Week 2: Data Structures and Loops

Session 4: Dictionaries and Break/Continue

Understanding Loop Logic using a Table

```
[1]: # Example 1
     sales=[10, 25, 5, 15, 20]
     total=0
     for value in sales:
         total+=value    # shorthand for total=total+value
     total
```

```
75
```

In-class Exercise: Do the following on paper WITHOUT a computer. (Use a table to "walk through" the following code line by line.)

a) What is the final value of `total`?

```
[2]: # Example 2
     sales=[10, 25, 5, 15, 20]
     total=0
     for value in sales:
         if value>=20:
             total+=value
```

b) What are the final values of `maxValue` and `maxIndex`?

```
[4]: # Example 3
     sales=[10, 25, 5, 15, 20]
     maxValue=-1
     maxIndex=-1
     for i in range(len(sales)):
         value=sales[i]
         if value>maxValue:
             maxValue=value
             maxIndex=i
```

Break and Continue

Sometimes you want to modify what is being iterated over in a dynamic way while the program is running:

- `break` can be used to terminate a loop.
- `continue` can be used to skip the rest of the current iteration, and continue to the next iteration.

[6]:
```
for i in [5,4,3,2,1]:
    if i==2:
        break
    print('Count down:',i)
print('Take off!')
```

```
Count down: 5
Count down: 4
Count down: 3
Take off!
```

[7]:
```
for i in [5,4,3,2,1]:
    if i==2:
        continue
    print('Count down:',i)
print('Take off!')
```

```
Count down: 5
Count down: 4
Count down: 3
Count down: 1
Take off!
```

[8]:
```
# Find first week in which sales drops below 10.
sales=[10, 25, 5, 15, 20]
week=0
found=False
for num in sales:
    if num<10:
        found=True
        break
    week+=1
if found:
    print(f'Sales first drops below 10 in Week {week} with {num} units.')
```

```
Sales first drops below 10 in Week 2 with 5 units.
```

Exercise 2.4. Modifying Loop Logic using Break and Continue

Download the Jupyter notebook attached to the Blackboard link for this exercise and submit it at the same link. The notebook contains the following questions.

a) Write a function called `firstOutbreak` with two input arguments:

- **cases**: a list of positive integers corresponding to the number of cases of a certain disease each week. The weeks are labelled starting from 0.
- **threshold:** a positive integer denoting what is an outbreak.

The function should print (not return) one sentence that specifies the first week in which the number of cases is at least equal to the threshold, along with the number of cases that week. If the number of cases never reaches the threshold, the program should print an alternative statement, as in the sample outputs below.

```
[9]: # Write your function here
```

```
[10]: firstOutbreak([10,20,30,15,50],25)
```

```
The first outbreak occured in week 2 with 30 cases.
```

```
[11]: firstOutbreak([10,20,30,15,50],51)
```

```
There is no outbreak in the given data.
```

b) Write a program that asks the user to type a sentence (in lower case without punctuations) and print the same sentence but removing all occurrences of the following words: “a”, “the”, “of”, “in”, “and”, “i”, “you”, “he”, “she”.

Hint: you can split a sentence into a list of words using

```
sentence.split()
```

assuming that the object `sentence` is a string. You can also print words one by one and separate them by spaces using `print(..., end=' ')`.

```
Please enter a sentence: i enjoy doing analytics with Python and R
i enjoy doing analytics with Python R
```

Python Dictionaries

A dictionary represents a mapping between keys and values. It can be used to store labelled data in which the relative order of entries does not matter.

```
[13]: sales={'USA':3000, 'China':2000, 'India':4000}
```

```
[14]: sales['China']
```

```
2000
```

```
[15]: len(sales)
```

```
3
```

```
[16]: # Adding a new entry
      sales['UK']=1000
      sales
```

```
{'USA': 3000, 'China': 2000, 'India': 4000, 'UK': 1000}
```

```
[17]: # Updating entries
      sales['China']=2500
      sales
```

```
{'USA': 3000, 'China': 2500, 'India': 4000, 'UK': 1000}
```

```
[18]: # Checking whether a key exists
      'USA' in sales
```

```
True
```

```
[19]: 'USA' not in sales
```

```
False
```

```
[20]: sales.keys()
```

```
dict_keys(['USA', 'China', 'India', 'UK'])
```

```
[21]: sales.values()
```

```
dict_values([3000, 2500, 4000, 1000])
```

```
[22]: # Checking whether a value exists
      4000 in sales.values()
```

```
True
```

```
[23]: # For loops with dictionaries
      for country in sales:
          print(country,sales[country])
```

```
USA 3000
China 2500
India 4000
UK 1000
```

Exercise 2.5. Practicing Dictionary Syntax

Download the Jupyter notebook attached to the Blackboard link for this exercise and submit it at the same link. The notebook contains the following questions.

a) Create an empty dictionary called "english".

b) Add the keys '0', '1', '2', '3', '4', '5', '6', '7', '8', '9' to this dictionary, and each number should maps to its name in English in lowercase. (i.e. the key '3' should map the the value 'three'). You should do this one key-value pair at a time.

c) Ask the user to input a digit. If the input is one of the keys of the above dictionary, then print "You entered XXX." where XXX is the corresponding value. For any other input, print "I cannot read your input."

```
[26]: # Code for part c)
```

```
Please enter a digit: 3
You entered three.
```

d) Loop through the keys of the dictionary and print the corresponding value, as below.

```
0 is zero.
1 is one.
2 is two.
3 is three.
4 is four.
5 is five.
6 is six.
7 is seven.
8 is eight.
9 is nine.
```

e) Write a program that asks the user to input a phone number and prints every numeric character into the English word, while ignoring all non-numerical characters. You should print the result in one line, separating words with space.

Hint: For loops can be used to iterate through the characters of a string. For example the character c loops through each character of the below phone number. Try it and see.

```
number='213-740-0000'
for c in number:
    print(c)
```

```
Please input a phone number: 213-740-3333
You entered: two one three seven four zero three three three three
```