

Please fill in your Student Number and Name.

Student Number : \_\_\_\_\_

Name:  
\_\_\_\_\_  
\_\_\_\_\_

Student Number:  
\_\_\_\_\_

University of Cape Town ~ Department of Computer Science  
Computer Science 3003S ~ 2009  
**Test 1**

| Question     | Max       | Mark | Internal | External |
|--------------|-----------|------|----------|----------|
| 1            | 12        |      |          |          |
| 2            | 10        |      |          |          |
| 3            | 8         |      |          |          |
| 4            | 15        |      |          |          |
|              |           |      |          |          |
|              |           |      |          |          |
|              |           |      |          |          |
|              |           |      |          |          |
| <b>TOTAL</b> | <b>40</b> |      |          |          |

**Marks : 40**  
**Time : 45 minutes**  
**Instructions:**

- a) Answer all questions.
- b) Write your answers in the space provided.
- c) Show all calculations where applicable.

**Question 1 – Lexical Analysis [12]**

a) What is a Deterministic Finite Automaton (DFA)? [2]

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

*A finite state machine [1]. At any given state, for any given symbol, there exists at most 1 transition [1].*

b) Consider the following regular expression and describe the strings it matches: [3]

$[ 0 - 9 ] ( [ a - z ] | [ 0 - 9 ] ) ^ *$

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

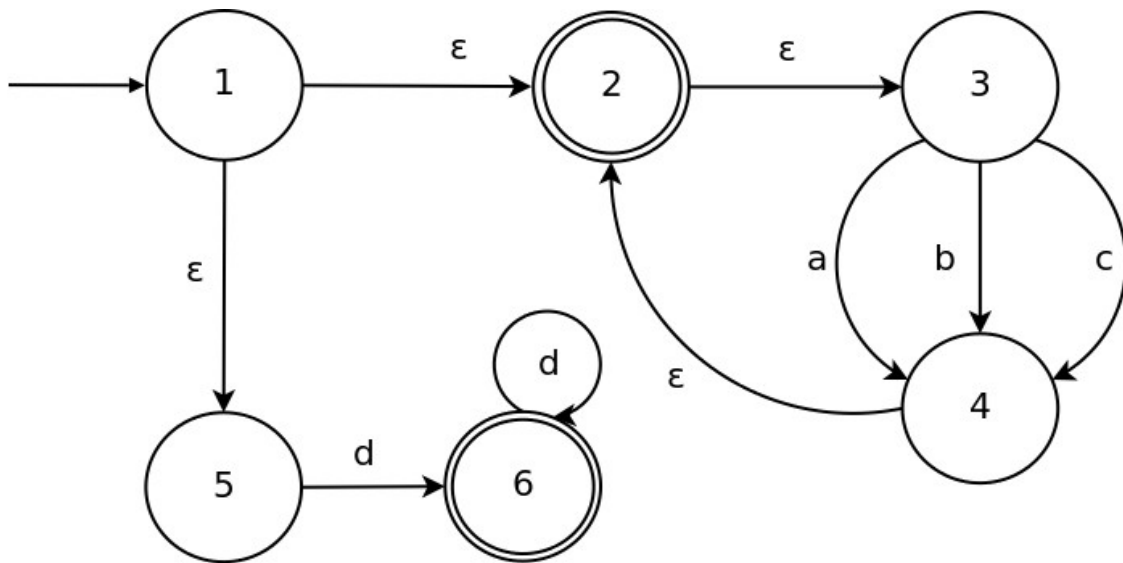
---

---

---

*Strings starting with a digit [1], followed by 0 or more [1] letters or digits [1]*

c) Convert the Non-Deterministic Finite Automaton (NFA) below to a Deterministic Finite Automaton (DFA). Remember to indicate the start and final state(s). [7]

