Functional Programming

Lambda Operator

- define simple auxiliary functions

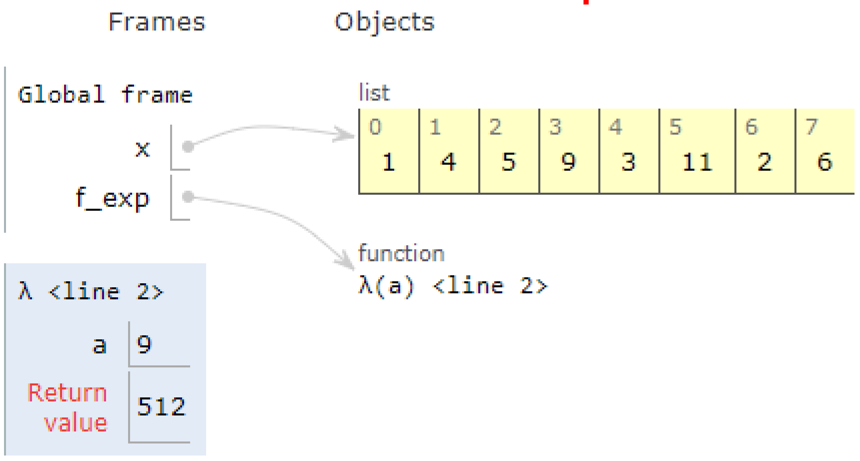
lambda argument_list: expression

```
>>> f_max = lambda a, b: a if (a>b) else b
>>> f_sum = lambda a, b: a+b
>>> f_even = lambda a: (a % 2 ==0)
>>> f_even = lambda a: (a % 2 ==0)
```

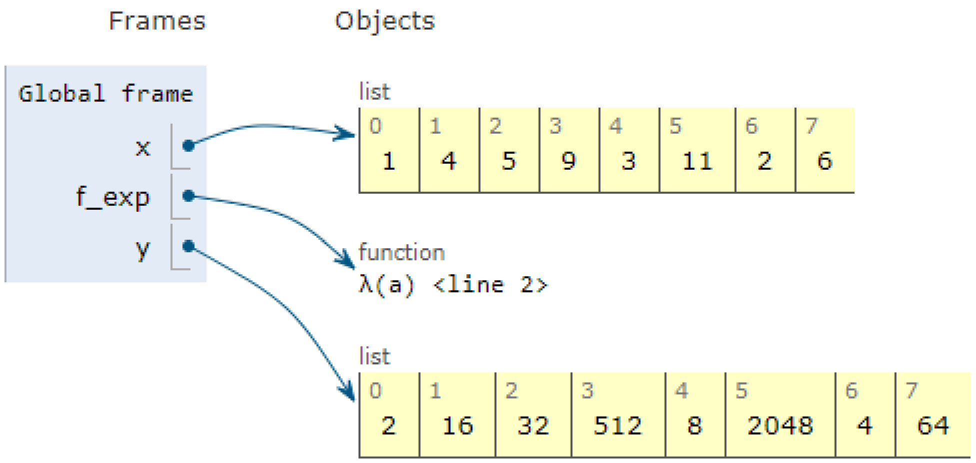
map(function, sequence)

- apply call function for each item

```
>>> x = [1, 4, 5, 9, 3, 11, 2, 6]
>>> f_exp = lambda a: 2**a
>>> y = list(map(f_exp, x))
>>> # intermediate step
```



