Haskell Introduction (1A)

Young Won Lim
6/30/16

Please send corrections (or suggestions) to youngwlim@hotmail.com.

This document was produced by using OpenOffice.

Based on

Haskell Tutorial by Medak & Navratil
ftp://ftp.geoinfo.tuwien.ac.at/navratil/HaskellTutorial.pdf

Function

```
increment :: Int -> Int
```

function name arg type return type

Int increment (Int)

```
increment    x = x + 1
```

Int arg Int return

```
increment    2
```

```
3 :: Int
```

Function Examples

```
and1 :: Bool -> Bool -> Bool
```

function name arg1 type arg2 type return type

Bool and1 (Bool, Bool)

```
and1 a b = if a == b then a
           else False
```

Bool arg1, arg2 Bool return

Function Examples

```
and1 :: Bool -> Bool -> Bool
and1 True True = True
and1 x    y    = False
```

⬅ Rule (I)

⬅ Rule (II)

```
and1 :: Bool -> Bool -> Bool
and1 True True = True
and1 _    _    = False
```

⬅ Rule (I)

⬅ Rule (II)

wild character

Where Statement – Local Definition

```
roots :: (Float, Float, Float) -> (Float, Float)
```

Input tuple

Output tuple

```
roots :: (a, b, c) = (x1, x2) where
   x1 = e + sqrt d / (2*a)
   x2 = e – sqrt d / (2*a)
   d = b*b – 4*a*c
   e = – b / (2*a)
```

```
p1, p2 :: (Float, Float, Float)
p1 = (1.0, 2.0, 1.0)
p2 = (1.0, 1.0, 1.0)
```

Variable declaration

Variable assignment

Variable assignment

Function Call Results

```
? roots p1 (-1.0,-1.0) :: (Float, Float)
(94 reductions, 159 cells)

? roots p2
( Program error: {primSqrtFloat (-3.0)}
(61 reductions,183 cells)
```

If-then-else Statement

```
roots :: (Float, Float, Float) -> (Float, Float)
```

Input tuple

Output tuple

```
roots :: (a, b, c) = (x1, x2) do
    if d < 0
       then error “sorry”
       else (x1, x2)
  where
    x1 = e + sqrt d / (2*a)
    x2 = e – sqrt d / (2*a)
    d = b*b – 4*a*c
    e = – b / (2*a)
```

Indentation

References

[1] ftp://ftp.geoinfo.tuwien.ac.at/navratil/HaskellTutorial.pdf