Monash University
School of Computer Science & Software Engineering
Exam – 2005

CSE3322 – Programming Languages and Implementation

Total Time Allowed: 3 Hours

1. Reading time is of 10 minutes duration.
2. Examination time is of 3 hours duration.
3. The total marks are 100.
4. Questions which require explanations will receive no marks unless an appropriate explanation is given.
5. All questions should be attempted.
6. All questions should be answered in the exam paper itself.

Fill in your name and Monash Student ID.

Name: _____ Student ID: _____

(Extra space for working)

Question 1 [6 marks]

Each of the following pieces of SML code is inefficient. For each one, write an efficient piece of code that achieves the same result. For each answer state why your solution is more efficient.

1. `length s > 0` [2 marks]

2. `take n (map f s)`
   Note: `take` *and* `map` *are standard functions from the list in the appendix.* [2 marks]

3. `map f (map g s)` [2 marks]

Question 2 [6 marks]

Consider the following ML function

```
fun foldl f z []      = z
  | foldl f z (x::xs) = foldl f (f z x) xs;
```

1. What does the SML compiler infer as the type of the above function foldl? [2 marks]

2. Briefly describe how the SML compiler infers this type. [2 marks]

3. Write an SML expression that uses foldl to find the largest value in a list of integers, xs. [2 marks]

Question 3 [6 marks]

The following program does not work in SML.

```
val rec ns = 0 :: map (fn n => n+1) ns;
take 5 ns
```

Note: `take` *is a standard function from the list in the appendix.*

1. Explain concisely why this program does not work. [2 marks]

2. Name the feature that ML would need to have for this program to work. [2 marks]

3. Give the value that would be computed given the feature under (2) [2 marks]

Question 4 [6 marks]

In SML it is not possible to write a function `sort` which has just one parameter, that parameter being a list, and which can sort a list of integers, or a list of reals, or a list of strings, etc.

1. Describe the reason why this is not possible in SML. [2 marks]

2. Name the language feature that Haskell has which makes the task possible. [2 marks]

3. Describe the standard "work around" for this general problem in SML? [2 marks]

Question 5 [5 marks]

There are four functions defined in the following SML program and five code sections. For each code section list the functions that are *in scope* for that code section.

```
let fun a() =
      let fun b() = ...code1...
      in ...code2...
    and c() =
      let fun d() = ...code3...
      in ...code4...
in ...code5...
```

Question 6 [8 marks]

The following program is in a hypothetical programming language which resembles C but which may use a different parameter-passing mechanism.

```
int i;
int a[4];

void p( int m, int n )
 { i++;
   m++;
   n++;
 }

void main()
 { i = 1;
   a[0] = 0;  a[1] = 1;  a[2] = 2;  a[3] = 3;

   p( i, a[i] );

   write( i );
   write( a[0], a[1], a[2], a[3] )
 }
```

For each of the following parameter-passing mechanisms write down the output of the program. Also explain for each answer how this output is computed.

1. By value (by input) [2 marks]

2. By value-result (by input-output) [2 marks]

3. By reference [2 marks]

4. By name [2 marks]

Question 7 [6 marks]

The following program is in a hypothetical programming language which somewhat resembles C but which may use a different binding mechanism.

```
procedure a()
 { print 1 };

procedure b()
 { procedure a()
    { print 2 };

   c()
 };

procedure c()
 { a() };

procedure main()
 { b();
   c()
 }
```

1. What would the program print if the language used *dynaming binding*? Explain your answer. [2 mark]

2. What would the program print if the language used *static binding*? Explain your answer. [2 mark]

3. State a possible advantage and a possible disadvantage of static binding vs. dynamic binding. [2 mark]

Question 8 [7 marks]

1. What is 'overloading' as a programming language concept? [1 mark]

2. What is 'context-independent overloading' as a programming language concept? [1 marks]

3. What is 'context-dependent overloading' as a programming language concept? [1 marks]

4. Does SML use 'context-independent overloading' or 'context-dependent overloading'? [1 marks]

5. Give a simple expression which is ambiguous under 'context-dependent overloading'. [1 marks]

6. Give two different meanings for the expression in your answer to the previous part. [2 marks]

Question 9 [8 marks]

