Department of Computer Science
University College London

Cover Sheet for Examination Paper to be sat in May 2012

COMPGC16: Functional Programming

Time allowed 2.5 hours

Calculators are allowed

Answer THREE questions

Checked by First Examiner: Date:

Approved by External Examiner: Date:

COMPGC16: Functional Programming, 2012

Answer any THREE questions
Marks for each part of each question are indicated in square brackets.
Calculators are permitted

1. (a) What is a Curried function? Explain how a Curried function can be partially applied. Give an example to illustrate your answer. [6 marks]

   (b) What is pattern-matching in the context of a function definition? What are the constructs that may appear as patterns in a Miranda or Amanda function definition? (State whether you are describing Miranda or Amanda patterns.) [6 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 in Miranda or Amanda to illustrate your answer. [7 marks]

   (d) What is a recursive function? Give examples of function definitions in Miranda (or Amanda) for both a stack recursive function and an accumulative recursive function. For each function, provide a sample function application and give the reduction steps (using any reduction order, to any normal form) for that application. [7 marks]

   (e) What is a recursive type? Give an example of a built-in recursive type in Miranda or Amanda. How can you define your own recursive type? Give an example in Miranda or Amanda of a user-defined type that is recursive. [7 marks]

[Total 33 marks]

2. The Miranda functions **rcons, rnil** and **id** are defined below, together with the definition of a function **rev** that reverses a list:

```
rcons a f b = f (a : b)

rnil = id

id x = x

rev items = foldr rcons rnil items []
```

(a) Give the Miranda type definitions for **rev** and **rcons**. [10 marks]

(b) Provide a hand evaluation, giving all intermediate evaluation steps, for the following application of **rev**:

```
rev [1,2,3]
```

[9 marks]

(c) Will **rev** correctly reverse any list? Explain your answer. [4 marks]

(d) Explain in detail how **rcons** and **rnil** work, with examples. [10 marks]

[Total 33 marks]

3. (a) Provide an **algebraic 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]

4. Consider the following λ-calculus expression:

```
(λy.(( λf.( λx.(f x))) (λg.(g y))) 37) (if True (+ 1) (+ (37 / 0)))
```

(a) In the sub-expression `λg.(g y)` explain which identifiers are free and which are bound. [4 marks]

(b) Explain the expected operation of the sub-expression `λg.(g y)` when applied to an argument. Give an example application of this sub-expression (assume that `y` = 37) and show the evaluation steps of your example. [6 marks]

(c) Give the evaluation steps for the above expression using a different reduction order to that used in your answer to part (b), and use your example to demonstrate how evaluation could produce a different outcome. Is this the only different outcome that can occur? Why? [12 marks]

(d) The above λ-calculus expression is very slightly altered by changing the bound names as follows:

```
(λa.((λf.(λa.(f a))) (λf.(f a))) 37) (if True (+ 1) (+ (37 / 0)))
```

(i) First give the evaluation steps to perform a simple beta-reduction of the following sub-expression: `λf.( λa.(f a)) (λf.(f a))`

(ii) In the full expression, replace the above sub-expression with its beta-reduced result. Then give the evaluation steps for any remaining redexes (in Normal Order). When an error occurs, explain what the error is and why it occurred.

Hint: in the above sub-expression, which identifiers (names) are free before the simple beta-reduction, and which (if any) are free after the simple beta-reduction? If a free identifier has become bound, then simple beta-reduction has failed - but what has caused the simple beta-reduction to behave like this?

[11 marks]

[Total 33 marks]

5. (a) Explain how a “free list” memory allocation algorithm operates. Give three different examples of sequential fit algorithms and compare their behaviour.

[16 marks]

(b) Give a detailed description of the contents of a cell in a graph reduction system that uses binary application cells, and explain how the cell could be augmented to accommodate a reference-counting garbage collector.

[8 marks]

(c) Give three limitations of reference-counting garbage collection, in terms of its functionality and its performance, and suggest possible solutions for two of these limitations.

[9 marks]

[Total 33 marks]

[END OF PAPER]