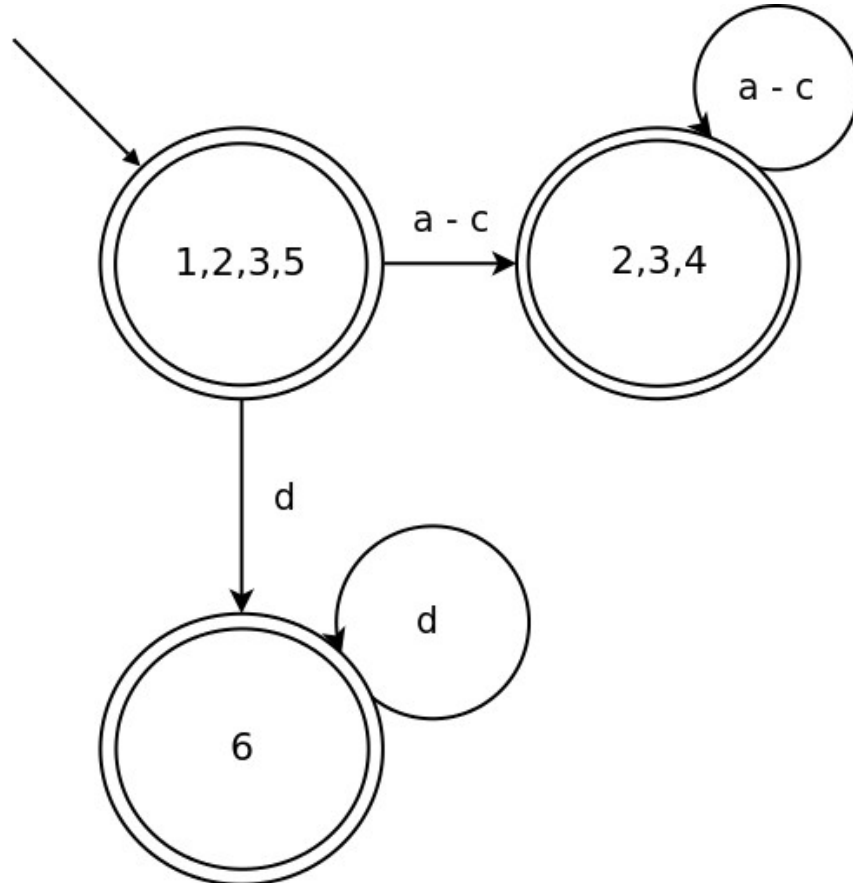


Solution DFA shown below

½ Mark for correct state (labels may differ), maximum 1½ marks

½ Mark for showing that state is final, maximum 1½ marks

1 Mark per correctly labelled transition (if student shows separate edges for a, b and c then only 1 mark is given for all of them), maximum 4 marks



**Question 2 – Parsing [10]**

a) Consider the following grammar and complete the table below by calculating its nullable, FIRST and FOLLOW sets (\$ should be treated as a terminal): [5]

- S → E\$
- E → aFc
- E → aF
- F → b
- F →

|   | nullable | FIRST | FOLLOW |
|---|----------|-------|--------|
| S |          |       |        |
| E |          |       |        |
| F |          |       |        |

