Term Test 1

COSC 4313 3.0 Software Engineering Testing

Section M, Winter 2006

Family Name: _____

Given Name(s): _____

Student Number: |___ |___ |___ |___ |___ |___ |___ |___ |___ |

Question	Out of	Mark
Q1	20	
Q2	20	
Q3	60	
Total	100	
Letter grade		

1. **[20 marks]** Following is a bug report for an Open Office bug. Describe four different follow-up tests that you would like to run while analyzing this bug.

   Summary: Deletion of a table row that contains merged cells causes double border
   Operating system: Windows XP
   Open Office version: 1.0.1
   Reproduction steps:

   (a) Start with a new text document in Open Office Writer
   (b) Select `Insert -> Table`. Enter 4 for number of rows and columns.
   (c) Select cells [1st column, 2nd row] and [1st column, 3rd row].
   (d) Select `Format -> Cell -> Merge` to merge these 2 cells.
   (e) Click to put the mouse cursor in cell [2nd column, 2nd row].
   (f) Select `Format -> Row -> Delete` to delete the 2nd row. The table now has 3 rows.

   You will now notice that between the 1st and 2nd row there is a double border.

   Any of the following follow-up tests would be worth 5 marks:

   - Delete rows without merging cells.
   - Delete the 3rd row rather than the 2nd one.
   - Select two other cells for merging.
   - Try the same experiment but with a bigger table (more than 4 rows and columns).
   - Merge more than two cells.
   - After the double border appears, keep merging more cells and deleting rows to determine if there is a cumulative effect.
   - Try to replicate in a different operating system.
   - Try to replicate with a different Open Office version.
   - Try with a different video resolution.

   Other correct answers are also possible.

2. **[20 marks]** Consider a logic function Z of four boolean variables A, B, C, D. The formula for Z is

$$Z = A{\sim}B{\sim}D + ABC + {\sim}ACD$$

Describe the process of applying the Variable Negation test strategy. What is the minimum test suite suggested by this strategy?

Following is the truth table (not necessary to draw for full marks).

Variant	A	B	C	D	Z
0	0	0	0	0	0
1	0	0	0	1	0
2	0	0	1	0	0
3	0	0	1	1	1
4	0	1	0	0	0
5	0	1	0	1	0
6	0	1	1	0	0
7	0	1	1	1	1
8	1	0	0	0	1
9	1	0	0	1	0
10	1	0	1	0	1
11	1	0	1	1	0
12	1	1	0	0	0
13	1	1	0	1	0
14	1	1	1	0	1
15	1	1	1	1	1

The following table presents all the candidate sets (an error in any of them was worth 2 marks).

Candidate set #	Term	Type	Set
1	A~B~D	Unique true point	{8,10}
2	A~B~D	Near false point ~A~B~D	{0,2}
3	A~B~D	Near false point AB~D	{12}
4	A~B~D	Near false point A~BD	{9,11}
5	ABC	Unique true point	{14,15}
6	ABC	Near false point ~ABC	{6}
7	ABC	Near false point A~BC	{11}
8	ABC	Near false point AB~C	{12,13}
9	~ACD	Unique true point	{3,7}
10	~ACD	Near false point ACD	{11}
11	~ACD	Near false point ~A~CD	{1,5}
12	~ACD	Near false point ~AC~D	{2,6}

Candidate sets of cardinality 1 determine that variants 6, 11, and 12 must be part of the test suite. This covers candidate sets 4, 8, and 12 as well. The rest of the candidate sets are disjoint, so a member from each set will be selected. The final test suite will contain the following test cases:

0 or 2
1 or 5
3 or 7
6
8 or 10
11
12
14 or 15

3. **[60 marks]** Following is the specification for a command-line software system that computes charges for telephone calls. Your task is to describe the application of any software testing strategies that you believe are appropriate in order to test this system. The derived test cases must be identified clearly in your answer. Answers that list test cases without describing how they were derived will receive a poor mark.

