Cheat Sheet

Lists and Strings

Splitting

str_var.split(separator)

Splits a string into a list at every specified separator.
If no separator is specified, default separator is whitespace.

Code

PYTHON

```python
nums = "1 2 3 4"
num_list = nums.split()
print(num_list)
```

Output

```
['1', '2', '3', '4']
```

Multiple WhiteSpaces

Multiple whitespaces are considered as single when splitting.

Code

PYTHON

```python
nums = "1  2 3 4 "
num_list = nums.split()
print(num_list)
```

Output

```
['1', '2', '3', '4']
```

New line

`\n` and tab space `\t` are also whitespace.

Code

PYTHON

```
1  nums = "1\n2\t3 4"
2  num_list = nums.split()
3  print(num_list)
```

Output

```
['1', '2', '3', '4']
```

Using Separator

Breaks up a string at the specified separator.

Example -1

Code

PYTHON

```
1  nums = "1,2,3,4"
2  num_list = nums.split(',')
3  print(num_list)
```

Output

```
['1', '2', '3', '4']
```

Example -2

Code

PYTHON

```
1  nums = "1,2,,3,4,"
2  num_list = nums.split(',')
3  print(num_list)
```

Output

```
['1', '2', '', '3', '4', '']
```

Space as Separator

Code

PYTHON

```
1  nums = "1  2 3 4 "
2  num_list = nums.split(" ")
3  print(num_list)
```

Output

```
['1', '', '2', '3', '4', '']
```

String as Separator

Example - 1

Code

PYTHON

```python
string_a = "Python is a programming language"
list_a = string_a.split('a')
print(list_a)
```

Output

```
['Python is ', ' progr', 'mming l', 'ngu', 'ge']
```

Example - 2

PYTHON

```python
string_a = "step-by-step execution of code"
list_a = string_a.split('step')
print(list_a)
```

Output

```
['', '-by-', ' execution of code']
```

Joining

str.join(sequence)

Takes all the items in a sequence of strings and joins them into one string.

Code

PYTHON

```python
list_a = ['Python is ', ' progr', 'mming l', 'ngu', 'ge']
string_a = "a".join(list_a)
print(string_a)
```

Output

```
Python is a programming language
```

Joining Non String Values

Sequence should not contain any non-string values.

Code

PYTHON

```python
list_a = list(range(4))
string_a = ",".join(list_a)
print(string_a)
```

Output

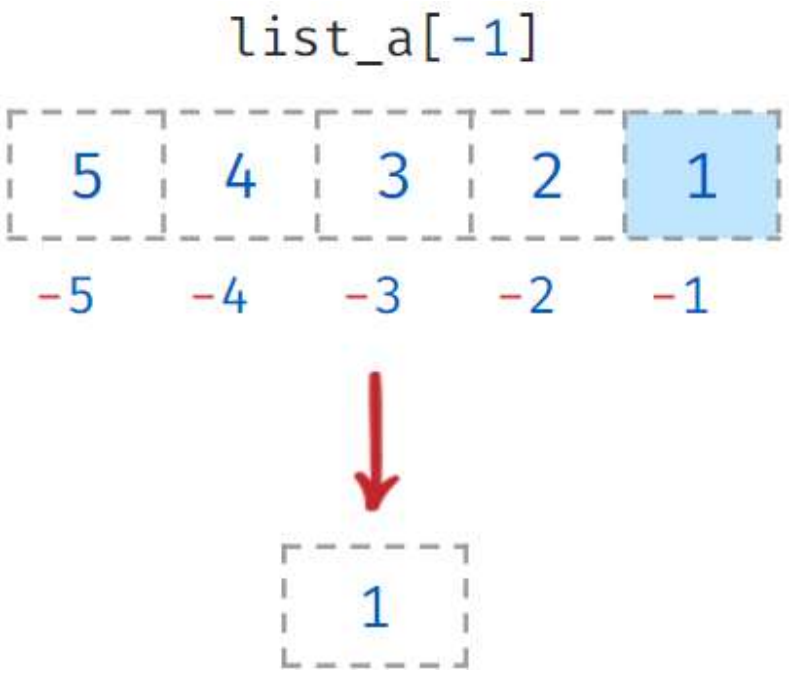
```
TypeError: sequence item 0: expected str instance, int found
```

Negative Indexing

Using a negative index returns the nth item from the end of list.

Last item in the list can be accessed with index

-1



Reversing a List

-1 for step will reverse the order of items in the list.

Code

PYTHON

```
list_a = [5, 4, 3, 2, 1]
list_b = list_a[::-1]
print(list_b)
```

Output

```
[1, 2, 3, 4, 5]
```

Accessing List Items

Example-1

Code

PYTHON

```
list_a = [5, 4, 3, 2, 1]
item = list_a[-1]
```

```
3 print(item)
```

Output

```
1
```

Example-2

Code

PYTHON

```
1 list_a = [5, 4, 3, 2, 1]
2 item = list_a[-4]
3 print(item)
```

Output

```
4
```

Slicing With Negative Index

You can also specify negative indices while slicing a List.

