ASGS1115 - Support for Science

Semester 1 - 2019

LAB week 5: Recursion

Function Syntax Review

- Pattern Matching
- Guards
- Cases
- If statement
- List Comprehension

Fibonacci Recursion

Load Haskell script Fibonacci.hs which contains the following Haskell function:

```haskell
-- returns the n-th fibonacci number
fibonacci :: (Integral x) => x -> x
fibonacci 0 = 0
fibonacci 1 = 1
fibonacci x = (fibonacci (x-1)) + (fibonacci (x-2))
```

Take a look at how Fibonacci sequence works and draw or imagine a Fibonacci tree (see resource) as you input a number into the `fibonacci` function.

List Recursion

Load Haskell script ListRecursion.hs, which contains the functions below:

```haskell
-- returns reversed inputted list
revl :: [x] -> [x]
revl [] = []
revl (x:xs) = (revl xs) ++ [x]

-- returns the sum of elements collected sequentially from
  ↪ beginning to end of [x] up to value of y
pickseq :: [Int] -> Int -> Int
pickseq [] y = 0
pickseq x 0 = 0
pickseq [x] y
      | x>y    = 0
      | x<=y   = x
pickseq (x:xs) y
      | x>y    = pickseq xs y
      | x<=y   = x + (pickseq xs (y-x))

-- returns a sorted list
sortasc :: (Integral a) => [a] -> [a]
sortasc [] = []
sortasc [x] = [x]
sortasc (x:xs) = (sortasc [e | e<-xs, e<x]) ++ [x] ++ (sortasc [e | e<-xs, e>=x])
```

Try inputting list of numbers to each of the function and understand the recursion happened in each of them.

Challenges:

- Rewrite the function `sortasc` as a `sortdesc`, which takes a list of numbers and return a list sorted in descending order.

- Write a recursive function `sumnum` that takes a list of numbers `[a]` as argument and a number `y`, and sums all numbers in `[a]` that is less than `y`.

  test cases:
  `sumnum [3,2,1,5] 3` should return `3`
  `sumnum [6,2,2,2] 4` should return `6`

- Write a function similar to `pickseq` called `maxSum` which takes two arguments: a list of numbers `[a]` and a number `y`, and returns the maximum sum of elements of list `[a]` that is $\leq$ `y`.

  test cases:
  `maxSum [2,5,3,6] 4` should return `3`.
  `maxSum [5,7,8,9] 5` should return `5`.
  `maxSum [8,17,19] 7` should return `0`.