G6021: Comparative Programming

Exam Practice 2

1. (a) Give an example for each of the following:
   i. a λ-term that is in normal form but does not have a type.
   ii. a λ-term that has a normal form but does not have a type.
   iii. a λ-term that has a normal form and does have a type.

   (b) Give the β-reduction graph of the λ-term $(\lambda xy.x)(II)I$, where $I = \lambda x.x$.

2. In the Rock-Paper-Scissors game, the rules are that *paper* beats *rock*, *rock* beats *scissors*, and *scissors* beats *paper*.

   (a) Define a type `RPS` to represent rock, paper and scissors as used in the game in both Haskell and Java. In no more that 10 sentences, compare the two.

   (b) Write a function in Haskell: `beats :: RPS -> RPS -> Bool` that encodes that paper beats rock, rock beats scissors, and scissors beats paper.

   (c) Suppose you wanted to write a function/method to return two values (say a pair of integers). Outline how you would do this in both Haskell and Java.

3. (a) Write two recursive functions in Haskell `sumAll` that returns the sum of all the elements of a list of numbers and `prodAll` that returns the product of all the elements in a list of numbers.

   (b) Write in Haskell the higher-order function `foldr` which takes three parameters: a function, a value and a list. Include the type of the function `foldr` in your answer, and give two examples of use of `foldr` to compute the sum and product of a list of numbers (as in the previous question).

   (c) Using the following function as an example

   ```
   g (x,y) = x+y
   ```

   Explain the concept of currying a function. Curry the function `g` and give the type of the resulting function.