CSE 341, Autumn 2005, Assignment 1
ML Warmup
Due: Wed October 5, 10:00pm

1. Write a function `spherevolume` that takes a real number representing the radius of a sphere, and that returns the volume of the sphere. To head off some of the usual questions ... "What's the formula for the volume of a sphere?" Ans: look it up. (OK, that wasn't super-helpful, was it?) "What value of π should we use?" Ans: use any reasonable number of decimal places, say at least 4. Or better, use `Math.pi`.

2. Write a recursive function `squares` that takes a list of integers, and returns a list of the squares of those integers. For example, `squares([1,4])` should evaluate to `[1,16]`, while `squares([])` should evaluate to `[]`.

3. Write a function `repeat` that takes any value x and an int n, and returns a list containing n occurrences of x. For example, `repeat(false,4)` should return `[false,false,false,false]`, and `repeat(false,0)` should return `[]`. If n is negative, also return an empty list. Your function should have the most general possible type. In a comment in the code, give the type of your function, and give two examples of using the function on different types of values for x. (Hint: if you just let SML infer the type of the function, it will be the most general possible — you'd actually have to go out of your way to get it to have a more restricted type.)

4. Write a function `repeatlist` that takes a list and an int n, and returns a new list that is n times as long, with each element repeated n times. For example, `repeatlist(["peter", "paul", "mary"], 2)` should return `["peter","peter","paul","paul","mary,"mary"]`, and `repeatlist([],10)` should return `[]`. Again, your function should have the most general possible type. In a comment in the code, give the type of your function, and give two examples of using the function on lists of different types. (Hint: you can call the function you wrote for Question 3.)

5. Write a function to test whether a list of ints is in strict ascending order. For example, `ascending([1,2,3])` should return true, while `ascending([2,3,1])` and `ascending([2,2])` should both return false. You should handle the empty list, and a list of one number. (What should these return? Justify your decision in a comment in the code.)

Turnin:

Turn in two files: the source listing for your functions (all in one file), and a log file showing the results of testing the functions. You should exhibit thorough tests that exercise the different cases. For example, if your function takes a list as an argument, test it with an empty list, a list with one element, and a list with several arguments. As another example, if your function takes a number, include tests that exercise the different possibilities (is something special supposed to happen with 0 or a negative number as an argument, for example)?

If you're running SML from emacs, you can produce the log file just by saving the sml buffer. If you're running from the command line, you can use the unix 'script' command.

Assessment: Your solutions should be:

- Correct
- Tastefully commented
- In good style, including indentation and line breaks
- Written in a functional style (no side effects).

- Well tested

Your solution should be short. (Mine was 8 lines, excluding comments and blank lines.) You don't need to be that terse, but if you find yourself writing lots of code, rethink your approach.