   The specification is provided as is. You may have to make assumptions about system behaviour that is not mentioned in the specification. If this is the case, choose reasonable ones and state them explicitly.

   Specification

   The system calculates the charge for a phone call on the basis of the following:

   - If the call duration is less or equal to 20 minutes, the charge is 5 cents per each minute.
   - If the call duration is more than 20 minutes, the charge is 1 dollar plus 10 cents per each minute in excess of 20 minutes.

   According to company policy, no phone call is allowed to last more than 6 days, 23 hours, and 59 minutes.

   The system accepts input from a command-line interface. There are 5 valid commands:

   (a) `start D HH:MM` This designates the day and time the phone call started.
   `D` is a number between 1 and 7 inclusive. 1 corresponds to Sunday, 2 to Monday etc. If anything other than a number between 1 and 7 inclusive is entered, the following message is printed

   ```
   Incorrect day
   ```

   `HH:MM` is time in 24-hour format, i.e. it ranges from 00:00 to 23:59. If that is not the case, the following message is printed

   ```
   Incorrect time
   ```

   (b) `end D HH:MM` This designates the day and time the phone call ended in the same way as in the `start` command.

   (c) `compute` This command computes and outputs the charge for the phone call. It uses the time information from the last `start` and `end` commands entered. For the purposes of this question, you can assume that each call starts and ends exactly at the beginning of the minute specified with the `start` and `end` commands. If either or both of the start and end times have not been entered, the system prints the following message:

   ```
   Time information missing.  Please re-enter both start and end time.
   Press Enter to continue.
   ```

   Both start and end times are cleared if this error occurs.

(d) `clear` This command clears both the start and end times. They will have to be re-entered before a charge can be computed.

(e) `quit` This command exits the program.

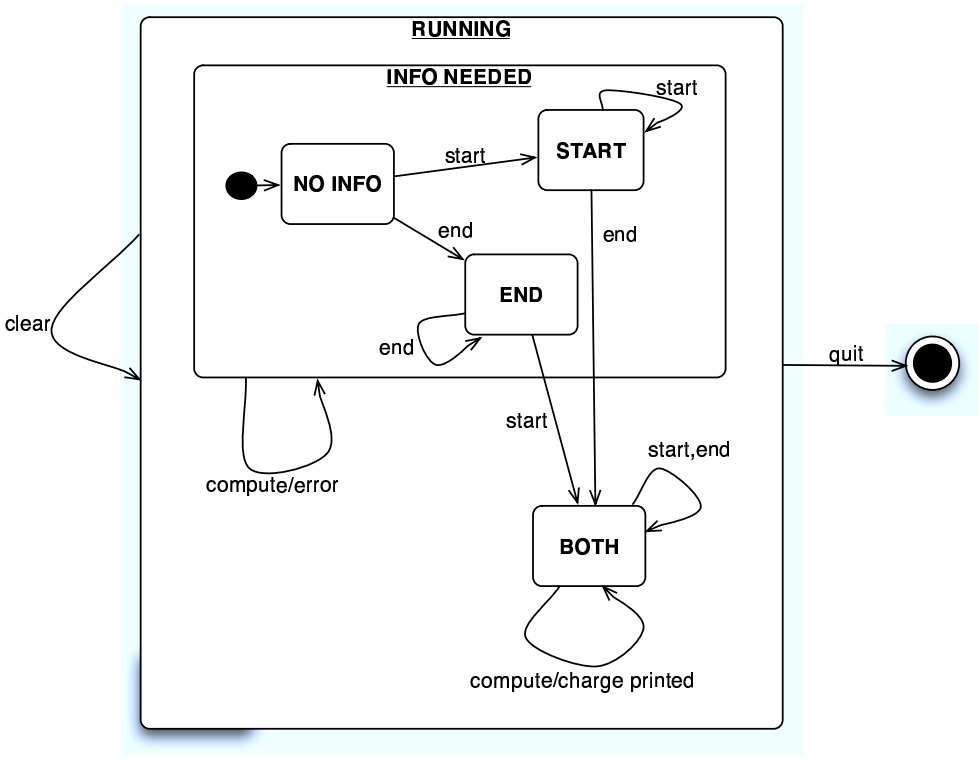
If any other command is entered, the system will simply print

```
Incorrect command
```

