# Homework 4 Function application Answers

1. The function multiply3 is defined as follows:

multiply3 :: Integer 🡪 Integer 🡪 Integer 🡪 Integer

multiply3 x y z = x \* y \* z

(a) What result is returned by the following statement?

multiply3 2 4 6 [1]

48

(b) A further function multBySeven is defined as follows:

multBySeven x y = multiply3 7

(i) What result is returned by the following statement?

multBySeven 5 3 [1]

105

(ii) What is partial function application? Explain, in terms of partial function   
application, how this result is arrived at. [3]

Partial function application involves passing less than the full number of arguments (1) to a function, by binding one value (1), in this case 7. multiply3 actually takes one argument and returns another function which takes the next argument (1) until an integer is returned (1). This can be written as *multiply3: integer 🡪 (integer 🡪 (integer 🡪 integer))* (1) Max 3

2. (a) What is a higher-order function? [2]

A function is higher-order if it takes a function as an argument or returns a function as a result, or does both.

(b) Explain what the **map** function does in a functional programming language such as Haskell. [2]

**Map** is a higher-order function that takes a list and the function to be applied to the elements in the list as inputs, and returns a list made by applying the function to each element of the old list.

(b) Use map to write a function that doubles each element of the list [6, 9, 22, 103] [2]

map (\*2) [6, 9, 22, 103]

(nb It is not necessary to be too fussy about the correct syntax in these questions,   
so long as the general format is clear of an operation being performed on a list

3. Write statements that will define a list called listA containg several integers, and produce a list containing only the negative numbers from listA.  
 [3]

listA = [-1,0,1] – - for example

filter (<0) listA

4. Haskell uses a fold function.

(a) Explain what a fold function does in a functional programming language   
such as Haskell. [1]

A fold function reduces a list to a single value, using recursion.

[2]

(b) Explain the difference between foldl and foldr [1]

foldl stands for fold left, i.e. the recursion starts with the leftmost value. foldr or fold right starts with the rightmost value

(c) Use a fold function to return the sum of all the numbers in a list, listA [2]

foldl (+) 0 listA

(or foldr (+) 0 listA)

(Accept statements that have minor syntax errors)

5 Write code to return names from listB = [“Albert”, “Victoria” , “Caroline”, “Maximilian”, “Frederick”, “Henry”] which are before “George” alphabetically [2]

filter (< "George") listB

[Total 20 Marks]