Practice 1

Create a recursive function that accepts two integer argument a and b, and return the result of a power b

Solution Practice 1

```
1 def pow (a, b):
2     if (b == 0):
3         return 1
4     else:
5         return a * pow (a, b-1)
```

Practice 2

Create a recursion function that accepts a list as the argument, and return the length of the list

Solution Practice 2

```
1 def panjang_list(a_list):
2     if not a_list:
3         return 0
4     else:
5         return 1 + panjang_list(a_list[1:])
```

Practice 3

Create a recursion function that accepts a non-negative integer a as the argument, and return the sum of the digit. Example -> 45 = 4+5=9

Solution Practice 3

```
1 def sum_of_digits (a):
2     if (a == 0):
3         return 0
4     else:
5         return a%10 + sum_of_digits(a//10)
```

Practice 4

Create a recursion function that accepts a string argument, and return the result if the string is a palindrome or not (True/False)

Solution Practice 4

```
1  // Solution 1
2  def is_pal_rec (a_str, first_index, last_index):
3      if (first_index == last_index):
4          return True
5
6      if (a_str[first_index] != a_str[last_index]) :
7          return False
8
9      if (first_index < last_index + 1) :
10         return is_pal_rec(a_str, first_index + 1, last_index - 1);
11
12 def is_palindrome (a_str):
13     n = len(a_str)
14
15     if (n == 0):
16         return False
17
18     return is_pal_rec(a_str, 0, n - 1)
19
20 // Solution 2 - Credits to Zuhdy
21 def is_palindrome (a_str):
22     if (len(a_str) == 0):
23         return True
24
25     panjang_string = len(a_str)
26     return (a_str[0] == a_str[panjang_string-1]) and is_palindrome(a_str[1
           :panjang_string-1])
```

Practice 6

Create a recursion function that accepts a string argument a, and return the reverse of the string

Practice 7

Create a recursion function that accepts an integer list a, and return the maximum value of the list