*Solution table shown below*

*1 Mark for correct nullable column*

*1 Mark for correct FIRST column*

*1 Mark for correct FOLLOW entry for E*

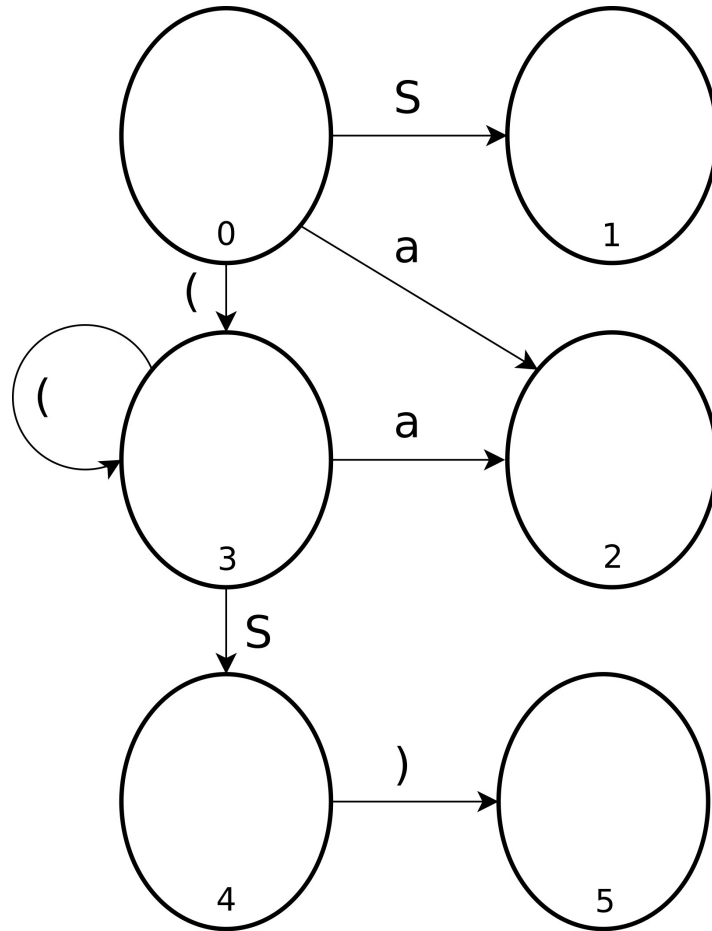
*2 Marks for correct FOLLOW entry for F (ignore comas)*

|          | <i>nullable</i> | <i>FIRST</i> | <i>FOLLOW</i> |
|----------|-----------------|--------------|---------------|
| <i>S</i> | <i>FALSE</i>    | <i>a</i>     |               |
| <i>E</i> | <i>FALSE</i>    | <i>a</i>     | <i>\$</i>     |
| <i>F</i> | <i>TRUE</i>     | <i>b</i>     | <i>c, \$</i>  |

b) Consider the following grammar and complete the LR(0) automaton below:

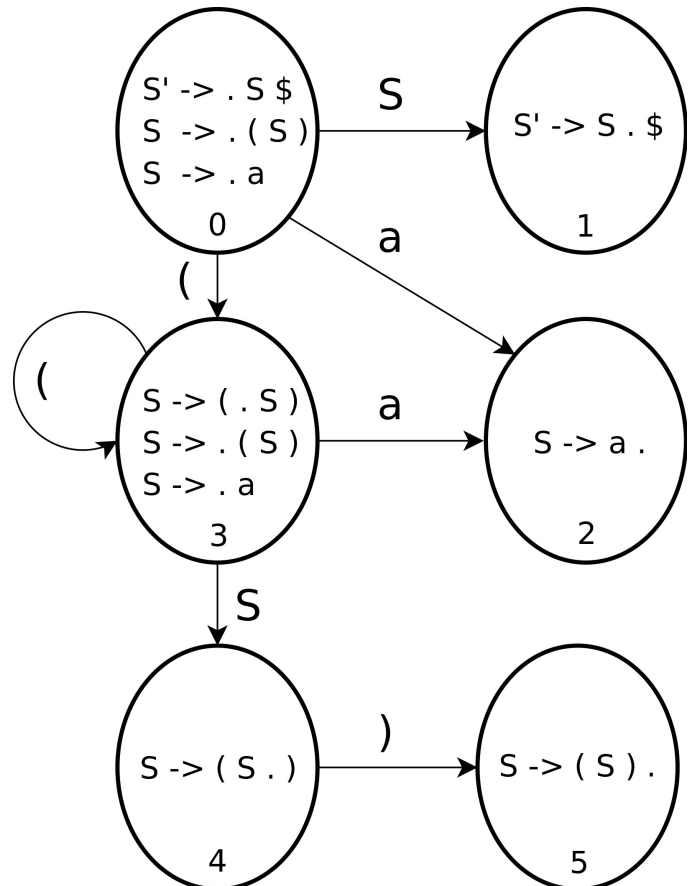
[5]

