Lists

- a type of variable in Python

my_list = [1, 2, 3, 4, 5]

- more commonly known as an "array" in other programming languages

Indexing

- a number that represents a position on the list
- indices start at zero (0) and increase in value

```
print(nums[2]) # display the third value in the list
```

- a for loop can be used to access all indices in the list

```
for i in range(5):
    print(nums[i])
```

- can't access indices for elements that don't exist (e.g. nums[100])

Negative Indexing

- starts indexing at the last element in the list and the more negative the index, the more you move backward through the list

```
nums[-1] # the last element in the list
nums[-2] # the 2nd last element in the list
...
```

- can't access indices for elements that don't exist (e.g. nums[-100])

No Limits to Data Types

- a list can contain all the same type of data (e.g. all numbers, all strings, etc.)

- a list can contain all different types of data as well (e.g. one number, one string, one ....)

- a list can contain no data as well

```
nums = []
```

Lists are Mutable

- new data can to added after the list has been created

```
nums.append(6)
nums.insert(3, 8)
```

- data can be removed from the list

```
del nums[3]
nums.remove(5) # finds the first 5 and deletes it
nums.pop() # removes last one
```

- existing data can be changed in the list

```
nums[2] = 12
```

List Functions

len(name_of_list) - counts the items in the list

list.index(val) - returns index number of value

list.sort() - sorts the list itself

min(name_of_list) - returns the lowest value

max(name_of_list) - returns the greatest value

list.reverse() - reverses the list itself

list.count(val) - returns the number of elements that contain the value

list(not_a_list) - converts a non-list into list

Lists and For Loops

- you can 'iterate' through the values in a loop using a standard For loop

```
for n in nums:
    print(n)      # prints out each value in list
```

- the list doesn't have to be in a variable

```
for n in [1, 2, 3, 4, 5]:
    print(n)
```

Lists within Lists

- similar to multidimensional arrays in other languages

```
grid = [[1, 2, 3, 4, 5],[6, 7, 8, 9, 10]]
```

Tuples

- another type of variable, like list, that can contain multiple values

- there are almost no additional functions, like sorting, reversing...

- once a tuple is created, it can't be modified

- it is "immutable"

```
nums = (1, 2, 3, 4, 5)
nums = (1,) # for single numbers
```

Packing/Unpacking Tuples

- "packing" is the process of storing multiple numbers in a single variable

```
nums = (1, 2, 3)
```

- "unpacking" is the process extracting multiple numbers from a single variable into separate variables

```
a, b, c = nums
```

```
a = 1, b = 2, c = 3 # these are the values that are extracted from nums
```

Strings act like immutable lists

- you can use the index number to access a single character

```
print(str[3])
```

- you can use a For loop to access all the elements

```
for c in str:
    print(c)
```

- you can't modify the string

```
str[3] = "x"   # this generates an error
```