Python Lambda

A lambda function is a small anonymous function.

A lambda function can take any number of arguments, but can only have one expression.

Syntax

lambda arguments : expression

- The expression is executed and the result is returned:

In [1]:
```
# Example
# Add 10 to argument a, and return the result:

x = lambda a : a + 10
print(x(5))
```

```
15
```

Lambda functions can take any number of arguments:

In [2]:
```
# Example
# Multiply argument a with argument b and return the result:

x = lambda a, b : a * b
print(x(5, 6))
```

```
30
```

In [3]:
```
# Example
# Summarize argument a, b, and c and return the result:

x = lambda a, b, c : a + b + c

print(x(5, 6, 2))
```

```
13
```

Why Use Lambda Functions?

The power of lambda is better shown when you use them as an anonymous function inside another function.

Say you have a function definition that takes one argument, and that argument will be multiplied with an unknown number:

In [4]:
```
def myfunc(n):
    return lambda a : a * n
```

In [5]:
```
# Use that function definition to make a function that always doubles
# the number you send in:

# Example
def myfunc(n):
    return lambda a : a * n

mydoubler = myfunc(2)

print(mydoubler(11))
```

```
22
```

Or, use the same function definition to make a function that always triples the number you send in:

In [ ]:
```
# Example
def myfunc(n):
    return lambda a : a * n

mytripler = myfunc(3)

print(mytripler(11))
```

Or, use the same function definition to make both functions, in the same program:

In [6]:
```
# Example
def myfunc(n):
    return lambda a : a * n

mydoubler = myfunc(2)
mytripler = myfunc(3)

print(mydoubler(11))
print(mytripler(11))
```

```
22
33
```

In [ ]:
```

```