Liz is worth 30 points, with 5 points extra credit.

1. (10p) Create the function *LongestString(L)* which will be given a non-empty list of strings and will return the longest string, or one of the longest, if there is a tie.
Examples:

```
LongestString([“hi”,”there”,”Fred”]) # “there”
LongestString([“whatever”])  # “whatever”
LongestString([“This”,”is”,”weird”,”very”,”crazy”])  # “weird” or “crazy”
```

2. (10p) Create the function *IsIncreasing(L)* which will be given a (potentially empty) list of numbers and will return *True* if the numbers are in strictly increasing order and *False* otherwise. That is, return *True* if and only if each number is larger than the previous one. Return *False* if the list is empty.
Examples:

```
IsIncreasing([1,1.02,3])  # True
IsIncreasing([1,1.02,3,2.9])  # False
IsIncreasing([1,1]) # False
IsIncreasing([3]) # True
IsIncreasing([]) # False
```

3. (10p) Create the function *Interleave(A,B)* which will be given two lists, and will return a list that interleaves the elements from the two input lists by taking elements alternately from each. That is, *Interleave()* effectively zips the two lists together for as long as each list has an element available. If one list is exhausted first, include the remaining elements of the other.
Examples:

```
Interleave([‘a’,’b’,’c’],[2,4,6]) # produces [‘a’,2,’b’,4,’c’,6]
Interleave([“hi”, “there”],[1,5,-3,6,7]) # produces [“hi”,1,”there”,5,-3,6,7]
Interleave([3,4,5],[‘whatever’]) # produces [3,’whatever’,4,5]
```

Extra Credit: (5p) Create the function *FullHouse(L)* which will be given a list of 5 elements, and will return *True* if there is one pair (2 identical elements) and one triplet (3 identical elements), otherwise *False*. The elements of the pair and the triplet must be distinct – a single element cannot be counted as a member of both a pair and a triplet.
Examples:

```
FullHouse([2,2,7,7,7]) # True
FullHouse([‘a’,12,12,’a’,12]) # True
FullHouse([4,4,4,4,4]) # False
```