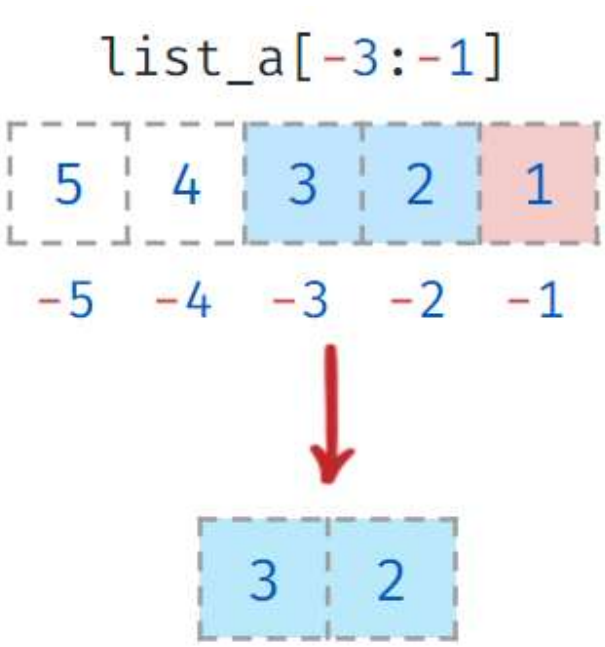
Code

PYTHON

```
1 list_a = [5, 4, 3, 2, 1]
2 list_b = list_a[-3:-1]
3 print(list_b)
```

Output

```
[3, 2]
```



Out of Bounds Index

While slicing, Index can go out of bounds.

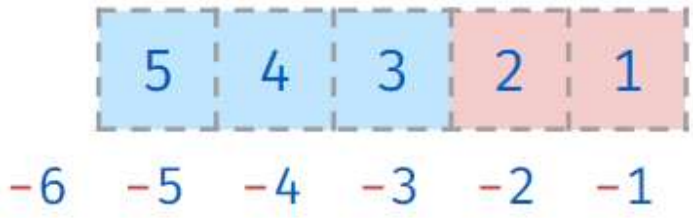
Code

PYTHON

```
list_a = [5, 4, 3, 2, 1]
list_b = list_a[-6:-2]
print(list_b)
```

Output

```
[5, 4, 3]
```



Negative Step Size

variable[start:end:negative_step]

Negative Step determines the decrement between each index for slicing.

Start index should be greater than the end index in this case

- start > end

Negative Step Size Examples

Example - 1

Code

PYTHON

```python
list_a = [5, 4, 3, 2, 1]
list_b = list_a[4:2:-1]
print(list_b)
```

Output

```
[1, 2]
```

Example - 2

Negative step requires the start to be greater than end.

Code

PYTHON

```python
list_a = [5, 4, 3, 2, 1]
list_b = list_a[2:4:-1]
print(list_b)
```

Output

```
[]
```

Reversing a List

-1 for step will reverse the order of items in the list.

PYTHON

```
1  list_a = [5, 4, 3, 2, 1]
2  list_b = list_a[::-1]
3  print(list_b)
```

Output

```
[1, 2, 3, 4, 5]
```

Reversing a String

-1 for step will reverse the order of the characters.

Code

PYTHON

```
1  string_1 = "Program"
2  string_2 = string_1[::-1]
3  print(string_2)
```

Output

```
margorP
```

Negative Step Size - Strings

Code

PYTHON

```python
string_1 = "Program"
string_2 = string_1[6:0:-2]
print(string_2)
```

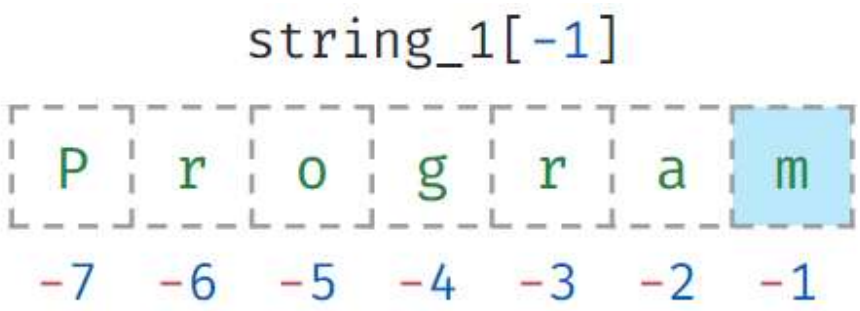
Output

```
mro
```

Indexing & Slicing - Strings

Example - 1

Code



PYTHON

```python
string_1 = "Program"
string_2 = string_1[-1]
print(string_2)
```

Output

```
m
```

Example - 2

Code

PYTHON

```python
string_1 = "Program"
string_2 = string_1[-4:-1]
print(string_2)
```

Output

```
gra
```

Submit Feedback