MSc in COMPUTER SCIENCE

BASIC PAPER

D0b

QUESTION PAPER

DUMMY COVER

ORIGINAL

EXAMINERS COPY

EXAM Co-ordinator

Peter Rounce

Ext 7291

2 HOURS 30 MINUTES

ANSWER ALL 3 QUESTIONS

Electronic calculators are not permitted.

Each question carries a mark of 25. The total mark will be scaled to a percentage.

Question 1

Answer part (a) and either part (b) or part (c) for this question.

(a) **Answer this part**

(i) Explain what we mean when we say that one decision problem reduces to another in polynomial time. [Marks 4]

(ii) Prove that *p*-time reduction is reflexive and transitive. [Marks 4]

(iii) Define the class of all **NP-hard** decision problems. [Marks 4]

(b) **Answer either this part or part (c)**

Let *A, B* be decision problems and suppose (i) there is a deterministic algorithm that solves *B* in worst-time $2e^n$ for any instance of *B* of size *n* and (ii) there is a reduction of *A* to *B* that runs in worst-time n^3 for any instance of *A* of size *n*.
What can you conclude about the complexity of *A* from this? [Marks 13]

(c) **Answer either this part or part (b)**

(i) Define the *Hamiltonian Circuit Problem* (HCP) and the *Hamiltonian Path Problem* (HPP). [Marks 3]

(ii) Give a *p*-time reduction of HPP to HCP. [Marks 6]

(iii) Assuming that HPP is **NP-complete** what can you deduce about the complexity of HCP? [Marks 4]

END OF QUESTION 1

TURN OVER

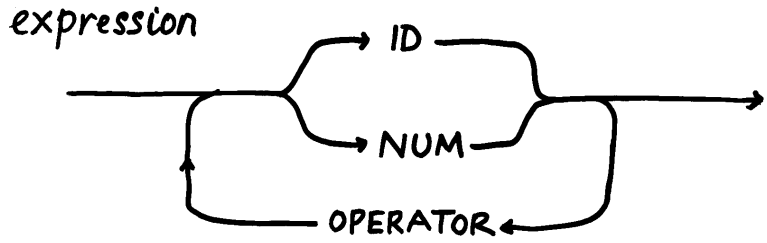
Question 2

D0B 1999

Answer part (a) and EITHER part (b) OR part (c)

a) Answer this part.

i) Give *three* reasons why the following syntax diagram is not suitable for the implementation of fully-parenthesised arithmetic expressions in a recursive-descent compiler.



[4 marks]

ii) The following are three examples of *actual-parameter-list*s:

( ) (x) (1, x+1, (x))

Draw syntax diagram(s) to define *actual-parameter-list*. You may assume that *expression* has already been defined.

Add lookaheads to your syntax diagram(s) to aid parsing.

[3 marks]

QUESTION 2 CONTINUES ON NEXT PAGE

b) **Answer either this part or part (c)**

Suppose it is desired to add a new construct to C/C++, a **repeat** statement, which executes a statement a number of times determined by the value of an expression. Thus

```
repeat (20)
        printf("*");
```

would call printf twenty times.

i) The following code would also call printf twenty times

```
n = 10;
repeat (2*n) {
        printf("*");
}
```

Show the kind of target code which might be generated from this source code for execution on a stack-based machine. Translate 'printf("*");' as 'print "*"'.

ii) Draw a syntax diagram for the repeat construct. You may assume that expression and statement are already defined.

iii) Annotate your syntax diagram to generate code for a stack machine.

You may need to make some choices about the semantics of your implementation. Make these choices explicit.

iv) Sketch an implementation of your annotated syntax diagram as a function in a recursive-descent compiler. Briefly describe the nature and purpose of any other functions or variables that you use.

Your implementation should work for nested constructs, and you should briefly justify that your implementation does so.

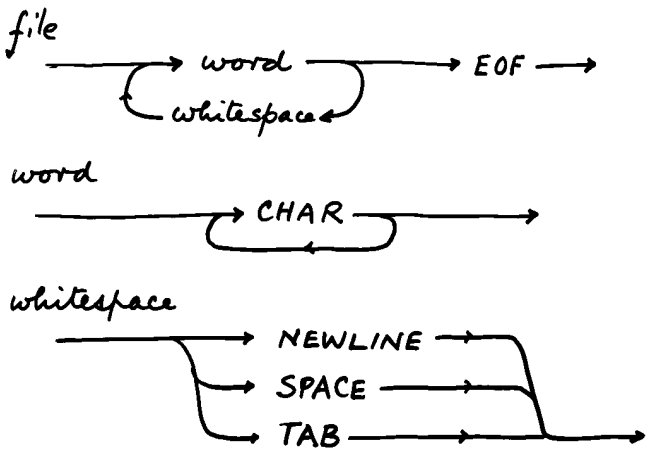
[18 marks]

QUESTION 2 CONTINUES ON NEXT PAGE

c) Answer either this part or part (b)

The Unix command *wc* counts and prints the number of lines, words, and characters in a file. It can process any kind of file, of arbitrary content. The line count is simply a count of the number of newline characters in the file. A word is defined to be a maximal string of characters delimited by whitespace. Whitespace consists of the characters space, newline, and tab (vertical tab and form feed may be ignored here). All characters are included in the character count.

The following syntax diagrams are a first attempt at a definition of a *file* for processing by *wc*:



In the syntax diagrams the token CHAR represents any character except whitespace characters. EOF is a token representing end-of-file.