1. What is the purpose of a 'garbage-collector' in an implementation of a programming language? [2 marks]

2. Describe how a 'mark-scan' garbage-collector works. [2 marks]

3. Give an advantage and a disadvantage of a programming language having garbage-collection. [2 marks]

4. Give two features of a programming language that require, or strongly imply, the use of a garbage-collector in implementations of the language. [2 marks]

(Extra space for working)

Question 10 [9 marks]

Consider the following grammar with terminal symbols

$$b \quad c \quad d$$

non-terminal symbols A, B where B is the start symbol and with the productions

$$\begin{array}{lcl} B & \rightarrow & c\ A\ B \\ B & \rightarrow & c\ c\ c \\ A & \rightarrow & A\ b\ b\ A \\ A & \rightarrow & d\ d \end{array}$$

(a) Explain concisely why this grammar cannot be used directly (ie. without any modifications) to implement a recursive descent parser. Name the technique that you would use to rectify this problem. [2 marks]

(b) Give a modified grammar that recognises the same language but is suitable as the basis of a recursive descent parser. [2 marks]

(c) Is your solution to part (b) a LL(k) grammar? If no, explain why not. If yes, show what is the smallest k for which it is LL(k)? [2 mark]

(d) In the space below, write down an ML function that implements the recursive descent function for the non-terminal B. You can assume that a datatype for tokens is given as

```
datatype token = b | c | d;
```

Note that the function does not *need to return a parse tree.* [3 marks]

(Extra space for working)

Question 11 [16 marks]

Consider the following grammar with terminal symbols

$a \quad b \quad c \quad d$

non-terminal symbols S, A, B where S is the start symbol and with the productions

Number	*Production*
1	$S \rightarrow A$
2	$S \rightarrow B$
3	$B \rightarrow c\ A\ B$
4	$B \rightarrow b\ c$
5	$B \rightarrow \epsilon$
6	$A \rightarrow a\ B\ d$
7	$A \rightarrow d\ A\ d$

(a) Compute $FIRST(S)$, $FIRST(A)$, $FIRST(B)$. [3 marks]

(b) Compute $FOLLOW(S)$, $FOLLOW(A)$, $FOLLOW(B)$. [3 marks]

(c) Fill in the production numbers in the LL(1) parsing table below for a predictive table parser for this grammar. [6 marks]

	a	b	c	d	$
S					
B					
A					

(d) Detail using the table below how the sentence *d a c a b c d d d* would be parsed with a predictive table parser using the LL(1) parsing table you gave in (c). For each step of the process give the parser actions (production number / *advance* / *accept*), the stack state and the unprocessed rest of the input.
Note that you may not need all lines of the table. [4 marks]

ACTION	STACK	INPUT

Question 12 [17 marks]

Consider the following grammar with terminal symbols

$a \quad b \quad c \quad d$

non-terminal symbols S, A, B where S is the start symbol and with the productions

Number	*Produktion*
1	$S \rightarrow A\ B$
2	$B \rightarrow c\ A\ B$
3	$B \rightarrow b\ c$
4	$B \rightarrow \epsilon$
5	$A \rightarrow a\ B\ a$
6	$A \rightarrow d\ A\ d$

(a) Give the canonical collection of LR(0) items for this grammar. When computing the item sets, use the symbols in the order a, b, c, d, S, A, B. [7 marks]

(b) Fill in the **SLR parsing table** for this grammar in the table below. *Note that you may not need all rows, i.e. there may be fewer than 16 canonical items.* [5 marks]

	ACTION					*GOTO*		
STATE	*a*	*b*	*c*	*d*	$	S	A	B
0								
1								
2								
3								
4								
5								
6								
7								
8								
9								
10								
11								
12								
13								
14								
15								

(c) Detail using the table below how the sentence *a a c d a a d* would be parsed with a LR parser using the table you gave in (b). For each step of the process give the parser actions (*shift*/*reduce* and production number) and stack state.
Note that you may not need all rows. [4 marks]

ACTION	STACK	INPUT

(d) The sentence *a a c d b a a d* is **not** in the language defined by this grammar. Give the stack state and the current input symbol at the point where the LR parser discovers the syntax error. Describe an appropriate way to recover parsing at this point. [1 mark]