

MONASH UNIVERSITY
SCHOOL OF COMPUTER SCIENCE & SOFTWARE ENGINEERING
SAMPLE EXAM – 2002

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. You receive 2 points for each correct answer.

(a) FORTRAN was developed by

- IBM in the late 1930s
- S. Wolfram in the mid 1960s
- J. Bachus and his team in the mid 1950s
- J. Von Neumann in the early 1940s.

(b) Which of the following is **not** true about the language SML:

- it has type variables
- it has type constructors
- it has automatic type coercion
- it has automatic type deduction
- it has polymorphic types.

(c) Consider the SML program

```
fun dummy x [] = 1
  | dummy x (y::ys) = x (dummy x ys) ;
```

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

- `val it = 1 : int`
- `val it = 15 : int`
- `val it = ~6 : int`
- `val it = 120 : int`
- none of the above

(d) What is the type of the function `dummy` given in (c):

- `int -> 'a list -> int`
- `real -> 'a list -> real`
- `(int -> int) -> 'a list -> int`
- `(real -> real) -> 'a list -> real`
- none of the above.

(e) An **abstype** in ML is

- used to initialize data inside a structure
- a higher-order data structure
- hide the definition of higher-order functions
- define the interface of a structure
- none of the above

(f) What will the ML function **mystery** defined as follows do

```
fun mystery L = foldr (op +) 0.0 (map (fn x => x*x) L);
```

- add the elements of a list
- square the elements in a list and add them together
- square the elements in a list and add 0.0 to each one
- give a syntax error
- none of the above

(g) What is the type of ML function **mystery** defined above

- `real list -> real`
- `real list -> real list`
- `'a list -> real`
- `'a list -> 'a list`
- none of the above

(h) Given the Prolog program

```
p([]).  
p([a|X]) :- q(X).  
q([b|X]) :- p(X).
```

what is the first answer returned by the Prolog system for the query `p(Y)`

- `Y = [a]`
- `Y = [b]`
- `Y = [a, b, a]`
- `Y = [a, b, a, b]`
- none of the above

(i) Consider the overloaded operator I which denotes both the functions $f_1 : S_1 \rightarrow T_1$ and $f_2 : S_2 \rightarrow T_2$. Context dependent overloading requires that

- Types S_1 and S_2 are different.
- Types T_1 and T_2 are different.
- Types S_1 and S_2 are different or types T_1 and T_2 are different.
- Types S_1 and S_2 are different and types T_1 and T_2 are different.

(j) Consider the Cascal program:

```
int function tricky(int x, int y) {
    y := 11;
    x := y;
}
```

```
void main(void) {
    int y = 3;
    int x = 4;
    tricky(x,y);
    writeln(x+y);
}
```

What will be written by the above program if Cascal uses **call-by-reference** parameter passing:

- 7
- 14
- 21
- 6
- none of the above.

(k) Consider the Cascal program:

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

```
void main(void) {
    int s := 3;
    inc(s,s);
    writeln(s);
}
```

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

- 3
- 4
- 5
- 6
- it will generate a run-time error.

(l) In which phase of a compiler is type analysis typically performed?

- lexical analysis
- syntax analysis
- semantic analysis
- code generation
- language-independent optimization

(m) Consider the context-free grammar with terminal symbols a, b, c , non-terminal symbols A and B where A is the start symbol and productions

$$\begin{aligned} A &\rightarrow BAB \mid a \\ B &\rightarrow b \mid c \mid \epsilon \end{aligned}$$

Which of the following strings is **not** in the language of the grammar:

- abb
- $bcacb$
- $bbccab$
- $bcab$
- $bcacba$

(n) Which of the following statements is true for error correction in Burke-Fisher Parsing?

- it is a form of panic mode recovery
- it relies on LL(1) grammars
- it is a form of local error correction
- it works by modifying the input string
- it aborts after the first error

(o) Which of the following operations is **not** part of the language-independent optimization phase?

- moving invariants out of loops
- eliminating tail recursion
- eliminating constants
- selecting more efficient target code instructions
- in-lining procedure code

Question 2 [12 marks]

Define an ML function `intToString : int -> string` such that `intToString i` returns a string representation of integer `i` in decimal. Example:

```
intToString ~12345
```

has answer `it = "~12345" : string`. You should **not** call the library function `Int.toString!` Hint: the ML operators for integer division and remainder are `div` and `mod` while `chr` takes an integer and returns the corresponding ASCII character.

Question 3 [18 marks]

A file system contains files and directories. A file has a name which is a string and some contents which has type `char list`. A directory has a name and contains files and directories. It is convenient to consider both a file and a directory as “file systems” so that a directory contains file systems. Define

- (a) an ML datatype, `T`, for representing a file system. [6 marks]
- (b) a function `name : T -> string` which returns the name followed by a blank character. [4 marks]
- (c) a function `ls : T -> string` which returns the name of the file for a file argument and a string containing the names of all components of a directory argument. The predefined functions `map : ('a -> 'b) -> 'a list -> 'b list` and `concat : string list -> string` may be used in the solution. [8 marks]

Example: If `f` is a file and `d` is a directory the returned values could be:

```
ls f = "main.c "  
ls d = "a.out main.c main.o RCS "
```

Question 4 [10 marks]

- (a) Briefly explain how **call-by-name** parameter passing works. [4 marks]
- (b) Give an example of a language or system that uses call-by-name parameter passing. [2 marks]
- (c) Give the main reason why call-by-name parameter passing is not widely used and give a supporting example to explain the difficulty with call-by-name parameter passing. [4 marks]

Question 5 [12 marks]

Consider the context-free grammar

$$\begin{aligned} S &\rightarrow X S \mid d S \mid \epsilon \\ X &\rightarrow Y \mid Z b \mid a Y \\ Y &\rightarrow c Z \\ Z &\rightarrow e \end{aligned}$$

The symbols S, X, Y and Z are non-terminals with S as the start symbol while a, b, c, d, e are terminal symbols.

- Give the *FOLLOW* and *FIRST* sets for each non-terminal symbol. [5 marks]
- Construct the parsing table for a non-recursive predictive parser for this grammar. [4 marks]
- Is the grammar $LL(1)$? [1 mark]
- Detail how an non-recursive predictive parser will parse the sentence *dace* using the table you constructed above. [2 marks]

Question 6 [4 marks]

Consider again the context-free grammar from Question 5.

- Why is this grammar not directly suitable for implementing a recursive descent parser. Identify the productions that cause the problem? [2 marks]
- Modify the grammar (of course without changing the language it defines) such that it can be implemented directly with a recursive descent parser. [2 marks]

Question 7 [14 marks]

Consider the context-free grammar

$$\begin{aligned} S &\rightarrow a X \\ X &\rightarrow b X \mid b Y \\ Y &\rightarrow c \end{aligned}$$

The symbols S, X, Y are non-terminals and S is the start symbol while a, b and c are terminal symbols.

- Give the canonical collection of $LR(0)$ items for this grammar (remembering to first augment it with a new start symbol S'). [6 marks]
- Compute the *FOLLOW* sets for all non-terminals and give the SLR parsing table (action and goto) for this grammar. [4 marks]
- Detail how an SLR parser will parse the sentence *abbc* using the SLR table you constructed above. [4 marks]

***** END OF EXAM *****