Monash University
School of Computer Science & Software Engineering
Sample Exam – 2004

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. All questions should be attempted.
5. Question 1 should be answered in the exam paper itself, the remaining questions in a script book.

Fill in your name and Monash Student ID.

Name: ____________________________ Student ID: ____________________________

Question 1 [30 marks]

Answer the following multiple choice questions by ticking the box corresponding to the statement which best answers the question. If you wish to change your answer then cross out the wrong box and tick the new box. You receive 2 points for each correct answer.

(a) Simula 67 is famous because it was

- ☐ the first database programming language
- ☐ the precursor to Prolog
- ☐ the precursor to Basic
- ☐ it was designed for graphical user interfaces
- ☐ the first object-oriented programming language

(b) A **signature** in SML is

- ☐ used to initialise global variables in a structure
- ☐ a higher-order function
- ☐ a higher-order function with multiple arguments
- ☐ a function used to handle exceptions
- ☐ none of the above

(c) What will the SML function `mystery` defined as follows do?

```
fun mystery c = map real c;
```

- ☐ convert the elements of a list of integers to a list of reals
- ☐ covert an integer to a real number
- ☐ convert a character to a real number
- ☐ give a syntax error
- ☐ none of the above

(d) Consider the SML program

```
fun dummy [] = 0
 |  dummy [x] = 0
 |  dummy (x::y::xs) = dummy(xs) + x;
```

What does the expression `dummy [4,5,6]` evaluate to?

- ☐ `val it = 15 :  int`
- ☐ `val it = 9 :  int`
- ☐ `val it = 15 :  real`
- ☐ `val it = 9 :  real`
- ☐ None of the above.

(e) Consider the SML program

```
fun silly f g x = f (g x)
```

What does the expression `silly ~ (fn x => 2*x)` evaluate to?

- ☐ `val it = ~2 : int`
- ☐ `val it = 2 : int`
- ☐ A function which takes an integer and negates it
- ☐ A function which takes an integer and multiplies it by -2
- ☐ Gives a type error because an argument to `silly` is missing.

(f) Which of the following statements about DNA programming is **not** true

- ☐ one of the first problems DNA programming was used to solve was the Hamiltonian path problem
- ☐ DNA programs can be used to solve the halting problem for Turing machines
- ☐ in a DNA program it is more expensive to read DNA strands than to write them
- ☐ in theory you can replicate a DNA strand 2^n times in time proportional to n
- ☐ a problem with DNA computing is that long DNA strands are fragile and may break.

(g) Consider the query `state(City,vic)` run with the Prolog program:

```
state(adelaide,sa).
state(sydney,nsw).
state(newcastle,nsw).
state(geelong,vic).
state(melbourne,vic).
```

What is the second answer found?

- ☐ `City = sa`
- ☐ `City = nsw`
- ☐ `City = newcastle`
- ☐ `City = melbourne`
- ☐ none of the above

(h) Why does SML use static binding rather than dynamic binding? Because:

- ☐ dynamic binding doesn't work well with compile-time type checking and inference
- ☐ it's an implementation decision which does not change the meaning of a program
- ☐ dynamic binding only works in imperative programming languages
- ☐ there is no need to since abstract datatypes provide this functionality
- ☐ context dependent overloading only works with static binding

(i) Which of the following is **not** true about the language SML?

- ☐ it provides pattern matching
- ☐ it provides type classes
- ☐ it has polymorphic types
- ☐ it has data constructors
- ☐ it has functors

(j) Recall that Cascal is the hypothetical programming language introduced in the lectures. Consider the Cascal program:

```
int s = 2;
int d = 4;

int function temp(void) {
  return d+s;
}

void main(void) {
  int s := 1;
  int d := 2;
  s := temp();
  writeln( s );
}
```

What will be written by the above program if Cascal uses **static binding**?

- ☐ 6
- ☐ 3
- ☐ 2
- ☐ 1
- ☐ none of the above

(k) Consider the Cascal program:

```
int function temp(int x, int y) {
  x := y + 1;
}

void main(void) {
  int s := 3;
  int t := 4;
  temp(s,t);
  writeln(s);
}
```

What will be written by the above program if Cascal uses **value-result** parameter passing?

- ☐ 3
- ☐ 4
- ☐ 5
- ☐ 6
- ☐ nothing, but it will generate a run-time error

(l) Which of the following items is **not** usually stored in an activation record?

- ☐ procedure parameters
- ☐ return address
- ☐ global variables
- ☐ local variables
- ☐ temporary variables

(m) In which phase of a compiler is the **type** of a variable typically determined?

- ☐ lexical analysis
- ☐ screening
- ☐ syntactic analysis
- ☐ semantic analysis
- ☐ target code generation

(n) Which of the following **parsing methods** can process the largest class of grammars?

- ☐ LL(1) parsers
- ☐ canonical LR parsers
- ☐ recursive descent parsers without backtracking
- ☐ SLR parsers
- ☐ CYK parsers

(o) Which of the following is **not** a machine independent code optimisation?

- ☐ Tail-recursion optimisation
- ☐ Procedure call inlining
- ☐ Peephole optimisation
- ☐ Elimination of dead code
- ☐ Moving loop-invariant computation from inside loops

Question 2 [6 marks]

Write an SML function

```
last :'a list -> 'a
```

such that `last xs` returns the last element in the list `xs`. For example, evaluating `last [4.0,2.0,3.0]` should return the value 3.0. If the list is empty the function should raise an appropriate exception. This should be declared.

