FP Lab 3

1. In Haskell, `String` is a synonym for `[Char]`. Test whether the following two lists are the same:

   ```
   ['a','b','b'] :: String
   "abb" :: String
   ```

2. (a) Define a function `allEven :: [Int] -> Bool` that tests whether the elements of the input list are all even numbers.

   (b) Using a list comprehension to define a function `addp :: [(Int,Int)] -> [Int]` that adds each pairs of integers together. For instance:

   ```
   > addp [(1,2), (4,3), (2,7)]
   [3,7,9]
   ```

   (c) Define `addpp :: [(Int,Int)] -> [Int]` which is the same as `addp` above except that the sum of `(m,n)` only appears in the result list if `m<n`. For instance,

   ```
   > addpp [(1,2), (4,3), (2,7)]
   [3,9]
   ```

3. A triple (x, y, z) of positive integers is called *pythagorean* if $x^2 + y^2 = z^2$. Using a list comprehension, define a function

   ```
   pyths :: Int -> [(Int,Int,Int)]
   ```

   that maps an integer `n` to all pythagorean triples (a_1, a_2, a_3) with components a_i in `[1..n]`. For example:

   ```
   > pyths 5
   [(3,4,5),(4,3,5)]

   > pyths 10
   [(3,4,5)], (4,3,5), (6,8,10), (8,6,10)]
   ```

4. A positive integer is *perfect* if it equals the sum of all of its factors, excluding the number itself. Using a list comprehension, define a function `perfects :: Int -> [Int]` that, given the input integer `n`, returns the list of all perfect numbers up to `n`. For example:

```
> perfects 500
[6,28,496]
```

(Hint: Use the library function `init` and the function `factors` as defined in the lecture slides VI.)