The University of British Columbia
Department of Computer Science
Midterm Examination 2 — Fall 2019

Computer Science 312
Functional and Logic Programming

Question 1 [8 marks]

The function `flip` is defined in the Prelude as

```
flip f x y = f y x
```

Suppose the function `mtr` is defined as

```
mtr a b c = 100*a + 10*b +c
```

so that `mtr 2 4 6` returns `246`.

(a) [2 marks] Give the inferred type of `flip`

(b) For each of the following, give the output when it is input it to ghci. If there is an error, say why it occurred. [You do not need to show your reasoning if you get the correct answer, but if you want partial marks, you need to explain your answer.]

i) [2 marks] `mtr (flip 2 4) 6`

ii) [2 marks] `flip mtr 2 4 6`

iii) [2 marks] `flip (mtr 2) 4 6`

Question 2 [12 marks]

This function is supposed to regularize a list (reducing the effect of outliers, by averaging each pair of elements in the list, so e.g., $[1, 3, 9, 9]$ is regularized to $[2, 6, 9]$):

```
regularize :: [t] -> [t]
regularize [] = []
regularize [a,b] = [(a+b)/2]
regularize a:b:r = (a+b)/2 : regularize b:r
```

This was put into the file `regularize.hs`. As it is, the function has errors. The line number and the first line of the error message are given below. When answering each part, assume the bugs in previous parts have been fixed. For each case, briefly explain why the error occurs and make a fix on the code above.

(a) [3 marks]

```
regularize.hs:4:1: error: Parse error in pattern: regularize
4 | regularize a:b:r = (a+b)/2 : regularize b:r
  | ^^^^^^^^^^^^
```

Why error occurred:
Show fix on the code above.

(b) [3 marks]

```
regularize.hs:4:43: error:
    • Couldn't match expected type '[t0]' with actual type 't'
```

Why error occurred:
Show fix on the code above.

(c) [3 marks]

```
regularize.hs:3:21: error:
    • No instance for (Fractional t) arising from a use of '/'
```

Why error occurred:
Show fix on the code above.

(d) [3 marks] What other error can arise in the use of this function? How can this be prevented?

Question 3 [10 marks]

For this question you may use the following Haskell functions:
`id x = x`
`:` is defined so that `h:t` is the list with first element `h` and rest of the list `t`
`[]` is the empty list
`+` is arithmetic addition
`++` is list append

Consider the type `Tree`, the value `mytree` and the function `foo` defined in Haskell as follows:

```
data Tree t = Leaf t | Node (Tree t) (Tree t)
mytree = Node (Leaf 1) (Node (Leaf 3) (Node (Leaf 5) (Leaf 2)))

foo f v (Leaf l) = f l v
foo f v (Node t1 t2) =  foo f  (foo f v t2) t1
```

(a) [3 marks] What is the inferred type of `foo`?

(b) [3 marks] Give the result of `foo (+) 0 mytree`
(You do not need to show your reasoning if you have the correct answer, but if you want partial marks you need to show your reasoning).

(c) [4 marks] Give a query that uses `foo` to return a list of the values at the leaves of mytree (the order does not matter). You may not use recursion and you may only use functions specified above.

Question 4 [12 marks]

A binary search tree is a useful definition of a set. One way to keep a tree (approximately, on average) balanced is to hash the values, and to keep the tree sorted on the hash values.

Suppose the class `Hashable t` contains the types that implement the function `hash`, with type:

```
hash :: Hashable a => a -> Int
```

Consider the following Haskell code:

```
data Set t = Empty
           | SNode t (Set t) (Set t)

member _ Empty = False
member v (SNode e lt ut)
    | v==e = True
```

```
  | hash v <= hash e = member v lt
  | otherwise = member v ut
```

(a) [2 marks] What does `t` represent in the data declaration of Set?

(b) [4 marks] The inferred type of `member` is

```
member :: (Eq t, Hashable t) => t -> Set t -> Bool
```

Explain why Haskell infers these class restrictions.

(c) [6 marks] Implement the function `insert e s` that returns the set containing `e` and the elements of set `s`. It should not have duplicate elements. You must include a type declaration for `insert`.