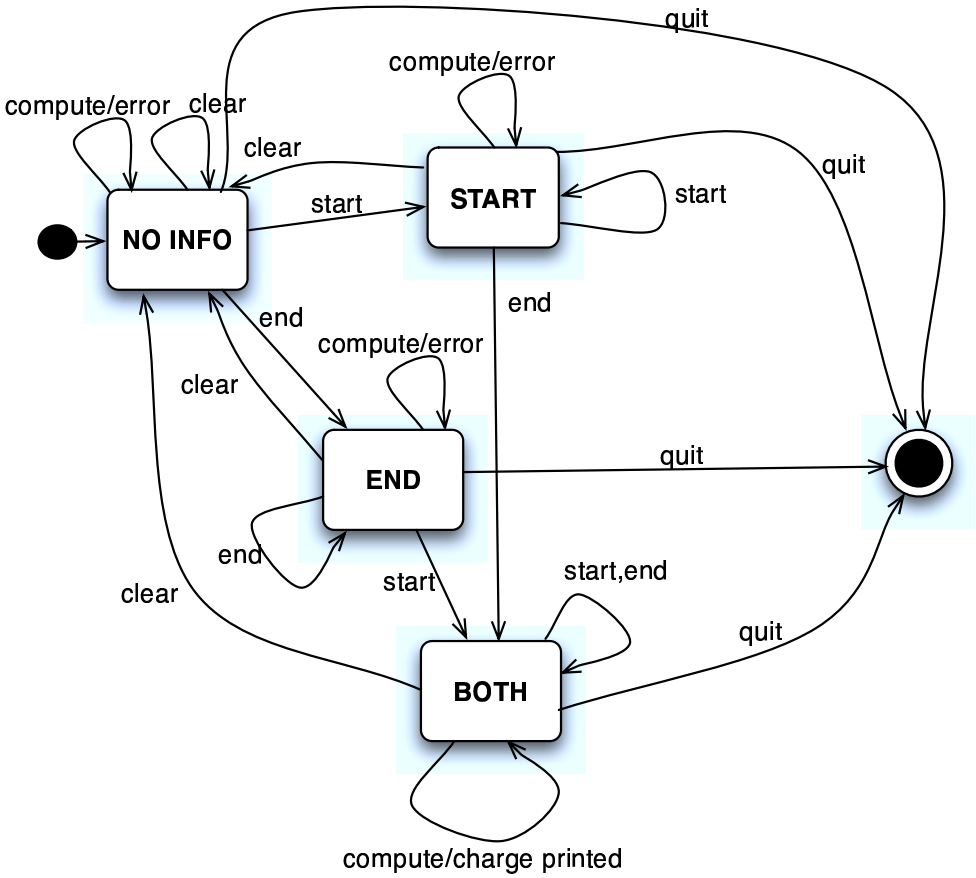
Two different testing strategies are applicable to this system (each worth 30 marks):

