Valid parentheses (solution)

```
def validParentheses(s):
  stack = []
  open = '([{'
  closed = ')]}'
  for c in s:
    if c in open:
      stack.append(c)
    elif c in closed:
      if not stack:
        return False
      if open.find(stack[-1]) == closed.find(c):
        stack.pop()
      else:
        return False
    else:
      return False
  return not stack
```

- Runtime: O(n)
- Space: O(n)

"New" data structure: stack.

Basically just another way of using a dynamically extendable list, like Python's default list, with O(1) append/pop operations.

Rest of the problems...

- 55. Jump game - dynamic programming
- 62. Unique paths - dynamic programming OR combinatorial mathematical solution
- 136. Single number - XOR (bitwise operations)
- 226. Invert binary tree - classic example of a graph question