Write Ruby programs that will solve problems listed below:

1. Write a program that behaves like this:
   ```
   What is your name? > Daniel
   How old are you? > 24
   Daniel was born in 1986.
   ```
   advice: see the methods `print, puts, gets, gets.chomp`

2. Write a program that asks for a number and a sentence and prints the sentence backwards that many times.
   ```
   Type a sentence: Hello world!
   Type a number: 4
   !dlrow olleH
   !dlrow olleH
   !dlrow olleH
   !dlrow olleH
   ```

3. Write a factorial program, which takes a number from the user and computes the factorial.
   ```
   Type a number: 7
   7! = 5040
   ```

4. Write a program that asks for the maximum number and computes the corresponding power of two no more than the entered number.
   ```
   Enter a maximum value: 124123
   2^16 = 65536 is the highest power of two less than 124123
   ```

5. Write a Deaf Grandma program. Whatever you say to grandma (whatever you type in), she should respond with HUH?! SPEAK UP, SONNY!, unless you shout it (type in all capitals). If you shout, she can hear you (or at least she thinks so) and yells back, NO, NOT SINCE 1938! To make your program really believable, have grandma shout a different year each time; any year at random between 1930 and 1950. You can't stop talking to grandma until you shout BYE.
   ```
   You: Hello
   Deaf Grandma: HUH?!  SPEAK UP, SONNY!
   You: HELLO
   Deaf Grandma: NO, NOT SINCE 1938!
   You: BYE
   ```

6. Write a function that allows for checking whether a given string starts with the correct determination of protocol in the URL. The correct name for the protocol consists of 3 to 5 lowercase letters, followed by a sequence “: / /”. Use the regular expressions.
   ```
   puts ‟http://www.wi.pb.edu.pl”.url_protocol? #produces true
   puts ‟ala ma kota”.url_protocol? # produces false
   ```

7. Make an `OrangeTree` class. It should have a `height` method which returns its height, and a `oneYearPasses` method, which, when called, ages the tree one year. Each year the tree grows taller (maximum height is 10m), and after 50 years the tree should die. For the first few years, it should not produce fruit, but after a while it should, and older trees produce more each year than younger trees. And, of course, you should be able to `countTheOranges` (which returns the number of oranges on the tree), and `pickAnOrange` (which reduces the `@orangeCount` by one and returns a string telling you how delicious the orange was, or else it just tells you that there are no more oranges to pick this year). Make sure that any oranges you don't pick one year fall off before the next year. Your program should behaves like this:

   ```
   orange_tree = OrangeTree.new      #creates 0 years old tree
   orange_tree.oneYearPasses
   puts orange_tree                  #produces ‟Orange tree - age: 1 year, hight: 0.5 m”
   orange_tree.pickAnOrange          #produces ‟there are no more oranges to pick this year”
   7.times {orange_tree.oneYearPasses}
   orange_tree.pickAnOrange          #produces ‟orange is delicious”
   orange_tree.countTheOranges       #produces ‟there are 24 oranges on the tree”
   orange_tree.pickAnOrange          #produces ‟orange is delicious”
   orange_tree.countTheOranges   #produces ‟there are 23 oranges on the tree”
   55.times {orange_tree.oneYearPasses}
   puts orange_tree                  #produces ‟Orange tree - the tree is died”
   ```

8. Write in the class `Range` method `todown` behaves like this (use `block_given?` and `yield` functions):

   ```
   (3..5).todown {|i| print "hi#{i} "}     #produces hi5 hi4 hi3
   (2...7).todown                          #produces 65432
   ```

Prepared by Tomasz Łukaszuk with the help of:
http://ruby-doc.org/docs/Tutorial/
http://pine.fm/LearnToProgram by Chris Pine