1. SAT

(not a or not b or not c) and (not a or c) 
and (not a or b or c) and (a or not b or not c) and (not a) 
and (a or not c) and (not a or not b or c) and (not b or c) 
and (b or c) and (a or c)

- Is this formula satisfiable? If so, give a short “yes” proof.
2. Valid or Not

• For each problem, say whether it is valid.

i. Input: A list of integers. Output/Property: The mode---the number that appears the most times in the list (ties broken arbitrarily).

ii. Input: A list of integers. Output/Property: The integer in the list that would make the best punchline to “How many Computer Scientists does it take to screw in a lightbulb?”

iii. Input: A list of integers. Output/Property: The length of the longest consecutive run of integers in the list.

3. Yes Proofs

• Show that each of these problems is in NP by explaining the short yes proof.

i. Input: A list of integers. Output/Property: Is the list out of order?

ii. Input: A word. Output/Property: Does it have any repeated letters in it?

iii. Input: A sudoku puzzle. Output/Property: Does it have a solution?

iv. Input: A list of words. Output/Property: Is there a pair of words on the list whose concatenation is on the list?
4. NP Problem

- Pick an NP problem you think might be useful for a computer to solve.
- Formalize it as a computer science problem by describing the input, output, property, and short yes proof.

5. Local Search

i. For the puzzle discussed in class, what is the score for this grid?

ii. What local move (swapping two tiles) improves the score?
6. Halting

- For what integer values of $n$ will the following programs halt?
  
i. tuna1?
  
ii. tuna2?
  
iii. tuna3?

```python
def tuna1(n):
    while (n == n + 1):
        print n
        n = n + 1

def tuna2(n):
    while (n > 10):
        print n
        n = n - 1

def tuna3(n):
    while (n < 40):
        print n
        n = n - 1
```

Extra Credit

- The remaining problems require a bit more thought!

- Try these for fun.
7. Game Tree

- Label each node of this game tree as a win (W) or loss (L) for the player who moves at that node (red or blue).

8. Mystery Routines

- The python routines on the next page take a list as input and return either true or false. Which routine answers each of these questions?
  
i. Are there two duplicate numbers in a row?
  
ii. Is the last number in the list the largest number in the list?
  
iii. Is the list a palindrome: the same forwards and backwards?
The Mysteries

def mystery1(l):
    for i in range(len(l)-1):
        if (l[i] == l[i+1]): return True
    return False

def mystery2(l):
    for i in range(len(l)):
        if (l[i] != l[len(l)-i-1]): return False
    return True

def mystery3(l):
    for i in range(len(l)-1):
        if (l[i] > l[len(l)-1]): return False
    return True