- $S' \rightarrow S\$$
- $S \rightarrow (S)$
- $S \rightarrow a$



*Solution automaton on the right*

*½ Mark for each correct LR(0) item (production with a dot), max 10 items => 5 marks*



### Question 3 – Error Recovery [8]

There are two main classes of error recovery techniques:

- Local Error Recovery
- Global Error Recovery

a) What is a difference between these two classes? [2]

---

---

---

---

---

---

*Global Error Recovery adjusts the input before the point where the error was detected [1]  
Local Error Recovery adjusts the input stream at the point of error (or immediately after) [1]*

b) What adjustments can be made to the input stream to enable parsing to continue? [3]

---

---

---

---

---

---

*Tokens can be deleted [1], inserted [1] or replaced/substituted [1]*

Burke-Fisher is an example of an error recovery technique.

c) Given a token window size of 5, how many alterations can be made to the input stream to enable parsing to continue? [1]

---

---

---

---

*One [1]*

d) For which edge case(s) does this technique fail (assuming there are no semantic actions that affect lexical analysis) and why? [2]

---

---

---

---

---

---

---

---

---

---

---

---

*Burke-Fisher works by single token manipulation of the input stream [1], thus it fails if there is no single token adjustment that will allow parsing to continue [1].*

*i.e. 1 mark for stating single token manipulation and 1 mark for linking it to edge case*

#### Question 4 – Semantic Analysis [15]

Semantic analysis follows syntactic analysis and is the process of determining context-sensitive information and thereafter generating code. Answer the following questions related to semantic analysis.

- a) The starting point in semantic analysis is usually an Abstract Syntax Tree. What is the difference between a Concrete Syntax Tree and an Abstract Syntax Tree? [2]

---

---

---

---

---

*Concrete syntax tree contains all tokens from a grammar as derived from parsing. Abstract syntax trees contain equivalent semantic information in concise tree structures.*

- b) Symbols tables are used to perform context-sensitive checking. Briefly describe 2 distinct checks that may be conducted using a symbol table. [2]

---

---

---

---

---

*variables defined before use [1], array indices in bounds [1], unused variables [1], etc.*

- c) Scope must be taken into account in a symbol table. How does scope differ from lifetime? [2]

