Dictionaries and Sets

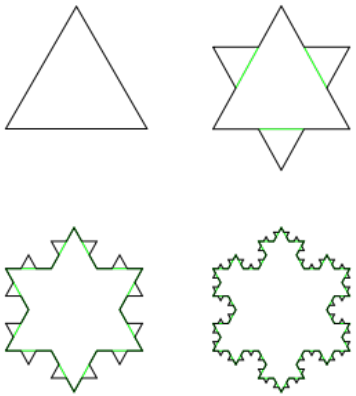
NMT CSE/IT 107

Resources

What is the difference between a list and a tuple?
Python 3 documentation `https://docs.python.org/3/`
Library Reference, Language Reference
Stack overflow. `stackoverflow.com`

Recursion

"Recursion is the process of repeating items in a self-similar way"
"...solution to a problem depends on solutions to smaller instances of the same problem" Wikipedia



Recursive definition of a list
Base case: A list is an element
Recursive step: A list is an element plus a list

In our context recursion means a function that calls itself

Recursive Functions

Recursive functions have two cases. **base case** and **recursive step**

```
1 def func(a,b):
2     if ...  # base case
3      return
4     else:
5      return func(a+1,b) # recursive step
```

Pascal's Triangle

```
"""
    n 0:                  1
    n 1:              1       1
    n 2:          1       2       1
    n 3:      1       3       3       1
    pascal(n,0) = 1 # left end
    pascal(n,n) = 1 # right end
"""
def pascal(n, k):
    """
    n = row, k= col
    """
    # base case
    if k==0 or k==n:
        return 1
    else:
        # recursive step
        return pascal(n-1, k-1) + pascal(n-1, k)
```

Unpacking Lists/Tuples

```
1 tupleA = 10, 3, 100
2 x,y,z = tupleA
```

```
1 listA = [10, 3, 100]
2 x,y,z = listA
```

Not Valid

```
1 tupleA = 10, 3, 100
2 x,y = tupleA
```

```
1 for numerator, denominator in [(1,2), (5,4), (4, 7)]:
2     fraction = numerator / denominator
```

in keyword

- New boolean operator `in`
- The `in` keyword is used to check whether a value is contained inside another object such as a string or list.
- `in` returns True or False

Examples: In

```
colors = ['red','yellow','green']
if 'red' in colors:
   print('Red is in colors')
else:
   print('Red is not in colors')
```

```
valid_commands = ['forward', 'backward', 'left', 'right']
command = input('Please enter a command ')
if command in valid_commands:
    print('Command is valid')
else:
    print('Command is not valid')
```

Slices

- Make sublists from existing list.
- `list_name[start_index : end_index: step]`.
  start_index is included. end_index is excluded. (Up to but not including the end_index)

Intro to Dictionaries

- A dictionary is a collection of key: value pairs
- **Key** is the word
- **Value** is the definition of the word

Python Dictionary Syntax

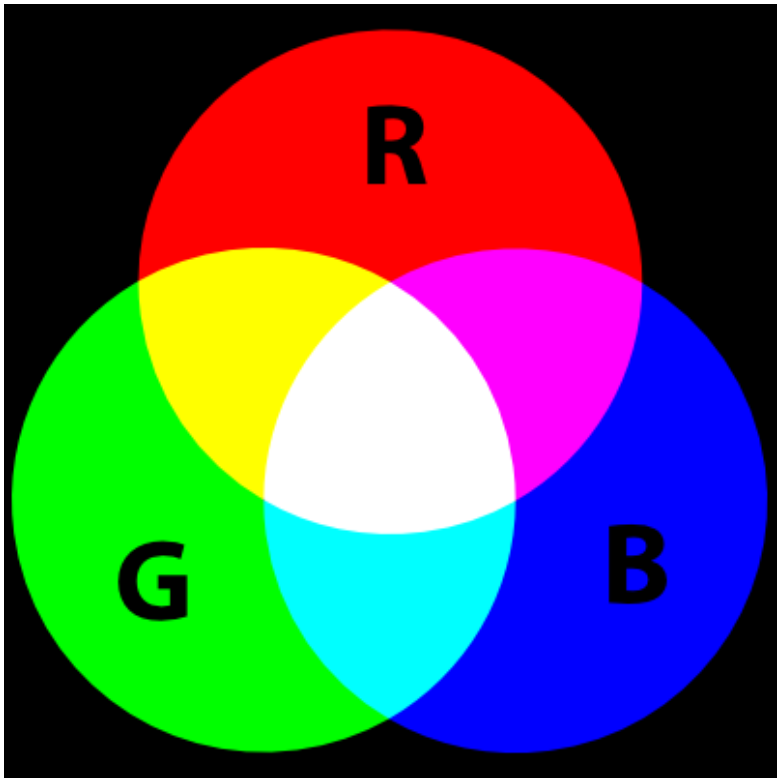
data type is **dict**

```
d = {key1: value1, key2: value2....}
```

```
d = {'hello': 'a typical greeting in the US',
     'earth': 'the planet we live on',
     'python': 'a popular programming language'}
```

```
# empty dictionary
d = {}
```

