3.5 Exercises on Chapter 3

3.5.1 Fibonacci Numbers

Write a program such that `java fibonacci n` prints out the nth Fibonacci number. The Fibonacci sequence goes 1,1,2,3,5,8,13,21,34,55,89,$\cdots$.

3.5.2 Multiplication

Multiplication of non-negative integers can be defined recursively in terms of addition:

$$\begin{aligned} \text{mult}(n,0) &= 0 \\ \text{mult}(n,m+1) &= \text{n+mult}(n,m) \end{aligned}$$

Write a class which has a method `mult` which implements such a function.

3.5.3 Exponentiation

Exponentiation of non-negative integers can be defined recursively in terms of Multiplication:

$$\begin{aligned} n^0 &= 1 \\ (n^{m+1}) &= \text{n*}(n^m) \end{aligned}$$

Write a class which has a method `power` which implements such a function.

3.5.4 Reversing Input

Without using `Vectors` or arrays write a program which reads in characters from the keyboard and prints them out in the opposite order to which they were typed in. The program should stop when the user presses the *enter* key.

3.5.5 The Syracuse Sequence

The *Syracuse Sequence* starting with 14 goes like this:
14 7 22 11 34 17 52 26 13 40 20 10 5 16 8 4 2 1
The rule is as follows: if n is even then the next number in the sequence is $n/2$ and if n is odd the next number is $3n + 1$.
The sequence stops at 1.
Using recursion, write a program not containing the word "`while`", such that for all positive integers, n, `java syr n` prints out the Syracuse sequence starting with n.
(Try running `syr.class` to see how your program should behave.) An interesting fact about the Syracuse sequence is that nobody knows whether it always ends with 1 or not! No-one has proved it and no one has found an example that does not end in 1.