No Leading Zeros

```
def compile_formula(formula, verbose=False):
    """Compile formula into a function. Also return letters foun
    in same order as parms of function. The first digit of a mul
    number can't be 0. So if YOU is a word in the formula, and t
    is called with Y eqal to 0, the function should return False

    # modify the code in this function.

    letters = ''.join(set(re.findall('[A-Z]', formula)))
    firstletters = set(re.findall(r'\b([A-Z])[A-Z]', formula))
    parms = ', '.join(letters)
    tokens = map(compile_word, re.split('([A-Z]+)', formula))
    body = ''.join(tokens)
    if firstletters:
        test = ' and '.join(L+'!=0' for L in firstletters)
        body = '%s and (%s)' % (test, body)
    f = 'lambda %s: %s' % (parms, body)
    if verbose: print f
    return eval(f), letters
```

Floor Puzzle

```
import itertools

def floor_puzzle():
    floors = bottom,_,_,_,top=[1,2,3,4,5]
    orderings=list(itertools.permutations(floors))
    for (Hopper,Kay,Liskov,Perlis,Ritchie) in orderings:
        if(Hopper is not top
            and Kay is not bottom
            and Liskov is not top
            and Liskov is not bottom
            and Perlis > Kay
            and abs(Ritchie - Liskov)>1
            and abs(Liskov - Kay)>1):
            return [Hopper, Kay, Liskov, Perlis, Ritchie]

print floor_puzzle()
```

Subpalindrome

```
def longest_subpalindrome_slice(text):
    "Return (i, j) such that text[i:j] is the longest palindrome ir
    if text == '': return (0,0)
    def length(slice): a,b = slice; return b-a
    candidates = [grow(text, start, end)
                  for start in range(len(text))
                  for end in (start, start+1)]
    return max(candidates, key=length)

def grow(text,start,end):
    while(start>0 and end<len(text)
          and text[start-1].upper()==text[end].upper()):
        start -= 1; end +=1
    return (start, end)
```