RGB Color Model



What is the RGB tuple for yellow?

```
colors = {'red': (1,0,0),
          'green': (0,1,0),
          'blue': (0,0,1)}
```

Dictionaries are Mutable

Retrieve a value by specifying its key. Get a definition of a word by looking up the word in a dictionary

```
>>> d = {'red':(1,0,0), 'green':(0,1,0)}
>>> d['red']
1,0,0
```

```
counts = {'red': 103, 'green': 452,
 'orange': 98, 'black': 143}
# access key
red_count = counts['red']

# Dictionaries are Mutable
# reassign key
counts['black'] = 100
# increment key
counts['black'] += 10
```

Python Dictionary Syntax

The keys of the dictionary can be any hashable type (lets just say int, str, tuple)
The values of the dictionary can be any datatype

```
colors = {(1,0,0): 'red',
          (0,1,0): 'green',
          (0,0,1): 'blue',
          (0,0,0): 'black'
          (1,1,1): 'white'}
```

```
numbers = {1: 'one', 652: 'six hundred fifty two'}
```

Dict Attributes

- d.keys() Return a list of keys in the dictionary
- d.values() Return a list of values in the dictionary
- d.items() Return a list of key, value tuples

Caveat. Dictionaries are collections not sequences. You can not count on the order of keys to be same as the way you enter it demo in terminal

Exercise

Count the frequency of words in a string.
Algorithm

- Initialize an empty dictionary
- split the string into a list of words
- for every word in the list of words
  - if word in dictionary, increment count
  - else insert word into dictionary with a value of 1

Dictionary Membership testing

Use the `in` reserve word to test if a dictionary as a given key

```
1 colors = {'red':(1,0,0), ...}
2 if 'red' in colors: ...
```

To test if a value is in the dictionary use .values()

```
1 colors = {'red':(1,0,0), ...}
2 if (1,0,0) in colors.values(): ...
```

Word Count

```
1  text =  'to be or not to be'
2
3  word_count = {}
4  words = text.split(' ')
5  for word in words:
6      if word in word_count:
7          word_count[word]+=1
8      else:
9          word_count[word] = 1
10 print(word_count)
11 # {'not': 1, 'to': 2, 'or': 1, 'be': 2}
```

Exercise

Make a make a phonebook application. What is the key? What is the value?
Make a contacts application. Store phone number, address, email, nickname

Set

- What is a Set?
- A collection of objects in which only ope copy of each object may exist.

```
>>> x = set()  # an empty set
>>> a_set = {1,2,3,4}
>>> a_set
{1,2,3,4}
>>> b_set = {1,1,2,2} # duplicates ignored
>>> b_set
{1,2}
```

Typical Operations on Sets

- len()

```
>>> my_set = {'a','b','c'}
>>> len(my_set}
6
```

- in

```
>>> 'a' in my_set
True
>>> 'z' in my_set
False
```

- for

```
>>> for element in my_set:
        print(element)
a
c
b
```