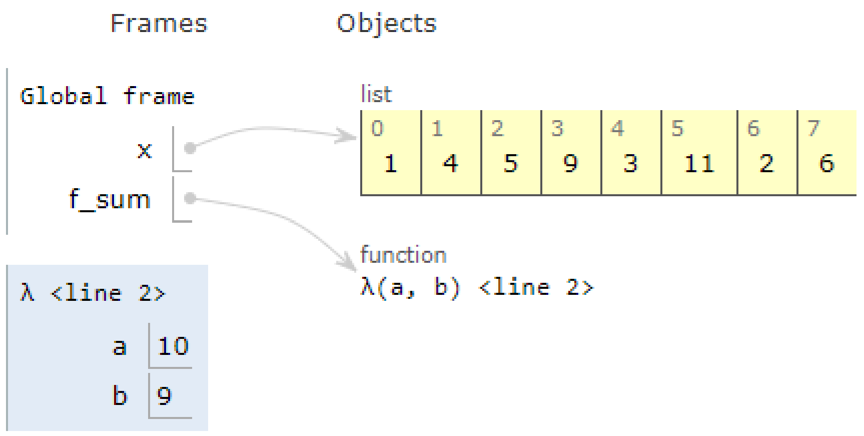
```
>>> # final result
```



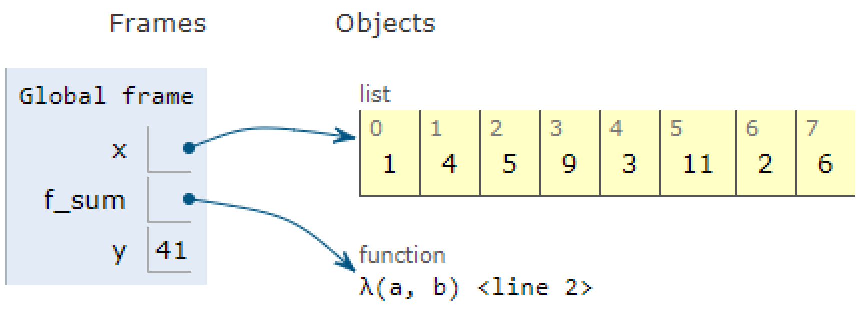
reduce(function, sequence)

- calls binary function on first two items
- iterate on result and next item

```
>>> x = [1, 4, 5, 9, 3, 11, 2, 6]
>>> f_sum = lambda a, b: a+b
>>> y = reduce(f_sum, x)
>>> # intermediate step
```

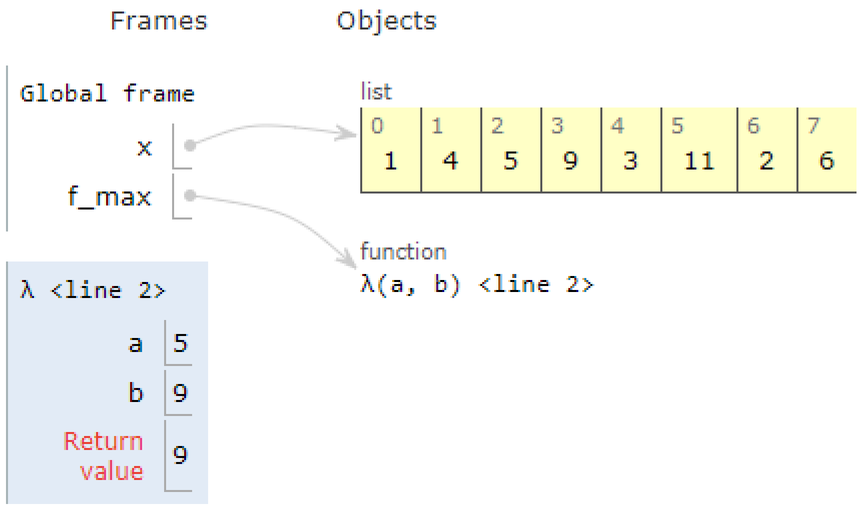


```
>>> # final result
```

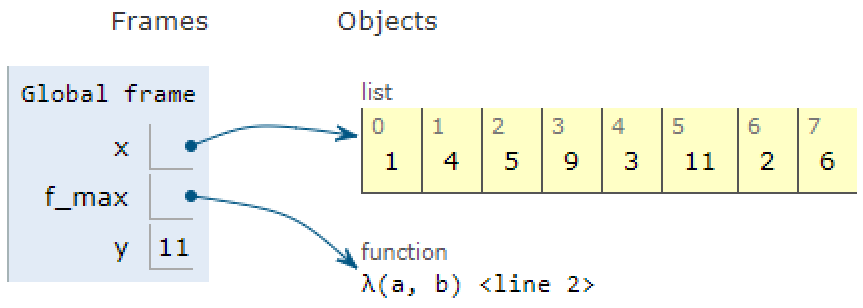


Example: Computing Max Value in List with Reduce

```
>>> x = [1, 4, 5, 9, 3, 11, 2, 6]
>>> f_max = lambda a, b: a if (a>b) else b
>>> y = reduce(f_max, x)
>>> # intermediate step
```



