Foundations of Computer Science – ML Workshop Preparation

Heidi Howard

January 7, 2015

In preparation for the ML workshop, please work through the follow questions. They are designed to be fairly straight forward and refresh you knowledge after the Christmas break. Please try not to use your lecture notes and bring your solutions with you to the workshop.

Exercise 1:
Why might a programmer choose to use ML, instead of an imperative language such as Java or C++?

Exercise 2:
Write a function `xor` which takes two booleans and return true if either of them are true and false otherwise.

Exercise 3:
Write a function to compute the factorial of an integer, how can you adapt this for floating point values?

Exercise 4:
Solve the recurrence equations

$$T(0) = 1$$

and

$$T(n+1) = T(n) + n$$

Exercise 5:
Write a function to test if a given list is empty.

Exercise 6:
Write a function which given a list of integers, returns the sum of the integers in the list.

Exercise 7:
Write a function to reverse a list in $0(n)$.

Exercise 8:
Write a function which given a list, returns true if all the elements of the list are the same and false otherwise. What is the type of this function? Justify your answer.

Exercise 9:
Write a function called `zip`, which takes tuple of two lists and returns a list of tuples containing the elements of the lists, for example:

```
zip ( [1,2,3,4,5], [2,4,6,8,10] );;
> [ (1,2), (2,4), (3,6), (4,8), (5,10)]
```

Exercise 10:
Write a function to check if a list is sorted.

Exercise 11:
Write a function to merge two sorted lists.

Exercise 12:
Give a ML datatype for binary trees.

Exercise 13:
Give an ML datatype for ML's built-in lists.

Exercise 14:
Implement ML's `hd` function which returns the first element of a list or if the list is empty, throws an exception called `Empty`.

Exercise 15:
Write a function which given a integer binary tree, returns the sum of the integers in the binary tree.

Exercise 16:
An association list is a list of pairs which can be used for a simple implementation of a dictionary. Implement a dictionary as an association list with the following common dictionary operations: empty, lookup, update and delete.

Exercise 17:
Describe the structure of binary search trees. With a diagram, show how inserting the same set of keys in a different order can result in different shaped binary search tree.

Exercise 18:
Implement a dictionary using a binary search tree with the following dictionary options: empty, lookup and update.

Exercise 19:
Explain the difference between preorder, inorder and postorder tree traversal and write a function to do each (do not worry about efficiency).

Exercise 20:
What is a functional array? Write a function to lookup a value in a functional array.

Exercise 21:
Write a function called `map` which takes a function and a list and applies the function to each of the elements and returns the result.

Exercise 22:
Give the ML datatype for lazy lists.

Exercise 23:
Write a function which given a lazy list and a positive integer n, returns a normal ML list containing the first n elements of the lazy list.

Exercise 24:

Write a function called `interleave` which takes two lazy lists and joins them, interleave the elements from the two original lists.

Exercise 25:

Describe the difference between breadth first, depth first and iterative deepening tree traversal.

Exercise 26:

Implement breadth first and depth first tree traversal in ML, use lists for both and do not worry about efficient.

Exercise 27:

Describe how a pair of lists can be used to implement a queue in ML, include ML code for the common queue operations

Exercise 28:

Describe how we can represent univariate polynomials in ML using `(int * real) list`.

Exercise 29:

Comment, with examples, on the difference between an `int ref list` and an `int list ref`.

Exercise 30:

Write a version of map (from question 21) for arrays instead off lists.