CS 116 TUTORIAL 4

LISTS, MUTATION
REMINDER

- Assignment 04 due next Friday, February 9<sup>th</sup> (at 10:00AM)
- Temporary ISA Office Hours at MC4065, next Wednesday (Feb 7<sup>th</sup>) from 10 AM to 2PM.

- Midterm is on February 26<sup>th</sup> at 7 PM
COMMON LIST FUNCTIONS

- \texttt{len(L)} => returns length of \texttt{L}
- \texttt{L[i]} => returns element at index \texttt{i}
- \texttt{L[i:j]} => returns \texttt{L} from \texttt{i} to \texttt{j-1}
- \texttt{x in L} => returns True if \texttt{x} is in \texttt{L} and False otherwise.

- \texttt{L.append(x)}
- \texttt{L.remove(x)} \hfill \text{Examples of functions that \textit{mutates} lists.}

- See Module 04 Slide 8 for other list functions and use Python's help function
**ABSTRACT LIST FUNCTIONS: MAP, FILTER**

- **map**
  - applies function to each element in list
  
  ```python
  list(map(fun_name, L))
  ```

  *Used to turn it into a list*

- **Filter**
  - returns an iterator matching the elements in list for which function returns true.
  
  ```python
  list(filter(fun_name, L))
  ```

  *Used to turn it into a list*

*Note: map and filter can also be applied to strings.*

*Note: fun_name must have only one parameter/argument.*
**LAMBDA**

\[ \text{lambda } x, y, \ldots, z: \text{ body here} \]

**Parameters of lambda (no brackets)**

**Note:** Can be applied to strings as well

For example:

```python
def just_letter(sentence):
    return list(filter(lambda c: c.isalpha(), sentence))
```

Note: `just_letters(sentence)` can also be applied to strings. However it will return a list of strings of length 1, rather than a new string. The list can later be joined into a string with `list.join(str)`
ITEM DEFINITION

A Card is a list of length 2 where
- the first item is an integer between 1 and 10, inclusive, representing the value of the card, and
- the second item is a string ("hearts", "spades", "clubs", or "diamonds") representing the suit of the card.

Example: [1, "hearts"] represents the ace of hearts
Write a function `create_cards` that consumes two lists with same length, which are a list of card values (integers between 1 and 10), and a list of suit values (one of the four suit strings), and returns a sorted list of Card from smallest to largest, created pair-wise from the consumed lists (values and suits).

- For example,

```python
create_cards([4,1,10], ['hearts', 'diamonds', 'clubs'])
=> [[1, 'diamonds'], [4, 'hearts'], [10, 'clubs']]```
Write a function `choose_by_colour` that consumes a list of Card (hand) and a string "red" or "black" (colour) and returns a list of the values of the Card in hand of the appropriate colour (spades and clubs are "black", hearts and diamonds are "red").

For example,

```python
choose_by_colour([[1,'hearts'],
                  [9,'spades'],
                  [3,'diamonds']], 'red')
⇒ [1,3]
```

Write this function twice. First, use regular recursion. Then, use abstract list functions.
QUESTION 3

a) Write a function `flip_colour` that consumes a Card, `c`, and mutates the suit of that Card to a different colour: if `c` is a heart, it is mutated to a spade (and vice versa), while if `c` is a club, it is mutated to a diamond (and vice versa).

b) Write a function `flip_hand` that consumes a list of Card (hand), and mutates the suit of each Card in the list so that their colours are flipped in the same way as in `flip_colour`.

![Card suit icons with heart and spade arrows, and diamond and club arrows]
Write a function `modify_list` that consumes a list of integers (called `nums`) and a single integer (`n`). The function returns `None`, but mutates the list in the following way:

- If `n` does not appear in `nums` then add it to the end of `nums`.
- If `n` appears once, then remove `n` from `nums`.
- If `n` appears at least twice, remove the first and last occurrences of `n`. 
Write a function `sanitize` that consumes a string, s, and returns a similar string but with any non-alphanumerical characters removed.

- **For example:** `sanitize("@Test@") => "Test"`