1. State-based testing

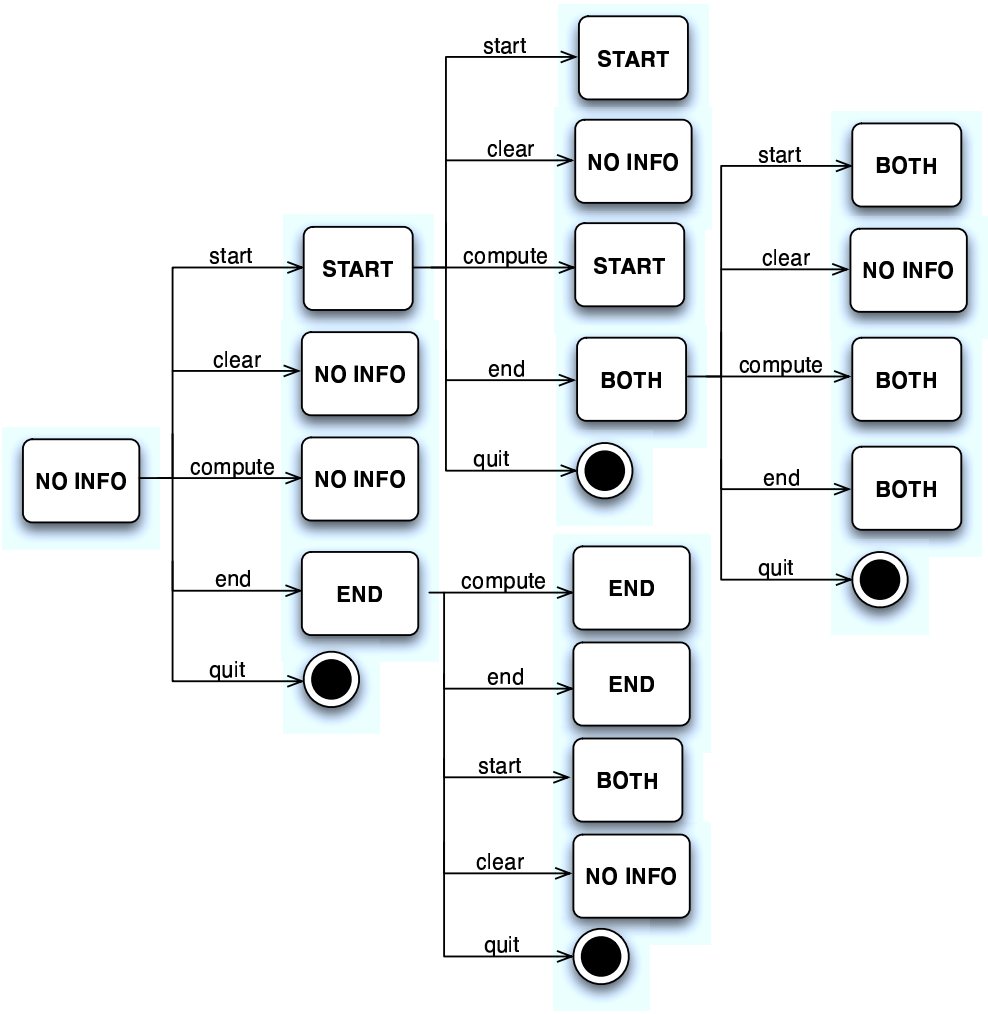
The behaviour of the system can be described with the following statechart (10 marks).



The statechart can be expanded to the following state transition diagram (10 marks).



A conformance test suite can be derived using the N+ testing strategy (10 marks). The derived test cases are shown on the round-trip path tree below.



Since the statechart specifies all combinations of states and events, there is no sneak path test suite to create.

2. Domain analysis

There are five interesting variables to consider:

1. Specified date[4 marks]. The acceptable range is 1-7. Choosing one ON and one OFF point at each boundary, one should test for 0,1,7, and 8.

2. Specified time[4 marks]. Hours and minutes must be handled separately. Similarly to the specified date, test cases for hours include -1, 0, 23, and 24, while for minutes -1, 0, 59, 60.

3. Specified command[4 marks]. Most test cases will involve ON points, so only an OFF point is necessary here. An interesting choice would be a variation on the longest command name, e.g. computf.

4. Price calculation [9 marks]. Calls of less than 0 minutes, or more than 10079 minutes are impossible to implement with the current interface, so interesting test cases here are: 0, 20, 21, 10079.

5. Call duration calculation [9 marks]. One of the risks associated with the system is that call duration is calculated incorrectly when a phone call is active at the hour, day, and week boundary. Test cases need to be developed for each boundary. For example, does the following sequence of commands produce a charge of 5 cents?

```
start 7 23:59
end 1 00:00
compute
```

The interesting variables may be identified in a different fashion but all the test cases derived above need to be covered.