Quiz 16: ArrayLists and recursion SOLUTION

CSCI 110 Section 1

Monday, October 3, 2016

1) Write a recursive function that takes a List<Integer> and an int, and returns true if the int exists in the List<Integer>. (this should be recursive i.e. do not use loops, indexOf, or contains). [warmup]

We have to write a function.
What should our return type be? **boolean**
What should our parameters be? **a List<Integer> (the one we're searching in) and an int (the int we're trying to find)**

We have to write a recursive function.
What should our base case be? **if the list is empty, the answer must be false.**
How can we use the solution of a smaller problem (or problems) to solve a bigger problem? **The element we're looking for is either the first element, or it's in the rest of the list. We can check whether it's the first element, and make a recursive function call to check whether it's in the rest of the list.**

```java
boolean hasElem(List<Integer> ints, int toFind) {
    // base case
    if (ints.size() == 0) {
        return false;
    }

    // is the first element what we're looking for?
    if (ints.get(0) == toFind) {
        return true;
    }

    // search the rest of the list
    List<Integer> restOfList = ints.subList(1, ints.size());
    return hasElem(restOfList, toFind);
}
```

2) Write a recursive function that takes a List<Integer> and returns the element with the maximum value. Hint: Math.max() returns the greater of two int values, so Math.max(5, 7) == 7. [100 points]

We have to write a function.
What should our return type be? **int (the maximum value)**
What should our parameters be? **a List<Integer> (the one we're searching in)**

We have to write a recursive function.
What should our base case be? **If the list has only one element, that element is obviously the max element.**
How can we use the solution of a smaller problem (or problems) to solve a bigger problem? **The max element we're looking for is either the first element or it's in the rest of the list. Compare the current element with the max from the rest of the list, and return the greater value using Math.max(). We get the max from the rest of the list by making a recursive function call.**

What if we pass in a List with zero elements? It doesn't really make sense to get the max element of an empty List. You could throw an Exception, but the route we've taken here is to return the smallest possible integer.

```java
int maxElem(List<Integer> ints) {
    // it doesn't make sense to get the max element from
    // an empty list so we return the smallest possible number, -2^31
    if (ints.size() == 0) {
        return Integer.MIN_VALUE;
    }

    // for a list with one element the max is the only element
    if (ints.size() == 1) {
        return ints.get(0);
    }

    // otherwise compare the first element to the max
    // from the rest of the list
    List<Integer> restOfList = ints.subList(1, ints.size());
    int maxFromRestOfList = maxElem(restOfList);
    return Math.max(ints.get(0), maxFromRestOfList);
}
```