University of Swaziland

Department of Computer Science

Final Examination

May 2012

Title of paper : Software Engineering II

Course number : CS452

Time Allowed : Three(3) hours

Instructions :

- ***Each question is worth 25 marks.***
- ***Answer question 1.***
- ***Answer any three (3) questions from questions 2 to 6***

This paper may not be opened until permission has been granted by the invigilator

Question 1 – 25 marks

(Compulsory)

To give an exam, an instructor first notifies the students of the exam date and the material to be covered. She then prepares the exam paper (with sample solutions), gets it copied to produce enough for the class, and hands it out to the students on the designated time and location. The students write their answers to exam questions and hand in their papers to the instructor. The instructor then gives the exam papers to the Teaching Assistants (TAs), along with the sample solutions to each question, and gets them to mark it. The instructor then records all the marks and returns the papers to the students.

a) Using UML notation, draw a USE CASE diagram based on the description above. 5 *marks*

b) Draw a sequence diagram that represents this process. *20 marks*

Question 2 – 25 Marks

a) Briefly discuss the following fundamental principles of user-interface design

(i) user profiling *3 marks*

(ii) metaphor *3 marks*

(iii) feature exposure *3 marks*

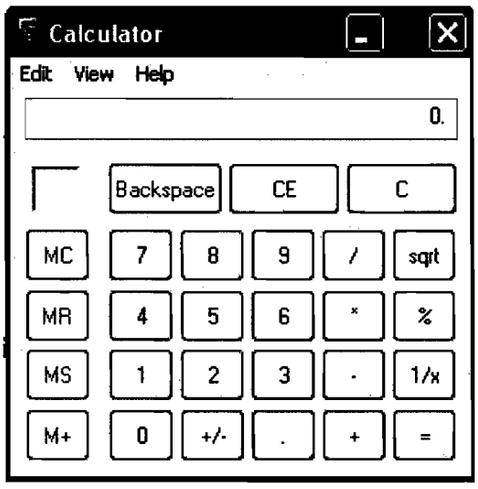
(iv) coherence *3 marks*

(v) help *3 marks*

b) Discuss the contributions of Object-Oriented Integrated Development Environments (IDEs) in improving the design of the user interfaces. 10 *marks*

Question 3 – 25 Marks

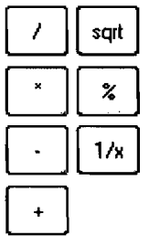
Consider the following calculator user interface.



(a) Using UML notation draw a general human-interaction component (HIC) class diagram for this user interface. *10 marks*

(b) Re-draw the HIC class diagram to indicate how the interface could be implemented in Delphi or Java. *5 marks*

(c) Using UML notation, draw a problem domain component (PDC) for the Calculator application showing a single class called **Integer** with member functions to implement all arithmetic operations shown on the interface. *5 marks*



(d) Re-draw the PDC class diagram to indicate how the class integer could be implemented in Delphi or java. Show connections between PDC and HIC components. *5 marks*

Question 4 – 25 Marks

(a) What is a test adequacy criterion? *2 marks*

(b) What is a test objective? Briefly describe the different categories of test objectives. *4 marks*

(c) Distinguish between the following terms

(i) Size-based and structure-based complexity metrics. *3 marks*

(ii) Procedural abstraction and control abstraction. *3 marks*

(iii) System testing and acceptance testing. *3 marks*

(iv) Fault-based testing and error-based testing *3 marks*

(v) Unit testing and integration testing *3 marks*

(vi) stepwise abstraction and stepwise refinement *4 marks*

Question 5 – 25 Marks

In relation to McCall's taxonomy of quality attributes:

(a) Distinguish between quality factors and quality criteria? How are these entities related? *6 marks*

(b) Define the quality factors: *integrity* and *interoperability.* *4 marks*

(c) Define the criterion of modularity, and explain why modularity is said to contribute to both portability and reusability. *5 marks*

(d) Select any 5 quality criteria and explain the steps you would take in order to qualify them in a software project. 10 *marks*

Question 6 - 25 Marks

(a) What is the difference between procedural abstraction and data abstraction? *3 marks*

(b) Explain the notion of cohesion and coupling. *3 marks*

(c) What is the essence of information hiding? *3 marks*

(d) Explain how coupling, cohesion and information hiding are interrelated design principles? *3 marks*

(e) Explain the notion of a software complexity metric. *3 marks*

(f) Give an outline of Halstead's software science. *4 marks*

(g) Consider the following insert sort routine.

```
procedure insert (a, b, n, x)
begin
bool found := true;
   for I := 1 to n do
     if a[i] = x
       then found := true; goto leave endif;
    Enddo;
leave:
     if found
     then b[i] := b[i]+ 1
      else n := n + 1; a[n] := x; b[n] := 1; endif;
end insert;
```

(i) Draw a control graph of the sort routine as described above. *4 marks*

(ii) Compute the cyclomatic complexity for the routine. *2 marks*