Programming Basics with Python

https://www.hydroffice.org/epom

Cheat Sheet

v1

Variables (#001)

Numeric types

```
a = 123  # int
print("The value of a is %d" % a)
b = 45.6789  # float
print("The value of b is %.2f" % b)
```

Numeric operators

```
a + b   # addition
a - b   # subtraction
a * b   # multiplication
a / b   # true division
a // b  # Euclidean (integer) division
a % b   # modulus (remainder)
a ** b  # exponentiation (power)
```

Text type

```
language = "Python"  # str
version = 3  # int
lang_vers = language + " " + str(version)
print(lang_vers)
# or
print("%s %d" % (language, version))
```

Convert text to numeric types

```
t0 = "123"  # str
i0 = int(t0)  # int
t1 = "45.6789"  # str
f1 = float(t1)  # float
```

Lists (#002)

Create a list

```
temp_list = [12.43, 12.9, 13.1, 15.23]
```

Retrieve an item at a specific index

```
first_temp = temp_list[0]
last_temp = temp_list[-1]
```

Loop through each item (#004)

```
for temp in temp_list:
   print(temp)
```

Add/Remove items

```
depth_list = list()
depth_list.append(14.51)
depth_list.append(120.2)
depth_list.append(231.3)
first_depth = depth_list[0]
depth_list.remove(first_depth)
```

Count items

```
nr_of_depths = len(depth_list)
print("Items in list: %d" % nr_of_depths)
```

Conditional Execution (#003)

Boolean Expressions

```
x == 80  # is equal to
x != 80  # is not equal to
x > 80   # is greater than
x >= 80  # is equal or greater than
x < 80   # is less than
x <= 80  # is equal or less than
```

Minimal if condition

```
if temp <= 0:
   print("It is cold!")
```

Full if-elif-else condition

```
if cur_strength < 1.0:
   print("Weak")
elif cur_strength < 5.0:
   print("Moderate")
else:  # any current >= 5.0
   print("Strong")
```

Functions (#005)

Pass arguments and return a value

```
def add(a, b):
   total = a + b
   return total

v0 = 23
v1 = 4
result = add(v0, v1)
print("%d + %d = %d" % (v0, v1, result))
```

Dictionaries (#007)

Create a dictionary and add key/value pairs

```
metadata = dict()
metadata["observer"] = "Darwin"
metadata["vessel"] = "HMS Beagle"
metadata["sea_state"] = 4
```

Access a value

```
print("Collected by %s" % metadata["observer"])
```

Loop through each key/value pair (#004)

```
for key in metadata:
   print("%s -> %s" % (key, metadata[key]))
```

Text Files (#006)

Retrieve paths

```
import os.path

# retrieve absolute path for current folder
cur_folder = os.path.abspath(os.path.curdir)

# create a file path
txt_path = os.path.join(cur_folder, "data.txt")

# check whether the new path exists
if not os.path.exists(txt_path):
   raise RuntimeError("Error for %s" % txt_path)
```

Read a text file

```
txt_file = open(txt_path)
txt_content = txt_file.read()
txt_file.close()
```

Loop through each text row

```
txt_lines = txt_content.splitlines()
for txt_line in txt_lines:
   print(txt_line)
```

Write a text file

```
out_path = os.path.join(cur_folder, "out.txt")
out_file = open(out_path, mode="w")
out_file.write("Coding is easy!")
out_file.close()
```

Classes as Data Containers (#008)

Define a class

```
class TemperatureData:
   """A class for temperature data"""

   def __init__(self):
      self.metadata = dict()
      self.values = list()
```

Instantiate a class

```
td = TemperatureData()
print("Values: %s" % (td.values))
print("Metadata: %s" % (td.metadata))
```

Populate class variables

```
temps_str = "10.3  15.9  3.7  25.0"
temp_list = temps_str.split()
for temp in temp_list:
   td.values.append(float(temp))

td.metadata["observer"] = "Darwin"
td.metadata["vessel"] = "HMS Beagle"
td.metadata["sea_state"] = 4
```