classmate
Date 29.8.2019
Page 1

HASKELL 3

parameterized types

useful haskell types

maybe a

either a b

functors

burritos

data GenericTree a = Empty | Node a GenericTree GenericTree

data Tree a = Empty | Node a (Tree a) (Tree a)

:t [] :k Tree (kind)

[] :: [a] Tree :: * -> *.

Hoogle maybe (float)

```
div :: Float -> Float -> Maybe (Float)
div a b = if b /= 0 then Just (a / b) else Nothing.
```

```
div :: Float -> Float -> Either Float String
div a b = if b /= 0 then Left (a / b) else Right "by zero"
```

```
f :: Tree -> Int
f Empty = -1
f (Node a _ _) = a
```

```
f :: Tree -> Either Int Tree
f Empty = Right Empty
f (Node a _ _) = Left a
```

```
f :: Tree -> Maybe Int
f Empty = Nothing
f (Node a _ _) = Just a.
```

```
getKey tree = case (getKey' tree) of
    Nothing -> -1
    Just x -> x
```

Quiz-1

① Define scalar-product

```
(lambda (n lst)
    (map (lambda (a) (* a n)) lst )))
```

② Define .*

```
(lambda (l1 l2)
```

✓learn ✓scheme ✓parameter ✓pop ✓recursive ✓list ✓length.