Python Data Structures: Range:

Programming for Data Science with Python

1. Overview

In Python, ranges are the objects of the class range that has the constructor range().

Range is an immutable sequence data type/structure, i.e., its contents can be changed after being created.

The range type:

- Represent an immutable sequence of numbers
- Is commonly used for looping a specific number of times in for loops.

Run the following code block:

```
In [1]: range(10)
```

```
Out[1]: range(0, 10)
```

1.1 Properties of ranges

The advantage of the range type over a regular list or tuple:

- A ***range*** object always takes the ***same (small) amount of memory*** (no matter the size of the range it represents because it only stores the start, stop, and step values).

1.2 Constructors

1.2.1 Constructor: range(stop)

1.2.2 Constructor: range (start, stop, [step])

The arguments to the range constructor must be integers:

- Either built-in int or any object that implements the index special method.
- If the step argument is omitted, it defaults to 1.
- If the start argument is omitted, it defaults to 0.
- If step is zero, ValueError is raised.

For a positive step, the contents of a range r are determined by the formula:

– ***r[i]=start+step*I where i>=0 and r[i] < stop***

- Start: The value of the start parameter (or 0 if the parameter was not supplied)
- Stop: The value of the stop parameter.
- Step: The value of the step parameter (or 1 if the parameter was not supplied).

2. Examples:

Using range() in creating other sequence objects

Run the following 7 code blocks:

In [2]:
```
list(range(10))
```

Out[2]:
```
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
```

In [8]:
```
list(range(1, 11))
```

Out[8]:
```
[1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
```

In [4]:
```
list(range(0, 30, 5))
```

Out[4]: [0, 5, 10, 15, 20, 25]

In [5]:
```
list(range(0, 10, 3))
```

Out[5]: [0, 3, 6, 9]

In [6]:
```
list(range(0, -10, -1))
```

Out[6]: [0, -1, -2, -3, -4, -5, -6, -7, -8, -9]

In [7]:
```
list(range(0))
```

Out[7]: []

In [8]:
```
list(range(1, 0))
```

Out[8]: []

In [ ]:
```

```