Office Use Only

MONASH University

Semester Two 2018
Examination Period

Faculty of Information Technology

EXAM CODES:	**FIT2102**
TITLE OF PAPER:	**Programming Paradigms – PRACTICE EXAM**
EXAM DURATION:	2 hours writing time
READING TIME:	10 minutes

THIS PAPER IS FOR STUDENTS STUDYING AT: (tick where applicable)

□ Berwick	☑ Clayton	☑ Malaysia	□ Off Campus Learning	□ Open Learning
□ Caulfield	□ Gippsland	□ Peninsula	□ Monash Extension	□ Sth Africa
□ Parkville	□ Other (specify)			

AUTHORISED MATERIALS

OPEN BOOK	□ **YES**	☑ **NO**
CALCULATORS	□ **YES**	☑ **NO**
SPECIFICALLY PERMITTED ITEMS **if yes, items permitted are:**	□ **YES**	☑ **NO**

Candidates must complete this section if required to write answers within this paper

Section A: Multiple Choice Questions (40 marks total, 4 marks each)

NOTE: For the purpose of the practice exam I am not writing these as multiple choice because I want to prompt you to ensure you really understand the concepts rather than some bare minimum. These questions are of equivalent difficulty and cover the topics of Section A of the real exam.

Question 1. Compare and relate the von Neumann architecture, Turing machines and Lambda Calculus. What are they? What programming paradigms have they influenced?

Question 2. Precisely what data does a closure have access to and how long does it retain access to that data?

Question 3. When a variable is declared const in JavaScript is it mutable or immutable? Is the object that it references mutable or immutable?

Question 4. In lectures we constructed lists of closures in JavaScript (which we called "cons lists").

```
const cons = x=> y=> f=> f(x)(y)
```

What exactly are the symbols cons, x, y, and f in the above code? Use words like function, parameter, argument, closure, apply/applied/application and so on.

Question 5. What do the following terms mean and how do they relate to one another?

referential transparency

pure function

declarative programming

mutable variables

side effects

How can consideration of these ideas lead to more maintainable code?

Question 6. In Haskell what are the names and type definitions of the MINIMAL functions that must be specified for instances of each of the following type classes:

Functor, Applicative, Foldable, Traversable and Monad?

Question 7. What is the sequence of mathematical operations that are performed because of each of the following, and what is the final result of each?

i) `foldl (*) 1000 [10, 5, 1]`

ii) `foldr (*) 1000 [10, 5, 1]`

iii) `foldl (/) 1000 [10, 5, 1]`

iv) `foldr (/) 1000 [10, 5, 1]`

Question 8. What type is required for the function that needs to be placed at `_hole` in the following expression? Also give a definition for the expression using <$> and <*>.

```
sequence :: Applicative f => [f a] -> f [a]
sequence = foldr (_hole (:)) (pure [])
```

Question 9. Give examples and definitions for the following terms relating to lambda calculus:

Alpha equivalence

Beta normal form

Beta reduction

Eta conversion

Combinator

Question 10. In MiniZinc, use a forall expression to constrain the rows of an n by n matrix to sum to the same value. Also declare that the numbers in each row are all different – but to make it more interesting don't use the built-in alldifferent constraint.

Section B: Short Answer and Coding Questions (60 marks total)

Once again, the following short answer questions sometimes have several parts in order to cover topics in the exam without giving the actual question directly. Questions worth 4 marks on the real exam have only one part.

Question 11. (4 marks) The Math.pow function in javascript takes two arguments and raises the first to the power of the second, e.g.:

```
Math.pow(3,2)
> 9
```

Write a curried function that uses Math.pow, first with the function keyword, and then give the function again using arrow syntax. Include typescript type annotations. Show how you would use it to square the values in an array.

Question 12. (6 marks) Write a function that can take a function like the one you just wrote for Math.pow, and give back a binary version. Include type annotations.

Question 13. (8 marks total) This question is about **tail recursion** and is in three parts. All code must be self contained, do not use any functions from the Prelude or other libraries.

(i) **(2 marks)** Write a **non-tail recursive** Haskell function to compute the sum of a list of values with type:

```
sum :: [Integer] -> Integer
```

(ii) **(3 marks)** Now write the function again **using tail recursion**. It should still expose the same type definition:

```
sum :: [Integer] -> Integer
```

(iii) **(3 marks)** What is the benefit of tail recursion?

Question 14. (6 marks) Given the following Lambda Calculus expressions:

TRUE = λxy. x
FALSE = λxy. y

IF = λbtf. b t f

AND = λxy. IF x y FALSE
OR = λxy. IF x TRUE y
NOT = λx. IF x FALSE TRUE

Show how the Lambda Calculus can be used to evaluate:

OR FALSE TRUE

Give all of the Beta reduction steps in your evaluation using the notation for substitution described in the lectures, i.e. `[<variable> := <substitute expression>]`.