The syntax diagrams given define only a subset of the files which *wc* would process.

i. Identify and state the limitations of the given definitions.
ii. Draw syntax diagrams which define *file* fully.
iii. Add lookaheads to your syntax diagrams to aid parsing.
iv. Annotate (with lassos and actions) your syntax diagrams so that they could form the basis of an implementation of *wc*.
v. Sketch an implementation of *wc* in pseudo-code.

[18 marks]

[25 marks for this question]

END OF QUESTION 2

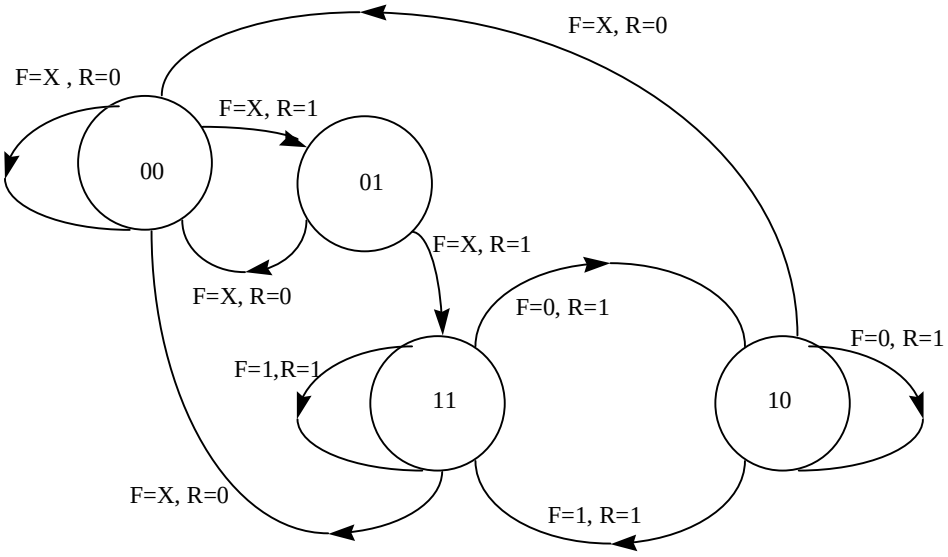
Question 3

D0B 1999

Answer 3 parts from (a)-(f):

(a) Outline the structure and operation of Finite State Machines as used in digital electronic circuits and explain why they are useful. [8 Marks]

(b) Draw the truth table for the Finite State Machine (FSM) with the following State Diagram:



Use Karnaugh maps or other technique to derive the equations for the next state bits. Draw the final circuit for the finite state machine using AND, OR and INVERTER gates, and flip-flops for the state store. [8 Marks]

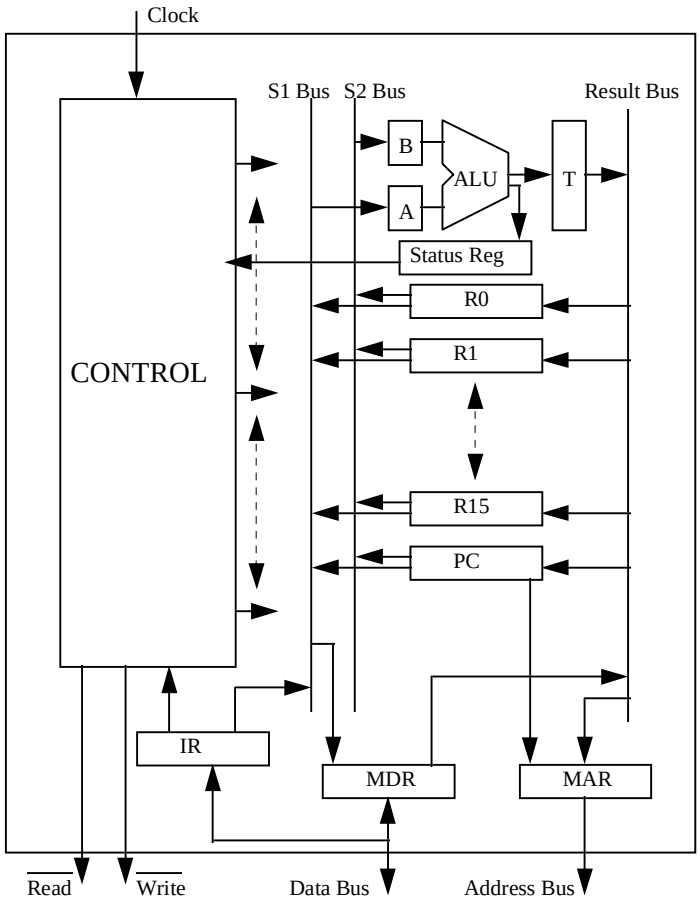
(c) Show how a 2x2 bit memory can be built from one-bit transparent flip-flops and tri-state devices, and briefly discuss its operation. [8 Marks]

(d) Draw a diagram of the physical structure of either a p-n-p or a n-p-n transistor, and describe its operation. [8 Marks]

(e) What is the role of transistors in digital electronic circuits, in particular CMOS circuits. Illustrate your answer with a description of a CMOS NAND or NOR Circuit. [8 Marks]

QUESTION 3 CONTINUES ON NEXT PAGE

(f) Describe the function of the major elements of the CPU in the diagram below and give a general outline of its operation.



[8 MARKS]

End of Question 3

END OF PAPER