School of Computing, Napier University

Assessment Brief

1. Module number	*SET07106*
2. Module title	*Mathematics for Software Engineering*
3. Module leader	*Peter Chapman*
4. Tutor with responsibility for this Assessment Student's first point of contact	*Peter Chapman*
5. Assessment	*Assessment 1 – Haskell*
6. Weighting	*33%*
7. Size and/or time limits for assessment	*Answer all 5 questions*
8. Deadline of submission	*23:55 Friday 8th April*
9. Arrangements for submission	*Submit Haskell file (only) to Moodle.*
10. Assessment Regulations All assessments are subject to the University Regulations**.**	*You are reminded that you must work on your own.*
11. The requirements for the assessment	*See following pages*
12. Special instructions	*See following pages*
13. Return of work	*You will receive feedback via Moodle.*
14. Assessment criteria	*See following pages*

SET07106 - Haskell Coursework

General Remarks

In this coursework you will be asked to create functions which solve specific problems. Some of these problems will be variations of problems you have seen in the practical exercises, and some will be derived from material we have covered in the lectures. You will be given the type-signatures, and order of the arguments, of the functions in the associated `cwk.hs` file.

> **DO NOT CHANGE THESE TYPE SIGNATURES OR THE ORDER IN WHICH ARGUMENTS APPEAR**

The coursework will be marked automatically, and if you change the type signatures or order of the arguments then you will make it impossible for you to get the answers correct. Your code will not be checked, which gives you freedom to use whichever method you want to create your functions (i.e. list comprehension, using `map`, using induction, if done correctly, will give the same results, and hence will gain the same marks.)

The marks for each function are contained in this document. Each function will be tested on 5 different inputs, and your mark displayed is the mark each *test* carries. The total number of marks for the coursework is then 100.

Instructions

- Download the file `cwk.hs`, and change the file name to `YOURSTUDENTNUMBER.hs`. For example, if your student number was 40101916, then your file would become `40101916.hs`.
- Uncomment each type signature and write the functions as intended. You may add as many extra functions to your file as you wish. However, `import` statements are not allowed.
- Upload your file to the submission point on Moodle by **2355 on Friday 8th April 2016**.
- To allow you to judge the effectiveness of your functions, in the file `cwk.hs` are some Test Sets, which detail the correct behaviour of the functions for certain inputs.

Questions

Question 1 - Relations

Write a series of functions which, given an input of a list of pairs, determines whether the given list represents:

- a reflexive relation, (`isReflexive`) (**1 mark**)
- a symmetric relation, (`isSymmetric`) (**1 mark**)
- a transitive relation, (`isTransitive`) (**2 marks**)

- an equivalence relation (`isEquivalence`). (**1 mark**)

Write a function which takes an equivalence relation and an element of the carrier set and returns a list of elements which form the equivalence class of the given element. (`eqClassOf` - **2 marks**). Note that this will only be tested on equivalence relations: *you can assume that the given input will be an equivalence relation.*

Question 2 - Multisets

Multisets are sets which allow multiple copies of the same element, but are unordered. In list form, for example, `[1,1,2,3]` is equal to `[1,2,3,1]`, but not to `[1,2,3]`. Write functions that calculate:

- whether two multisets are equal, (`multiEqual`) (**1 mark**)
- the union of two multisets, (`multiUnion`) (**2 marks**)
- the intersection of two multisets. (`multiIntersection`) (**2 marks**)

Question 3 - Matrices

Write a function `trace` which returns the trace of a square matrix. Note, however, that you should not assume that the input will always be a square matrix. (**1 mark**)

Write a function `matMult3` that multiplies together two 3×3 matrices. (**2 marks**)

Question 4 - Combinatorics

Write a function, `triNumbers`, which takes an integer inputs n and k, and returns the n-th row in the triangle generated via the following process, where $t_{i,j}$ is the i-th entry on the j-th row of the triangle:

1. $t_{1,1} = k$,
2. $t_{1,n} = t_{n-1,n-1}$, and
3. for $1 < i \leq n$, $t_{i,n} = t_{i-1,n} + t_{i-1,n-1}$.

You are strongly advised to draw these triangles in help you understand the problem. (**3 marks**)

Question 5 - Number theory

Write a function `combine` which given two integers n and m, returns a triple of integers $(a, b, \gcd(n, m))$ such that:

$$am + bn = \gcd(n, m)$$

You should not assume that the integers will always be positive. (**2 marks**)