CPSC 449: Exercise 1

Winter 2021 (Revised January 4, 2021)

Due: Friday, January 22 (2021) at 11:59 PM midnight

For this exercise you are expected to develop *at least 6* of the programs below. The tutorials will work through (some of) the solutions. You are expected to hand in a documented Haskell script containing your solutions ... I encourage you to discuss these programs with your classmates: the aim is to complete these exercises (somehow!) so that you get used to thinking in Haskell syntax. You should comment your code indicating, in particular, from where you got the solution if it is not your own. Recall that it is important that you understand the solutions as this comprehension will be tested in the tutorial written tests and your ability to program will be tested by the assignment.

Please name your functions according to what is prescribed below. If there are name conflicts with names defined in **`Prelude`**, then (a) explicitly import **`Prelude`**, and (b) use the **`hiding`** clause to hide the conflicting names when importing (see the grey box on page 53 of **[Thompson]**).

1. Write a function

   ```
   avgThree:: Int -> Int -> Int -> Float
   ```

   to take the average of three integers which returns a float.

2. Write a function

   ```
   maxThree:: Int -> Int -> Int -> (Int,Int)
   ```

   which works out the maximum of three integers and returns also how many times that maximum occurs.

3. Write a function

   ```
   data SF a = SS a | FF

   invFac:: Integer -> SF Integer
   ```

   which returns the largest number whose factorial is no greater than the given number (what happens if the given number is negative?).

4. Implement a function **`myGcd`** that takes two integers as arguments, and returns the greatest common divisor using the ***Euclid's Algorithm***.

```
myGcd :: Int -> Int -> Int
```

You may not assume that the arguments are both positive the greatest divisor does not depend on the sign of the integers! (However, what is the greatest divisor of 0 and 0?).

5. The binomial coefficient $\binom{n}{k}$ is defined as follows for integers $n \geq 1$ and $0 \leq k \leq n$:

$$\binom{n}{k} = \frac{n \times (n-1) \times \ldots \times (n-k+1)}{(1 \times 2 \times \ldots \times k)}$$

   Write a function:

```
binom:: Integer -> Integer -> Integer
```

   to calculate the binomial coefficients.

6. Write a function

```
grow :: String -> String
```

   which changes a string $a_1a_2a_3$... to $a_1a_2a_2a_3a_3a_3$... so `grow "now!" == "noowww!!!!"`.

7. Write a function

```
instrictorder:: [Int]-> Bool
```

   which tests whether the list of integers is strictly increasing.

8. Write a function

```
cheapItems:: [(String,Int)] -> Int -> [String]
```

   which given a list of items and their cost (here just an integer) returns a list of items whose cost is (strictly) below a given threshold cost.

9. Write a function

```
sortByCost :: [(String,Int)] -> [(String,Int)]
```

   which, given a list of items with a cost, returns a list in cheapest first order.

10. Write a function

```
divisors:: Integer -> [Integer]
```

   which calculates the list (in ascending order) of all prime divisors of a positive integer (returning the empty list if the number is less than or equal to one).

11. Defined function

    ```
    substring :: String -> String -> Bool
    ```

    which determines whether a given first string is a substring of a second string.

12. Write a function

    ```
    sublists:: [a] -> [[a]]
    ```

    which given a list of *any* type returns the list of all sublists of that list.