# ENSF 592: Programming Fundamentals for Data Engineers

Yves Pauchard

## Lecture 6: Dictionaries and Tuples

September 25, 2019

# Agenda

1. Key ideas in Ch 11 and 12

2. Computing list ranks (and Assignment03)

3. Preparation for next lecture

# Notes: Key ideas Dictionaries

- dict() key-value pairs or items
- dict is a hashtable
- `in` operator and `for` work on keys
- direct access a['key'] or with a.get('key', 0) with default value
- dict keys are in any order, use sorted(a_dict) to sort keys.
- raise statement causes an exception
- dict keys need to be hashable (immutable)
- Debugging tips: sanity and consistency checks
- Explore pprint

# Notes: Key ideas Tuples

- tuple(); 10, ; (10,) -> immutable
- Tuple unpacking, works on stings and lists too.
- use tuples to return multiple values from functions
- gather and scatter
- zip and enumerate
- use zip to initialize or update a dict -> dict(zip('abc', range(3)))

# Compute List Ranks

## Problem statement

*With a list of int as input, return a list where at to corresponding position the rank of the value is saved.*

## Motivation: Compute Spearman Rank Correlation

https://en.wikipedia.org/wiki/Spearman%27s_rank_correlation_coefficient#Example

# Compute List Ranks

## Ideas:

- Sort determines rank, 'glue' index to value and sort
- 'Glued' index can then be used to write rank in the proper position

## Python:

- `zip()` can 'glue' together values from different lists
- `sorted()` can sort a list, creating a new one.
- `enumerate()` is a special `zip()` used to setup loops that need access to both index and element
- `list()` converts a `zip` object to a list for printing.

# Compute List Ranks Code

The code is presented in 3 parts on the following slide matching the steps we discussed.

Copy these 3 parts into one python script file to experiment.

# Notes: compute_rank code (part 1)

```
# Reference: https://en.wikipedia.org/wiki/Spearman%27s_rank_correlation_coefficie

iq = [106,86,100,101,99,103,97,113,112,110]
# tv = [7,0,27,50,28,29,20,12,6,17]

print("Original list")
print(iq)


# create a list of value-index pairs
val_idx_list = list(zip(iq, range(len(iq))))

print("value-index list")
print(val_idx_list)
```

# Notes: compute_rank code (part 2)

```
# sort this list
# python will, so we hope, use the first entry in the tuple
# once sorted, we know that first element has rank 0
sorted_val_idx_list = sorted(val_idx_list)

print("sorted value-index list")
print(sorted_val_idx_list)
```

# Notes: compute_rank code (part 3)

```
# create the list of ranks
ranks = [0]*len(iq) # intialize list with zeros
for i, item in enumerate(sorted_val_idx_list):
    # item is (value, index), item[1]-> index,
    # set it to the rank i
    ranks[item[1]] = i

print("Ranks of elements in list")
print(ranks)
```

# Preparation For Next Lab/Lecture

## Read/follow Ch 14 in Think Python 2e