Homework 4

1. **(5.20)** Write a function intersect() that takes two lists, each containing no duplicate val- ues, and returns a list containing values that are present in both lists (i.e., the intersection of the two input lists).
   >>> intersect([3, 5, 1, 7, 9], [4, 2, 6, 3, 9])
   [3, 9]
2. **(5.23)** Write function pay() that takes as input an hourly wage and the number of hours an employee worked in the last week. The function should compute and return the employee's pay. Overtime work should be paid in this way: Any hours beyond 40 but less than or equal 60 should be paid at 1.5 times the regular hourly wage. Any hours beyond 60 should be paid at 2 times the regular hourly wage.
   >>> pay(10, 35)
   350
   >>> pay(10, 45)
   475.0
   >>> pay(10, 61)
   720.0
3. **(5.29)** Write function lastfirst() that takes one argument—a list of strings of the format <LastName, FirstName>—and returns a list consisting two lists:
   (a) A list of all the first names (b) A list of all the last names
   >>> lastfirst(['Gerber, Len', 'Fox, Kate', 'Dunn, Bob'])
   [['Len', 'Kate', 'Bob'], ['Gerber', 'Fox', 'Dunn']]

4. **(5.46)** An inversion in a sequence is a pair of entries that are out of order. For example, the characters F and D form an inversion in string 'ABBFHDL' because F appears before D; so do characters H and D. The total number of inversions in a sequence (i.e., the number of pairs that are out of order) is a measure of how unsorted the sequence is. The total number of inversions in 'ABBFHDL' is 2. Implement function inversions() that takes a sequence (i.e., a string) of uppercase characters A through Z and returns the number of inversions in the sequence.

   ```
   >>> inversions('ABBFHDL')
   2
   >>> inversions('ABCD')
   0
   >>> inversions('DCBA')
   6
   ```

5. **(5.48)** Let list1 and list2 be two lists of integers. We say that list1 is a sublist of list2 if the elements in list1 appear in list2 in the same order as they appear in list1, but not necessarily consecutively. For example, if list1 is defined as [15, 1, 100] and list2 is defined as [20, 15, 30, 50, 1, 100] then list1 is a sublist of list2 because the numbers in list1 (15, 1, and 100) appear in list2 in the same order. However, list [15, 50, 20] is not a sublist of list2. Implement function sublist() that takes as input lists list1 and list2 and returns True if list1 is a sublist of list2, and False otherwise.

```
>>> sublist([15, 1, 100], [20, 15, 30, 50, 1, 100])
True
>>> sublist([15, 50, 20], [20, 15, 30, 50, 1, 100])
False
```

* Note, problem 4 is probably harder than 1-3, and problem 5 is probabably harder than 4. Give yourself plenty of time to finish these.

Also note that none of the problems print anything. All problems RETURN their answers.