Problem Set 1

1. Consider the function `mac` shown below:

   ```
   mac n | n <= 100 = mac (mac (n + 11))
         | otherwise = n - 10
   ```

   For integer values of `n` between -1000 to 1000, plot `mac n` against `n`.

2. Consider the following function:

   ```
   ack 0 n = n + 1
   ack m 0 = ack (m-1) 1
   ack m n = ack (m-1) (ack m (n-1))
   ```

   How will you describe

   ```
   (a) ack 2 n, and
   (b) ack 3 n
   ```

   as mathematical functions of `n`?

3. Write a function `where_smallest f a b` which takes a function `f` and the integer range `[a, b]` and determines the integer point in the given range where the function has the smallest value.

4. Write a function `has_solution a b c` which returns `True` if the diophantine eqn. $ax+by = c$ has solutions for integer values of x and y.

   HINT: The diophantine equation $ax + by = c$ has a solution for integer values of x and y, if $gcd(a, b)$ divides c.

5. Write a function `ak_mult x y` to multiply two numbers using the Al-Khwarizmi method.

6. Define a function `div x y`, $y \neq 0$, which will return a pair of numbers (q, r) such that $x = yq + r$ and $r < y$. This should also work by repeated halvings of x.

7. Given two numbers `a` and `b`, write a function `coeffs` which will return a pair of numbers `x` and `y` such that $ax + by = gcd(a, b)$.

8. Given two integers `x` and `n` and an integer exponent `y`, write a function `modexp x y n` which will output: $x^y \ mod \ n$.

9. A Carmichael number is a non-prime number p such that for every $1 \leq a \leq p$, $a^{(p-1)} = 1 \ mod \ p$. Define a function `carmichael n` which will give the nth Carmichael number.

10. Suppose we wanted to write the following functions, explained through examples of how they may be called:

    (a) `f0 1 100`: Starting with 1, the sum of every 13th number in the interval 1 to 100.

(b) `f1 2 100`: The sum of squares of all odd numbers between 2 and 100.

(c) `f2 1 100`: The product of the factorials of multiples of 3 between 1 and 100.

(d) `f3 10 200`: The sum of the squares of the prime numbers in the interval 10 to 200.

(e) `f4 50`: The product of all positive integers less than 50 that are relatively prime to 50.

Write a higher-order function called filtered-accumulate so that the functions described above are instances of filtered-accumulate. Write the functions f0, f1, f2, f3 and f4 in terms of filtered-accumulate. You can assume that the arguments to f0, f1, f2, f3 and f4 are positive.

11. We want to implement sets. Since the most important thing you can do with a set is to test membership of a given element, we want to represent sets as the datatype `Set(a -> Bool)`. The function `a -> Bool` associated with a set is called the *charateristic function of the set* in set-theoretic jargon. Under such a representation, the empty set is represented as `Set(\x  -> False)`. This is because the empty set returns false when tested for membership with any element. Similarly the complement of `Set f` would be `Set(\x -> not(f x))`.

    Under such a representation, write definitions for the following set theoraetic operations:

    - insert (inserts an element in the set)
    - member(tests for membership)
    - union
    - intersection
    - difference

    What is the advantage of such a representation over a list based representation?

12. Write a function `convert` which will convert a at-most-six-digits number into its verbal description. For example, `convert 308000` returns `"three hunderd and eight thousand"`, `convert 369027` returns `"three hunderd and sixty nine thousand and twenty seven"` and `convert 369401` will returns `"three hunderd and sixty nine thousand and four hundred and one"`.