Anonymous functions (lambdas)

Anonymous functions (lambdas)

In Python, an anonymous function is a function that is defined without a name

Anonymous functions (lambdas)

```
lambda arguments: expression
```

Anonymous functions (lambdas)

```
lambda *args, **kwargs: print(args, kwargs)

# Smth wrong
>>> a = lambda *args, **kwargs: print(args, kwargs)
>>> a()
() {}

>>> a(1, 2, 3, k=1)
(1, 2, 3) {'k': 1}
```

Anonymous functions (lambdas)

```
>>> lst = [{'key': 20}, {'key': 1}, {'key': 11}]
>>> sorted(lst)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: '<' not supported between instances of 'dict' and
'dict'

'<' not supported between instances of 'dict' and 'dict'
```

Anonymous functions (lambdas)

```
>>> lst = [{'key': 20}, {'key': 1}, {'key': 11}]
>>> sorted(lst, key=lambda x: x['key'])
[{'key': 1}, {'key': 11}, {'key': 20}]
```

filter

```
>>> lst = [1, 2, 3, 4, 5]

>>> even_filter = filter(lambda x: not (x % 2), lst)

>>> even_filter
<filter object at 0x7fcac81bd7f0>


>>> list(even_filter)
[2, 4]
```

filter

The function is called with **all** the items in the list and a new list is returned which contains items for which the function evaluates to **True**

map

```
>>> lst = ['1', '2', '3', '4', '5']

>>> int_lst = map(lambda x: int(x), lst)

>>> int_lst
<map object at 0x7fcac8b68e80>


>>> list(int_lst)
[1, 2, 3, 4, 5]
```

map

The function is called with all the items in the list and a new list is returned which contains items returned by that function for each item