Question 15. (4 marks) Consider the following JavaScript definitions:

```
const cons = head => rest => f => f(head)(rest),
      head = list => list(head=> rest => head),
      rest = list => list(head=> rest => rest);

const aList = cons(1)(cons(2)(cons(3)(null)))

const forEach = f => l => {
    if (l) {
        f(head(l));
        forEach(f)(rest(l))
    }
}
```

implement the filter function such that the following code produces the output as indicated:

```
forEach(console.log)(filter(a=>a%2==1)(aList))
>1
>3
```

Question 16. (6 marks total) Consider the following TypeScript definitions:

```
interface LazySequence<T> {
  value: T;
  next(): LazySequence<T>;
}

function makeSequence<T>(getNext: (_:T)=>T, initial:T) {
  return function _next(v:T): LazySequence<T> {
    return {
      value: v,
      next: ()=>_next(getNext(v))
    }
  }(initial)
}
```

(i) What parameters would you give to makeSequence to create function which generates the following sequence:

1, -1, 2, -2, 3, -3…

(3 marks).

(ii) What parameters would you give to makeSequence to create function which generates the following sequence of arrays:

[1,2,3], [2,3,4], [3,4,5],…

(3 marks)

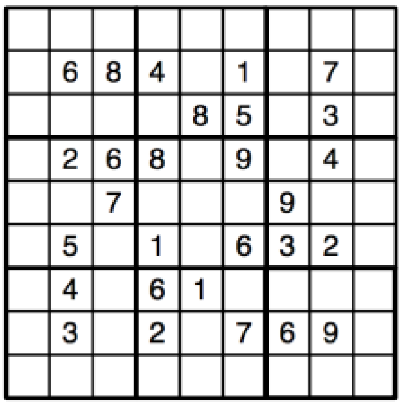
Question 17. (6 marks) Write the following Haskell functions in point-free form. On separate lines, show (and name) all of the reduction and conversion steps required to achieve the point-free form *(the actual exam has only one, but I thought I would give you extra practice).*

```
fun1 p l = part (<p) l

fun2 a b c = (a * b) + c

fun3 a b = a `div` (g b)
```

Question 18. (6 marks) Sudoku puzzles typically involve completing a 9 by 9 grid of digits (1..9) such that in each row, column and 3 by 3 subsquare each digit appears only once. For example:



	6	8	4		1		7	
				8	5		3	
	2	6	8		9		4	
		7				9		
	5		1		6	3	2	
	4		6	1				
	3		2		7	6	9	

In the MiniZinc code below which models and solves the above puzzle, complete the constraints for columns and sub-squares.

```
start=[|
  0, 0, 0, 0, 0, 0, 0, 0, 0|
  0, 6, 8, 4, 0, 1, 0, 7, 0|
  0, 0, 0, 0, 8, 5, 0, 3, 0|
  0, 2, 6, 8, 0, 9, 0, 4, 0|
  0, 0, 7, 0, 0, 0, 9, 0, 0|
  0, 5, 0, 1, 0, 6, 3, 2, 0|
  0, 4, 0, 6, 1, 0, 0, 0, 0|
  0, 3, 0, 2, 0, 7, 6, 9, 0|
  0, 0, 0, 0, 0, 0, 0, 0, 0|]

int: S = 3;
int: N = S * S;

set of int: PuzzleRange = 1..N;
set of int: SubSquareRange = 1..S;

array[1..N,1..N] of 0..N: start; %% initial board 0 = empty
array[1..N,1..N] of var PuzzleRange: puzzle;

% fill initial board
constraint forall(i,j in PuzzleRange)(
  if start[i,j] > 0 then puzzle[i,j] = start[i,j] else true endif );

% All different in rows
constraint forall (i in PuzzleRange) (
  alldifferent( [ puzzle[i,j] | j in PuzzleRange ]) );

% All different in columns.

% All different in sub-squares:
solve satisfy;
```

Question 19. (6 marks) How can Haskell ADTs be used as the product of a parser? How do they relate to the grammar of a language? How does such an ADT relate to an ADT? Refer to the examples discussed in lectures and tutes in your answer to the above.

Question 20. (8 marks total)

(i) **(4 marks)** A Rose-tree can store a value, and a list of sub-trees in its nodes – or it can be Nil. Give a generic algebraic data type for such a rose tree that also includes terms in its internal nodes for the minimum and maximum values in any of its subtrees.

(ii) **(4 marks)** Consider a binary search tree with ordered keys and a value associated with each key. Describe in words and diagrams how a pure function can change the value (not the key) stored at one of the leaves in better than linear time and memory.

END OF EXAMINATION