---

---

---

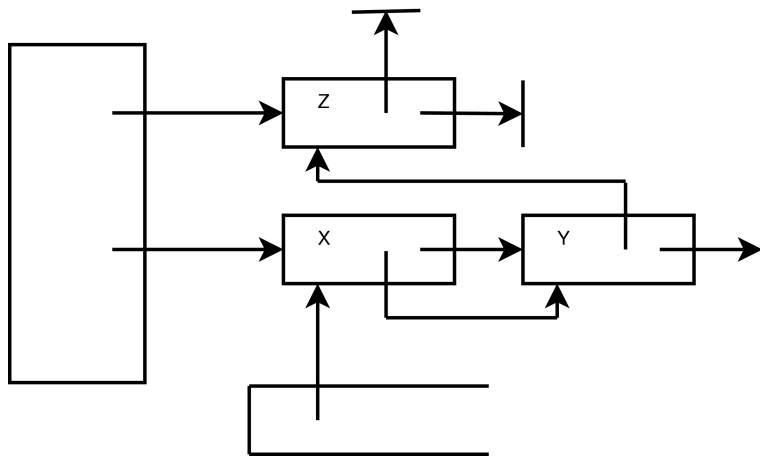
---

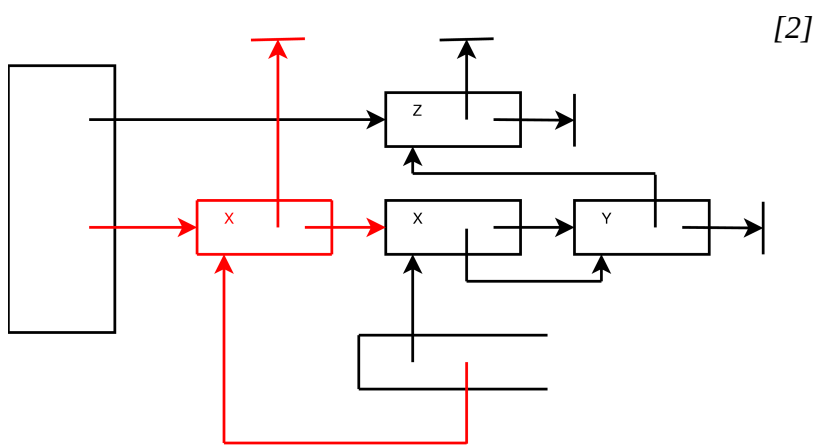
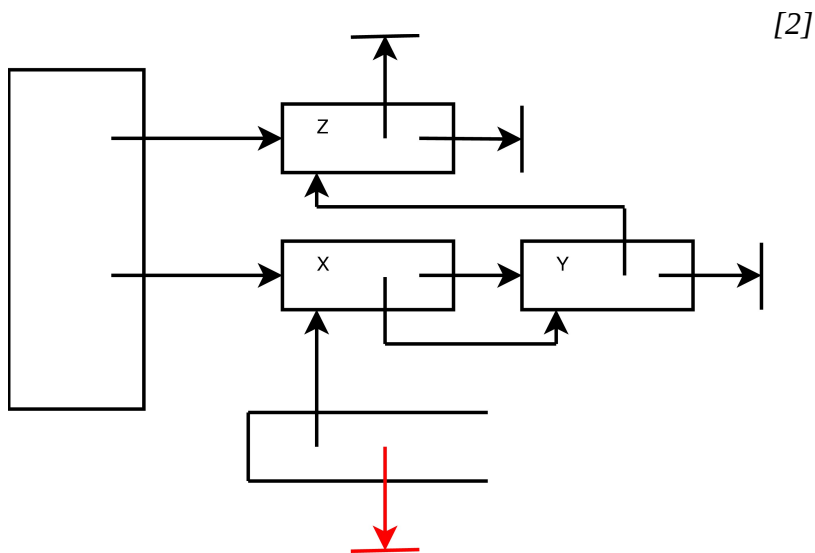
---

*scope = where/when a variable can be accessed [1]; lifetime = where/when a variable has memory associated with it [1]*

- d) Show diagrammatically how the following **imperative hash table** implementation of a symbol table is modified (A) when a scope is opened and (B) when a definition of X is added to that scope. [4]







e) Why is an activation record often called a stack frame? Is it always accurate to interchange these terms? Explain your answer. [2]

---



---



---



---



---

*it is usually stored on the stack [1]*

*no. [no mark] if there is no recursion the activation records need not be stored on the stack [1]*

f) The following DDD assembler source code listing corresponds to the unoptimised C function provided. In the context of subprogram invocation mechanics, explain the purpose of each of the marked lines of code (A, B and C). [3]

```

A   push   %ebp
    mov    %esp,%ebp
B   sub    $0x10,%esp
    mov    0x8(%ebp),%eax
    add    $0x1,%eax
    mov    %eax, -0x4(%ebp)

```

```
C    mov    -0x4(%ebp),%eax
      leave
      ret
```

```
int testsub ( int x )
{
    int a = x+1;
    return a;
}
```

---

---

---

---

---

---

---

---

*A – save frame pointer [1]*

*B – create activation record or frame on stack [1]*

*C – set return value in register[1]*