Question 3 [8 marks]

Write an SML function

```
find : ('a -> bool) -> 'a list -> 'a
```

such that `find p xs` returns the first element in the list `xs` for which the function `p` returns `true`. For example, evaluating `find (fn x => (x < 4.0)) [5.0,2.0,3.0]` should return the value 2.0. If the function is not true for any element in the list `find` should raise an appropriate exception. This should be declared.

Question 4 [16 marks]

You are required to write functions to manipulate simple geometric shapes. Each shape is either a circle, with a center and a radius, or a box, with a center and height and width. All attributes are reals.

(a) Define an SML datatype `Shape` for representing a circle or box. [4 marks]

(b) Write two SML functions

```
circle : (real*real) -> real -> Shape
box : (real*real) -> real -> real -> Shape
```

which respectively create a new circle or box given their center and size. For instance

```
circle (1.0,2.0) 3.0
```

should return a new circle shape with center $(1.0, 2.0)$ and radius 3.0 while

```
box (1.0,2.0) 3.0 4.0
```

should return a new box shape with center $(1.0, 2.0)$, width 3.0 and height 4.0. [6 marks]

(c) Write an SML function

```
contains : Shape -> (real*real) -> bool
```

which takes a shape and a point and returns true if the point is strictly inside the shape and false otherwise. For instance, both

```
contains (circle (1.0,2.0) 3.0) (1.0,1.0)
contains (box (1.0,2.0) 3.0 4.0) (1.0,1.0)
```

should return `true` while both

```
contains (circle (1.0,2.0) 3.0) (10.0,10.0)
contains (box (1.0,2.0) 3.0 4.0) (10.0,10.0)
```

should return `false`. [6 marks] fun between x y z = x ¡ y andalso y ¡ z:real;

fun contains (Circle(x,y,r)) (x1,y1) = ((x1-x)*(x1-x) + (y1-y)*(y1-y))¡ r*r — contains (Box(x,y,w,h)) (x1,y1) = (between x-0.5*w x1 x+0.5*w) andalso (between y-0.5*h y1 y+0.5*h);

Question 5 [8 marks]

Consider the following grammar with terminal symbols

$a \quad b$

and non-terminal symbols S and X where S is the start symbol and productions

$$\begin{array}{lcl} S & \rightarrow & X \\ X & \rightarrow & a\ X \\ X & \rightarrow & a \\ X & \rightarrow & b\ X \\ X & \rightarrow & b \end{array}$$

(a) The above grammar recognizes a regular language. Give a regular expression for this language. [2 marks]

(b) Add attributes and attribute computation rules and conditions to the above grammar so that it recognises non-empty strings with an equal number of a's and b's. I.e., $aabbab$ should be in the language of this new grammar while $aaabb$ should not. You should indicate whether the attributes are inherited or synthesized. [6 marks]

Question 6 [12 marks]

Consider the following grammar with terminal symbols

$a \quad b \quad +$

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

$$
\begin{array}{llll}
(P1) & S & \rightarrow & A\ +\ B \\
(P2) & S & \rightarrow & B \\
(P3) & A & \rightarrow & a\ A \\
(P4) & A & \rightarrow & \epsilon \\
(P5) & B & \rightarrow & b\ B \\
(P6) & B & \rightarrow & \epsilon
\end{array}
$$

(a) Compute $FIRST(A)$, $FIRST(B)$, $FIRST(S)$. [2 marks]

(b) Compute $FOLLOW(A)$, $FOLLOW(B)$, $FOLLOW(S)$. [2 marks]

(c) Consider the following LL(1) parsing table for a predictive table parser:

	a	b	$+$	$
S	$P1$	$P2$	$P1$	
A	$P3$		$P4$	
B		$P5$		$P6$

where Pi refers to the i^{th} production in the above grammar. Detail how the sentence $a + b$ would be parsed with a predictive table parser using this table. For each step of the process give the parser action, input and stack state. [4 marks]

(d) Is the parsing table given in (c) the correct LL(1) predictive parsing table for this grammar? If not identify and correct the errors in the table. [4 marks]

Question 7 [10 marks]

Consider the augmented grammar which consists of the symbols and productions in the grammar given in Question 6 and the new start symbol S' and the additional production

$$(P0) \quad S' \quad \rightarrow \quad S$$

(a) Compute

- $I_0 = closure(\{S' \rightarrow \bullet S\})$
- $goto(I_0, a)$
- $goto(I_0, b)$
- $goto(I_0, +)$
- $goto(I_0, A)$
- $goto(I_0, B)$
- $goto(I_0, S)$ [5 marks]

(b) Consider the following LR parsing table for this grammar

STATE	*ACTION*				*GOTO*		
	a	*b*	+	$	*S*	*A*	*B*
0	*s*1	*s*2	*r*4	*r*6	5	3	4
1	*s*1		*r*4			6	
2		*s*2		*r*6			7
3			*s*8	*r*6			
4				*r*2			
5				*acc*			
6			*r*3				
7				*r*5			
8		*s*2					9
9				*r*1			

where

si is shift and stack state *i*

rj is reduce using production *Pj*

acc is accept.

Detail how the sentence $a + b$ would be parsed with an LR parser using this table. For each step of the process give the parser action (shift/reduce), input and stack state. [5 marks]

Question 8 [10 marks]

Consider the core ML program

```
val mystery = fn f => (fn x => f(f(x)))
```

(a) Give its syntax tree and assign a type variable to each subexpression. [3 marks]

(b) Generate a set of type equations (or constraints) on the type variables based on the annotated syntax tree from (a) [4 marks]

(c) Solve the type equations from (b) and give the type for `mystery`. [3 marks]