CSCI-SHU 210 Data Structures

Assignment 3 Recursion

About 84% of this assignment can be auto graded. Again, we have two submission options: You can submit everything to gradescope; (Which means you can see 84% of your grade earlier) You can also submit everything on NYU Classes.

Question 1 (x to the power of n):

Implement function, power(x,n) **recursively**. n could be any integer (positive, negative, or zero). In the text book, power(x,n) python program (Page Number 173 Code Fragment 4.12) only handles non-negative integers. Your power(x,n) function should be able to handle both positive and negative integers.

Example 1:

```
Input: power(-5, 2)      # (-5) ^ 2 = 25
Returns: 25
```

Example 2:

```
Input: power(4, -1)      # 4 ^ (-1) = 0.25
Returns: 0.25
```

Question 2 (Covert recursion to iteration):

Rewrite the following **recursive** function using iteration instead of recursion.

```
def recur(n):
    if (n < 0):
        return recur(-n)
    elif (n < 10):
        return n
    else:
        return (n % 10 + recur(n // 10))
```

Question 3 (Element Uniqueness Problem):

Consider the Unique3 (Recursive Element Uniqueness Problem) program from the text book (Page Number 165 Code Fragment 4.6). Worst case runtime for this program could be O(2^n).

Implement an efficient **recursive** function unique(S) for solving the element uniqueness problem, which runs in time that is at most O(n^2) in the worst case without using sorting.

- parameter1: List
- Return True if there are no duplicate elements in List S.
- Return False otherwise

Example 1:

```
Input: unique([1, 7, 6, 5, 4, 3, 1])
Returns: False
```

Example 2:

```
Input: unique([9, 4, 'a', [], 0])     # All elements are unique
Returns: True
```

Question 4 (Pascal's Triangle):

In mathematics, Pascal's triangle is a triangular array of the binomial coefficients.
Implement a **recursive** function pascal(N) to generate a list of pascal values. Pascal triangle looks like the following:

```
          1
        1   1
      1   2   1
    1   3   3   1
  1   4   6   4   1
1   5  10  10   5   1
```

Implement recursive function pascal(N), which generates list of sublists, each sublist contains a level of Pascal values.

- parameter1: number of levels
- return: list of sublists. Those lists contain pascal values of each level.

Example:

```
Input: pascal(5)     # five levels of pascal values
Returns: [[1], [1,1], [1, 2, 1], [1, 3, 3, 1], [1, 4, 6, 4, 1]]
```

Problem 5 (All Possible Combinations)

Complete function generateBillboard(S): this function enumerates and displays all possible casts (combinations) of celebrities from a specified list that can appear on the billboard of a movie with a given number of lead roles.

- parameter1: Input List
- parameter2: Integer, number of celebrities appearing together.
- parameter3: helper List for recursion enumerating
- parameter4: helper Integer for recursion indexing
- Return: list of all possible combinations of celebrities.

Example:

```
casts = ["Johnny Depp", "Al Pacino", "Robert De Niro
", "Kevin Spacey", "Denzel Washington", "Russell Cro
we", "Brad Pitt"]
```

Input: generateBillboard(casts, 4, [], 0) # Choose 4 out of 7

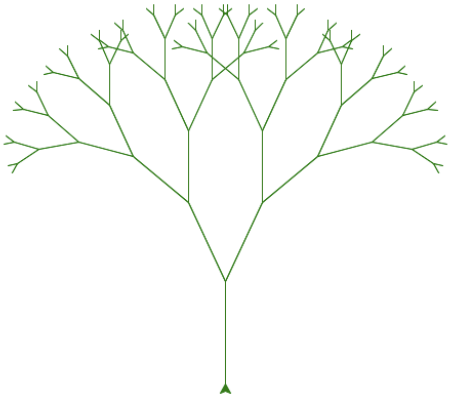
Outputs:

```
[['Kevin Spacey', 'Denzel Washington', 'Russell Crowe', 'Brad
Pitt'], ['Robert De Niro', 'Denzel Washington', 'Russell
Crowe', 'Brad Pitt'], ['Robert De Niro', 'Kevin Spacey',
'Russell Crowe', 'Brad Pitt'], ['Robert De Niro', 'Kevin
Spacey', 'Denzel Washington', 'Brad Pitt'], ['Robert De Niro',
'Kevin Spacey', 'Denzel Washington', 'Russell Crowe'], ['Al
Pacino', 'Denzel Washington', 'Russell Crowe', 'Brad Pitt'],
['Al Pacino', 'Kevin Spacey', 'Russell Crowe', 'Brad Pitt'],
['Al Pacino', 'Kevin Spacey', 'Denzel Washington', 'Brad
Pitt'], ['Al Pacino', 'Kevin Spacey', 'Denzel Washington',
'Russell Crowe'], ['Al Pacino', 'Robert De Niro', 'Russell
Crowe', 'Brad Pitt'], ['Al Pacino', 'Robert De Niro', 'Denzel
Washington', 'Brad Pitt'], ['Al Pacino', 'Robert De Niro',
'Denzel Washington', 'Russell Crowe'], ['Al Pacino', 'Robert De
Niro', 'Kevin Spacey', 'Brad Pitt'], ['Al Pacino', 'Robert De
Niro', 'Kevin Spacey', 'Russell Crowe'], ['Al Pacino', 'Robert
De Niro', 'Kevin Spacey', 'Denzel Washington'], ['Johnny Depp',
'Denzel Washington', 'Russell Crowe', 'Brad Pitt'], ['Johnny
Depp', 'Kevin Spacey', 'Russell Crowe', 'Brad Pitt'], ['Johnny
Depp', 'Kevin Spacey', 'Denzel Washington', 'Brad Pitt'],
['Johnny Depp', 'Kevin Spacey', 'Denzel Washington', 'Russell
```

Problem 6 (Fractal Tree)

Use Turtle module draw a Tree.

- Must use **recursion**
- Size should be greater than three branches.
- Try to make the tree look pretty by tuning variables.



Turtle module is a python built in module. Turtle module draws lines by moving the cursor.

```
turtle functions explained:
import turtle

t = turtle.Turtle()    # Initialize the turtle

t.left(30)             # The turtle turns left 30 degrees
t.right(30)            # The turtle turns right 30 degrees
t.forward(20)          # The turtle moves forward 20 pixels, leave a line on the path.
t.backward(30)         # The turtle moves backward 30 pixels, leave a line on the path.

… and more! In this recitation, that's all we need.
```

With the mind set of recursion, let's break down this problem.

1. Move forward
2. Make a turn
3. Recursion for a smaller problem
4. Make another turn
5. Recursion for a smaller problem
6. Come back

Base case: If the branch is too small, stop.
Otherwise, we should create two new branches (two recursions, two smaller problems