of {byte};
chan call = [0] of {byte};
chan doorbutton = [0] of {byte};
byte floor=ground;
bool calls[floors];

proctype door(byte i)
{ byte any;
  do
  :: opend[i]?any -> {opendoor[i]!any; closedoor[i]!any; closed[i]?any}
  od
}

proctype lift()
{
  byte x;

  bool uptag=true;
  do
  :: call?x -> calls[x]=true
  :: calls[floor] -> {opendoor[floor]?x;
                      do
                      :: doorbutton?x -> calls[x]=true
                      :: true -> break
                      od;
                      closedoor[floor]?x;
                      calls[floor]=false
                      }
  :: !calls[floor] -> if
                      :: (floor!=top) && uptag -> floor++
                      :: (floor!=ground)&& !uptag -> floor--
                      :: (floor==top) -> uptag=false
                      :: (floor==ground) -> uptag=true
                      fi
   od
}

proctype user(byte f; byte t)
  {call!f;opend[f]!f; doorbutton!t; closed[f]!f}

init {
       run door(ground);
       run door(1);
       run door(2);
       run door(top);
       run lift();
       run user(ground,top);
```

```
  run user(1,top);
  run user(2, ground)
}
```

a) Explain how the `lift` process works. [3 marks]

b) Define the term deadlock in the context of a PROMELA model. [2 marks]

c) Annotate the model on the sheet provided to indicate how to verify that the model does not deadlock. [5 marks]

d) How would you use SPIN to prove that the model has the property that when the lift is called it will eventually arrive? [5 marks]

e) Describe an observer process, and appropriate means of verification with suitable changes to the model to prove that if a user selects a floor then that floor will eventually be reached. [5 marks]

Question 9.

a) Give an informal but precise description of the formula <>(p^[]q). [2 marks]

b) Produce the finite state automaton that accepts precisely the traces that satisfy <>(p^[]q). [5 marks]

c) Produce the negated form of <>(p^[]q) that could be easily translated into a finite state automaton showing how you transformed one formula into the other. [3 marks]

d) Produce the never claim that would be used to demonstrate that the negated formula of c) is true of the model. [6 marks]

e) Compare and contrast the advantages and disadvantages of theorem proving and model checking. [4 marks]