```
S= "Spam"
S.isalpha()
```

```
True
```

```
S.isnumeric()
```

```
False
```

Getting Help

```
help(S.replace)
```

```
Help on built-in function replace:

    replace(old, new, count=-1, /) method of builtins.str instance
        Return a copy with all occurrences of substring old replaced by new.

          count
            Maximum number of occurrences to replace.
            -1 (the default value) means replace all occurrences.

        If the optional argument count is given, only the first count occurrences are
        replaced.
```

```
dir(S)
```

```
['__add__',
 '__class__',
 '__contains__',
 '__delattr__',
 '__dir__',
 '__doc__',
 '__eq__',
 '__format__',
 '__ge__',
 '__getattribute__',
 '__getitem__',
 '__getnewargs__',
 '__gt__',
 '__hash__',
 '__init__',
 '__init_subclass__',
 '__iter__',
 '__le__',
 '__len__',
 '__lt__',
 '__mod__',
 '__mul__',
 '__ne__',
 '__new__',
 '__reduce__',
```

```
line = 'aaa,bbb,ccccc,dd\n\n\n' #adding space in a string
print(line)
```

```
aaa,bbb,ccccc,dd
```

```
line = line.rstrip() # Remove whitespace characters on the right side
print(line)
```

```
aaa,bbb,ccccc,dd
```

```
line = 'aaa,bbb,ccccc,dd   ' #adding space in a string
print(line)
```

```
aaa,bbb,ccccc,dd
```

```
line = 'aaa,bbb,ccccc,dd\n\n\n' #adding space in a string
print(line)
```

```
aaa,bbb,ccccc,dd
```

```
S = 'A\nB\tC' # \n is end-of-line, \t is tab
print(S)
print(len(S)) # Each stands for just one character
```

```
A
B	C
5
```

```
ord('\t')
```

```
9
```

```
S = 'A\0B\0C' # \0, a binary zero byte, does not terminate string
print(S)
print(len(S))
```

```
A B C
5
```

3. Lists

positionally ordered collections of arbitrarily typed objects

```
L
```

```
[123, 'spam', 1.23]
```

```
len(L)
```

```
3
```

```
L[0] # Indexing by position
```

```
123
```

```
L[:-1] # Slicing a list returns a new list
```

```
[123, 'spam']
```

+ Code + Text

```
L + [4, 5, 6] # Concatenation makes a new list too
```

```
[123, 'spam', 1.23, 4, 5, 6]
```

```
L # We're not changing the original list
```

```
[123, 'spam', 1.23]
```

```
L.append('NI') # Growing: add object at end of list
```

```
L
```

```
[123, 'spam', 1.23, 'NI']
```

```
L = [123, 'spam', 1.23]
L.pop('spam') # Shrinking: delete an item in the middle
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-27-d8808010d678> in <cell line: 2>()
      1 L = [123, 'spam', 1.23]
----> 2 L.pop('spam') # Shrinking: delete an item in the middle

TypeError: 'str' object cannot be interpreted as an integer
```

```
L
```

```
[123, 'spam', 1.23]
```

```
M = ['bb', 'aa','Aa', 'cc']
```

```
L
```

```
[123, 'spam', 1.23, 'NI']
```

```
L = [123, 'spam', 1.23]
L.pop('spam') # Shrinking: delete an item in the middle
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-27-d8808010d678> in <cell line: 2>()
      1 L = [123, 'spam', 1.23]
----> 2 L.pop('spam') # Shrinking: delete an item in the middle

TypeError: 'str' object cannot be interpreted as an integer
```

```
L
```

```
[123, 'spam', 1.23]
```

```
M = ['bb', 'aa','Aa', 'cc']
```

```
M.sort()
```

```
print(M)
```

```
['Aa', 'aa', 'bb', 'cc']
```

```
M.reverse()
```

```
M
```

```
['Aa', 'aa', 'bb', 'cc']
```

```
L
```

```
[123, 'spam', 1.23]
```

```
L[99] #Checking bounds
```

```
---------------------------------------------------------------------------
IndexError                                Traceback (most recent call last)
<ipython-input-39-356ad8e766c0> in <cell line: 1>()
----> 1 L[99] #Checking bounds

IndexError: list index out of range
```

```
L[99]=1
```

```
L[99]=1
```

```
---------------------------------------------------------------------------
IndexError                                Traceback (most recent call last)
<ipython-input-40-29cb710440ca> in <cell line: 1>()
----> 1 L[99]=1

IndexError: list assignment index out of range
```

```
#Nesting

M = [[1, 2, 3], # A 3 × 3 matrix, as nested lists
 [4, 5, 6],
 [7, 8, 9]]
```

```
M
```

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

```
M[1]
```

```
[4, 5, 6]
```

```
M[1][2]
```

```
6
```