Pre-Engagement Session
Basics of Python

MIT-DSML

Agenda

- Intro to Variables
- Data Structures
- Conditional Statements
- Looping Statements

Intro to Variables

A **variable** is an object in Python that can store a **value**

You can create a variable by assigning a value to it with the “=” operator, like: **X** = **5**

This value can be of different data types, like: **you can check the type using type()*

- A Number - an **Integer** (5) or a **floating** value (5.46) or a **string** (“abc”) or a **boolean** (True)
- A collection of numbers, strings, or boolean values - a **list** like [5, 5.45, “abc”]
- A dictionary with **keys** and **values** like:

```
d = {
        “names”  :  [“x”, “y”, “z”]
        “height”  :  [156.8, 160.7, 145.2]
        “weight” :  [80, 76, 100]
}
```

Data Structures

A data structure is an object in Python that can hold a collection of values.

- List is a data structure which can be created with “,” comma separated values enclosed by square brackets

  lst = [“x”, “y”, “z”, “w”, 2, 10.1, False]

- Tuple is data structure similar to lists but unlike lists, it is an immutable data structure. The values are enclosed by round brackets.

  tup=(“x”, “y”, “z”, “w”, 2, 10.1, False)

- A set is an unordered collection of data type that is mutable, has unique entries and is iterable. The values are enclosed by curly brackets

  set_1 = {3,4,7,7,5} would give output {3, 4, 5, 7}

- A dictionary is a collection of pairs of **keys** and **values**:

```
d = {
        “names”  : [“x”, “y”, “z”]
        “height” : [156.8, 160.7, 145.2]
        “weight” : [80, 76, 100]
}
```

Conditional Statements

Logic is all about making **decisions** based on **rules**. The first and simplest construct is **if else**:

```
if (test expression):
    <execute statements> this is executed if the expression is True
else:
    <execute statements> this is executed if the expression is False
```

If you have more than one decision to make, you can use the **elif** construct:

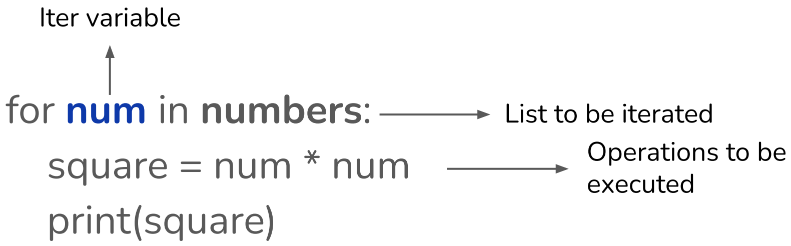
```
if (test expression - 1):
    <execute statements> run this if test expression - 1 is True
elif (test expression - 2):
    <execute statements> run this if test expression - 2 is True
else:
    <execute statements> run this if none of the above expressions are True
```

Looping Statements

When you want to repeatedly run an operation, you need to ask 2 questions:

1. What do you want to repeatedly run operations on? A list or a tuple or a dictionary, etc.?
2. What operation do you want to execute on each element of that list or tuple or dictionary?

Let’s take an example of squaring the elements in a list: numbers = [1, 2, 3, 4, 5]



Iter variable

```
for num in numbers:
    square = num * num
    print(square)
```

List to be iterated

Operations to be executed

Great Learning

Happy Learning !