Assignment 1: Programming with Higher-Order Functions

Due: Feb 2 2016 1AM
Support code: https://www.cs.umass.edu/~arjun/courses/cmpsci631-spring2016/hw/hof.zip

The goal of this assignment is to explore the kinds of programming patterns that can be encoded with common higher-order functions. You only need to write ten functions, the hardest of which is insertion sort. However, you must use higher-order functions and may not use linguistic features that you’re used to.

1 Restrictions

Your code must adhere to these restrictions:

- You may use the foldr and unfold functions from the first lecture as helper functions.
- You may not use any other built-in or library functions. (In particular, you must not use the List library.)
- You may not use explicit recursion (i.e., do not write let rec)
- You may not use loops or mutable state.

2 Programming Task

You should fill in the file Main.ml, which is included as part of the support code. You can run make to build the code and make test to run tests.

- Write the length lst function, which produces the length of lst.
- Write the filter p lst function, which produces a list that contains the elements of lst that satisfy the predicate p, in order.
- Write the build_list f n function, which produces a list of length n, where the ith element is the result of f i. i.e., build_list f 5 is [f 0; f 1; f 2; f 3; f 4].
• Write the `is_empty lst` function, which produces `true` if `lst` is `[]` and `false` otherwise. In addition to the restrictions above, you may neither use any (in-)equality-testing operators nor `match .. with`.

• Write the `zip [x1; x2; ..] [y1; x2; ..]` function, which produces the list of pairs `[(x1, y1); (x2, y2); ..]`. If the two lists have unequal lengths, the extra elements in the longer list are ignored.

• Write `map_using_fold f lst`, which should be equivalent to `map`. Use `fold` to write this function.

• Write `map_using_unfold f lst`, using `unfold`.

• Write `factorial n`.

• Write `insert n lst`, which inserts the integer `n` into `lst`. Assume that the list sorted in ascending order and the function should preserve the order.

• Write `insertion_sort lst`. 