G6021: Comparative Programming

Exercise Sheet 6

1 Higher-Order functions on lists

1. Write a function `sumAll` that returns the sum of all the elements of a list of numbers.
2. Write a function `multAll` that returns the product of all the elements of a list of numbers.
3. The pattern of recursion should be the same in the previous two questions: the only differences are the name of the function, the function applied to each of the elements, and the starting value. Haskell provides a function to capture this kind of recursion: `foldr f b l`, where the three parameters are the function to be applied, the starting value and the list.

   Examples of this functions in use include:

   ```
   foldr (*) 1 [1,2,3,4]
   foldr (+) 0 [1,2,3,4]
   ```

   Test these functions, and check that they give the same answers as your functions above.
4. Write your own version of the `foldr` function (call this function `fold`).
5. Using `fold`, write the following:

   - A function `len` to compute the length of a list.
   - A function `maxElem` to compute the maximum element of a list.
   - A function `flatten` to convert a list of lists to a list. Example: `flatten [[1,2,3],[4,5,6]]` should give `[1,2,3,4,5,6]`.
6. What is the type of the function `iter` below, and what do you think it does? What does the function `f` compute?

   ```
   iter p f x = if (p x) then x else iter p f (f x)

   f n = snd(iter (\(x,y) -> x>n) (\(x,y) -> (x+1,x*y)) (1,1))
   ```

Assignment Project Exam Help
https://powcoder.com
Add WeChat powcoder

2 Accumulating parameters

Test out the three versions of factorial from the notes: `fact` and `factcps` and `factacc`.

1. Write a version of the `len` function using an accumulating parameter (so that the function is tail recursive):

   ```
   len [] = 0
   len (h:t) = 1+len t
   ```

2. Write a version of the `rev` function using an accumulating parameter:

   ```
   rev [] = []
   rev (h:t) = (rev t) ++ [h]
   ```

3 Data types

1. Give a datatype, called `IntOrBool`, that can represent either an `Int` or a `Boolean`. Show how you can use this datatype to represent a list of mixed elements (Integers and Booleans).
2. Suggest a way to represent the λ-calculus as a data type in Haskell.

Assignment Project Exam Help

https://powcoder.com

Add WeChat powcoder