Assignment - Recursion

1. Number of Digits

For a given positive integer value, write a recursion function `digit_count(n)` to find out how many digits it has.

```
If n < 10, return 1.
Otherwise, return 1 + the number of digits in n/10.
```

In [2]:
```
# f(12345)
# 1 + f(1234)
# 1 + 1 + f(123)
# 1 + 1 + 1 + f(12)
# 1 + 1 + 1 + 1 + f(1)

def digit_count(n):
    if n < 10:
        return 1
    return 1 + digit_count(n // 10)

digit_count(1234)
digit_count(112233)
```

Out[2]:
```
6
```

2. Sum of 1..n

For a given value n, write a recursion function `sum_n(n)` to find the sum value of 1 + 2 + ... + n.

In [3]:
```
# return 1 + 2 + 3 + .... + n
# return f(n-1) + n
# return f(n-2) + (n-1) + n

def sum_n(n):
    if n == 1:
        return 1
    return sum_n(n-1) + n

sum_n(5)
```

Out[3]:
```
15
```

3. Sum of a List

Write a recursive function `sum_list(X)` to find sum of all elements in X.

In [6]:
```
# f([1,2,3,4,5,6])
# 1 + f([2,3,4,5,6])
# 1 + 2 + f([3,4,5,6])

# def sum_list(X):
#     if len(X) == 1:
#         return X[0]
#     return X[0] + sum_list(X[1:])

def sum_list(X):
    if len(X) == 0:
        return 0
    elif len(X) == 1:
        return X[0]
    n = len(X)//2
    return sum_list(X[:n]) + sum_list(X[n:])

sum_list([1])
```

Out[6]:
```
1
```

4. Palindrome Detector

A palindrome is a word, number, phrase, or other sequence of characters which reads the same backward as forward, such as madam or 10801.

Implement a recursive function `palindrome(str)` which returns True if `str` is a palindrome.

In [7]:
```
def palindrome(s):
    if len(s) == 0:
        return True
    if s[0] != s[-1]:
        return False
    else:
        return palindrome(s[1:-1])

s = 'madam'
palindrome(s)
```

Out[7]:
```
True
```

5. Find Max Value in List

Implement a recursive function `find_max(s)` to find max value in the input list.

- What should be the inputs to the function?

In [ ]:
```
# [a, b, c, d]
# a [b, c, d]
# a b [c, d]

def find_max(s):
    if len(s) == 1:
        return s[0]

    m = find_max(s[1:])
    if m > s[0]:
        return m
    else:
        return s[0]
```

Enhancement

Enhance your `find_max()` function to a `find_min_max(s)` function which finds both min and max value in the input list.

In [ ]:
```
def find_min_max(s):
    if len(s) == 1:
        return s[0], s[0]
    mi, ma = find_min_max(s[1:])
    if s[0] < mi:
        return s[0], ma
    elif s[0] > ma:
        return mi, s[0]
    else
        return mi, ma
```