IN3043 Functional Programming
Exercises 2

1. Write a function `threeDifferent` that takes three integer arguments and returns `True` if its arguments are all different from each other.

2. Define a floating point constant for the "golden ratio", defined as $\varphi = \frac{1+\sqrt{5}}{2}$. That is, define

   ```
   phi :: Float
   phi = ...
   ```

   Evaluate φ^2, $\varphi^2 - 1$, $\frac{1}{\varphi}$ and $\frac{1}{\varphi - 1}$

3. Write a function that computes the fractional part of a `Float`, e.g. mapping `3.7` to `0.7`.
   Hint: use the `floor` function (and another one – see the diagram of the slide *Functions between types*).

4. Write a definition of the function `factorial` as suggested in the lecture last week and apply it to 17. Then try changing its type to operate on values of type `Int` instead of `Integer`, and try again. Try bigger numbers.

5. Define the function `between` used on the slides. (Top-down design)
   Hint: you could use guards here, but it's probably cleaner not to.

   ```
   between 1 2 3
   between 3 2 1
   between 1 3 2
   between 1 2 2
   ```

6. Write a function that converts a character that represents a digit, like `'3'`, to the corresponding integer (`3`), or 0 if the character does not represent a digit. You may assume that the digit characters are contiguous and in ascending order. You will need to place "`import Data.Char`" in your module to gain access to the character functions.

7. In the Gregorian calendar, a year is a leap year if it is divisible by 4, except that that centuries are leap years only if divisible by 400. Thus 2000 was a leap year, but 2100 won't be.

(a) Write a function to test whether a year is a leap year. To test whether a number is divisible by another, use `mod` and compare the remainder with 0, e.g. `y `mod` 4 == 0`.

(b) Write a function that returns the number of days in a year given as argument.

8. Rewrite the function `middleNumber` using a local definition.