Day 4:
Python Intro

1 Dictionaries

How to use Dictionaries

- {} unlike lists
- structure: {"key1: value1", "key2: value2"...}
- Access value by using key as the index
- Duplicates not allowed

```
1  #Fruits and their cost
2  fruits_dict = {"apple": 2.99, "banana": 1.50, "pear": 3.70, "apple": 2, "grapes": 2.50}
3
4  print(fruits_dict)
```

```
{'apple': 2, 'banana': 1.5, 'pear': 3.7, 'grapes': 2.5}
```

```
1  #Fruits and their cost
2  fruits_dict = {"apple": 2.99, "banana": 1.50, "pear": 3.70, "grapes": 2.50}
3
4  print(fruits_dict["apple"])
```

```
2.99
```

Key "apple" is used as the index

```python
#Key: fruit name, Value: [number of fruits, cost]
fruits_dict = {"apple": [3, 2.99], "banana": [1, 1.50], "pear": [4, 3.70], "grapes": [13, 2.50]}

print(fruits_dict["banana"])
```

```
[1, 1.5]
```

Dictionary Methods

```
1  #Key: fruit name, Value: [number of fruits, cost]
2  fruits_dict = {"apple": [3, 2.99], "banana": [1, 1.50], "pear": [4, 3.70], "grapes": [13, 2.50]}
3
4  print(fruits_dict.pop("apple"))
5  print(fruits_dict)
```

```
[3, 2.99]
{'banana': [1, 1.5], 'pear': [4, 3.7], 'grapes': [13, 2.5]}
```

```
1  #Key: fruit name, Value: [number of fruits, cost]
2  fruits_dict = {"apple": [3, 2.99], "banana": [1, 1.50], "pear": [4, 3.70], "grapes": [13, 2.50]}
3
4  print(list(fruits_dict.keys()))
5  print(list(fruits_dict.values()))
```

```
['apple', 'banana', 'pear', 'grapes']
[[3, 2.99], [1, 1.5], [4, 3.7], [13, 2.5]]
```

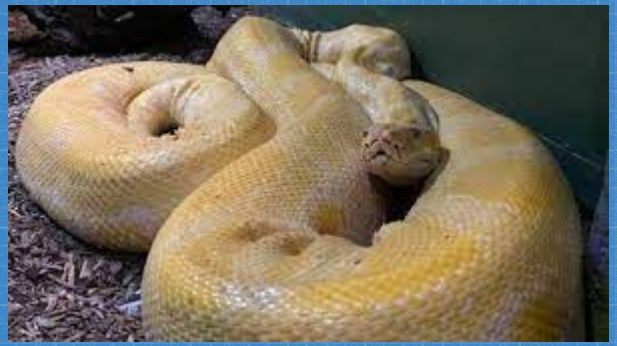
Without list()

```
dict_keys(['apple', 'banana', 'pear', 'grapes'])
```

Try it out

- Make a program using while loops and dictionaries
- The keys of the dictionary should be the items of the shopping list and the value should be the cost

2
Review



Day 1:

- Data types, variables, if/elif/else

Day 2:

- Functions
- Inputs

Day 3:

  - Lists, while/for loops

3

Algorithms Practice

Let’s Practice - Codingbat Warmup 1

- Sum Double
- Makes10
- Front3
- Missing Char
- Pos neg
- Front back

Advanced Problem

- Code a function to find the nth term in the Fibonacci Sequence
- Count vowels/consonants in string
- Using dictionaries, count the occurrence of each word in a given text.

  (Hint: use the .split() function to split a text (string) into a list of words)

Challenge Problem

TEST INPUT:

```
599 23 43
4326 1234 80
704 1776 200
6283 185 31
3141 59 26
```

TEST OUTPUT:

```
1. 218
2. 399
3. 1003
4. 154
5. 126
```

PROBLEM: Construct a Numeral Triangle according to the following rules. You will be given three positive integers: *s*, a starting number; *d*, a delta (the amount by which to increase each number in the triangle); and *r* the number of rows.

1. The first row contains the number *s*.
2. Each of the next rows has one more number than the previous row.
3. Each number in the triangle is *d* more than the previous number in the triangle.
4. Before putting a number in the triangle, it is transformed to a single digit. That is, if the number is more than one digit, replace it by the sum of the digits, repeating until the sum is one digit (for example, 1938 ⇒ 21 ⇒ 3).

Here are two examples of Numeral Triangles:

start=2, delta=3, rows=5	start=221, delta=2, rows=4
2 5 8 2 5 8 2 5 8 2 5 8 2 5 8	5 7 9 2 4 6 8 1 3 5

INPUT: There are 5 lines of data. Each line has 3 positive integers, *s*, *d*, and *r*. The numbers are separated by spaces and each is less than 100,000.

OUTPUT: For each line of data, print the sum of all numbers on the *r*th line of the Numeral Triangle.

SAMPLE INPUT:

```
2 3 5
221 2 4
184 231 35
71 5 27
1 24 100
```

SAMPLE OUTPUT:

```
1. 28
2. 17
3. 140
4. 135
5. 397
```