Department of Computer Science
University College London

# Cover Sheet for Examination Paper to be sat in May 2005

# COMP3C11: Functional Programming

Time allowed 2.5 hours

Calculators are allowed

Answer THREE questions

| | |
|---|---|
| Checked by First Examiner: | Date: |
| Approved by External Examiner: | Date: |

COMP3C11: Functional Programming, 2005

**Answer any THREE questions**

Marks for each part of each question are indicated in square brackets.

Calculators are permitted

1. (a) Give the syntax for the untyped lambda calculus without constants and give the definitions for the reduction rules that are used to evaluate expressions in that calculus. [10 marks]

   (b) Explain how a lambda calculus expression is evaluated, including an explanation of the term “reducible expression”. [11 marks]

   (c) Sometimes a function may appear to be applied to too many arguments. Explain how this can occur without any error – i.e. where the function and its application are correctly typed and there is no compile-time or run-time error. Give an example to illustrate your answer. [6 marks]

   (d) What is a recursive type? Give an example of a built-in recursive type in Miranda. How can you define your own recursive type? Give an example in Miranda of a user-defined type which is recursive. [6 marks]

[Total 33 Marks]

2. (a) Provide an algebraic data type definition to represent the four suits in a deck of cards. Make use of this algebraic type in a type synonym to represent the values that a particular card might take, for example the ten of hearts.
[6 marks]

(b) Write a function that will take as arguments (i) a number and (ii) a list of elements. Each element in the list represents a card (using the types defined in (a)). The function should produce as its output a "shuffled" version of the input list. The function should shuffle the list of cards as many times as is indicated by the first argument.

The action of shuffling should cut the deck in half and then interleave the cards, with the previous top card now being the second card in the pack. For example, if a list of four items A, B, C and D is shuffled once the result should be C, A, D, B. If that result is shuffled again the result should be D, C, B, A. As a simplification, you may assume that where there is an odd number of items in the list, the last item is discarded.

Your function should detect the case where there are more than 52 elements in the input list, which it should treat as an error. [18 marks]

(c) Write a function which takes two integer numbers and generates a result using the following rules:

1. Multiply the two numbers together
2. Then add all the digits of the result
3. If the sum of the digits has itself only one digit then return it as the result of the function, otherwise repeat from 2.

For example, if your function is called "f", a sample interaction with the Miranda system would be:

```
Miranda f 3 4
3
Miranda f 7 7
4
Miranda f 30 19
3
```

The last of the above examples is calculated as follows:

*30 * 19 is 570*
*5 + 7 + 0 is 12 (which has 2 digits)*
*1 + 2 is 3 (which has 1 digit)*
*the result is 3*

[9 marks]

[Total 33 marks]

3. (a) Provide definitions, including types, for the functions (from the Miranda Standard Environment) called **foldr** and **foldl**. [12 marks]

   (b) What values do the following five expressions compute?

```
foldr (*) 1  [1,2,3]

foldr (:) [] [1,2,3]

hd (foldr (:) [] [1..])

foldl (swap (:)) [] [1,2,3]

hd (foldl (swap (:)) [] [1..])
```

[8 marks]

   (c) How do the functions **foldr** and **foldl** differ in their ability to process infinite lists? [5 marks]

   (d) What does the following function do and what is its type?

```
mystery op = g
             where
             g r = (r:). rest
                   where
                   rest []    = []

                   rest (a:x) = g (op r a) x
```

[8 marks]

[Total 33 marks]

4. You are told that a novel memory allocator preferentially allocates large blocks at the top end of available memory and small blocks at the bottom end of available memory. In all other respects the allocator is conventional. The allocator uses a single free list in a heap memory of size ***H*** bytes (see Figure 1).

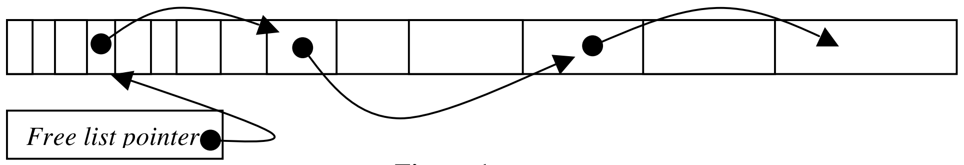

**Figure 1**

(a) How many blocks of memory (and of what size) are there on the free list when the program starts (before any allocation)? [5 marks]

(b) Give the details of how the very first allocation could be achieved if the requested block size is (i) small, and (ii) large. Give a diagram to illustrate the heap and the free list both before and after allocation in each case. [9 marks]

(c) If the free list is double-linked (see Figure 2), and each block's header contains its size, what is the size of a block header and what is the minimum size of an allocatable block? [5 marks]

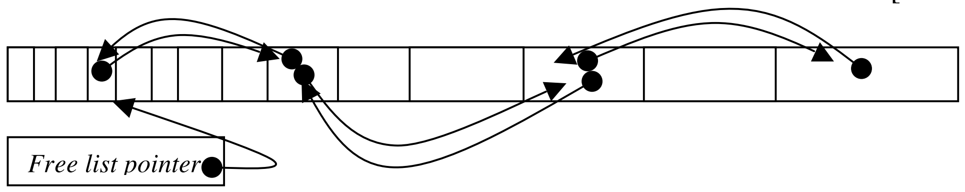

**Figure 2**

(d) You are told that the allocator maintains two pointers – one to the start of the double-linked free list and one to the end of the double-linked free list. Give an allocation algorithm using these two pointers which would, for multiple successive block allocation requests, place small blocks at the bottom end and large blocks at the top end of available memory. State any assumptions you make. [7 marks]

(e) Give a sequence of allocation requests and requests to free previously-allocated blocks that could defeat the intentions of this allocator, thereby causing small and large blocks to be intermingled. [7 marks]

[Total 33 Marks]

5. Given a heap memory with the following characteristics:

   - heap size = 2Mbytes
   - block header size = 4 bytes
   - minimum allocatable block size = 128 bytes
   - maximum allocatable block size = 16Kbytes
   - free-list allocator using LIFO first-fit allocation

   (a) What is the maximum number of live blocks that can exist in this heap? [2 marks]

   (b) What is the minimum information that needs to be stored in each block header and what extra information would each header need to include in order to support coalescing? [4 marks]

   (c) How would the performance of the allocator be affected if address-ordered first-fit allocation were used instead of LIFO first-fit allocation? [2 marks]

   (d) What problem is solved by Knuth's boundary tags mechanism? Give a detailed explanation of how boundary tags could be implemented in the above heap to provide a comprehensive solution to the stated problem. Specific attention should be paid to the contents of the block headers, the procedures involved, and the performance implications. [15 marks]

   (e) If the maximum block size were to be increased to 127Kbytes, how would this affect the block header (assume coalescing is supported)? Suggest two solutions and state the factors that would determine which solution would be preferable in a given context. [10 marks]

[Total 33 marks]

END OF PAPER