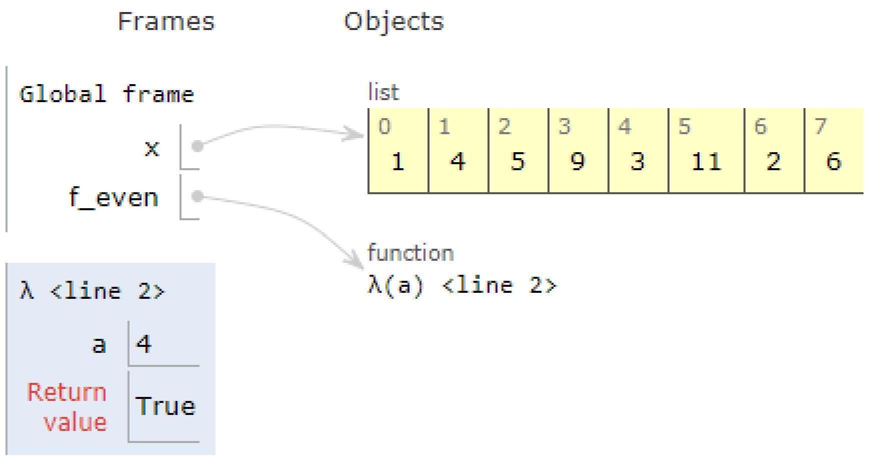
```
>>> # final result
```



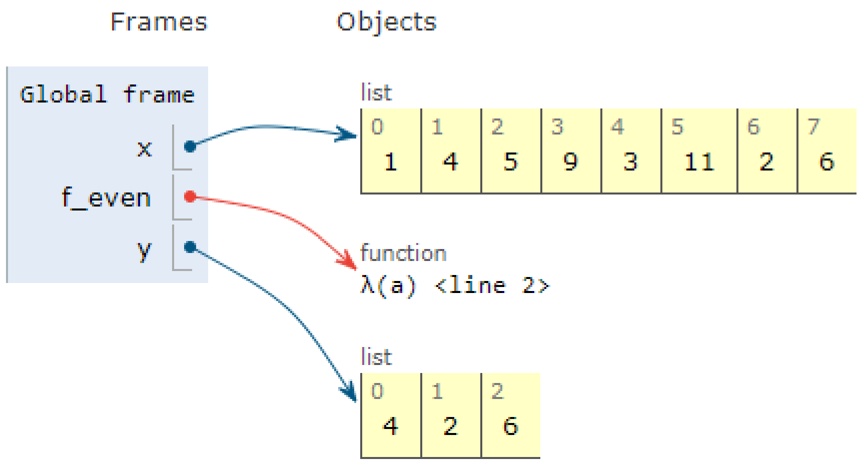
filter(function, sequence)

- returns items for which function is true

```
>>> x = [1, 4, 5, 9, 3, 11, 2, 6]
>>> f_even = lambda a: (a % 2 ==0)
>>> y = list(filter(f_even, x))
>>> # intermediate step
```



```
>>> # final result
```



Review Problems

Interview Problem

- what is *map*?

Interview Problem

- what is a lambda function?
- what does it do?

Interview Problem

- differences between the *lambda* and *def*

Interview Problem

- name methods used to implement functionally oriented programming

Interview Problem

- what do we mean when we say that a certain lambda expression forms a closure?

Interview Problem

- what is the l*ambda* operator?

Interview Problem

- give an example of filter and reduce over an iterable object