CS 30 Discussion 1A

2020.11.06

Map

- **map** is a function takes two arguments:

  the first argument is the **function `f` to use to transform each list element**

  the second argument is **the list `l` to transform**
- **map** always returns a transformed list `l’`
- `len(l) = len(l’)`
- order of elements in `l` is the same as `l’` but values can be different
- The return value of a function `f` should match the type of element `e`
- `list(map(f, [e1,e2,...,en])) = [f(e1), f(e2), ..., f(en)]`

Map practice

Write a function which will takes a list of strings, e.g. ['apple', 'banana', 'cherry'] and returns the number of character ‘a’ in the string, i.e. [1, 3, 0]

(Hint: you need recursion and map)

```
def count_a(s):
    if s == '': return 0
    else:
        if s[0] == 'a':
                return 1 + count_a(s[1:])
        else:
                return count_a(s[1:])

def num_of_a(x):
    return list(map(count_a, x))
```

Lambda function

- To avoid writing small function definitions, we can use lambdas
- Lambda is a keyword that means that we're defining an **anonymous function**.
- The function body is **a single expression**, whose value is implicitly returned as the result of the function.
- Often used in high order function like **map and filter**.

```
lambda x: x**2 + 2*x - 5
```

Lambda practice

Suppose you want to square each element in the list. How would you do that?

```
[1, 2, 3, 4, 5] - > [1, 4, 9, 16, 25]
```

Three solutions

1. using helper function:

```
def multiply(x):
    return (x*x)
squared = list(map(multiply, items))
```

2. Or using recursion:

```
def square(l):
    if l == []:
        return []
    return [l[0] * l[0]] + square(l[1:])
```

3. Or using lambda:

```
squared = list(map(lambda x: x*x, items))
```

Trace Map

1. What is the result of:

`list(map(len,[[1],[2],[3]]))?`

a. 1
b. [1,1,1]
c. [1,2,3]
d. [[1] ,[1], [1]]
e. [[1] ,[2], [3]]
f. error

2. What is the result of:

`list(map(len,[1,2,3]))?`

a. 1
b. [1,1,1]
c. [1,2,3]
d. [[1] ,[1], [1]]
e. [[1] ,[2], [3]]
f. error

3. What is the result of:

`list(map(len,['1','2','3']))?`

a. 1
b. [1,1,1]
c. [1,2,3]
d. [[1] ,[1], [1]]
e. [[1] ,[2], [3]]
f. error

Filter

- **filter** is a function takes two arguments:
  - the first argument is the **function `f` to use to transform each list element**
  - the second argument is **the list `l` to transform**
- **filter** always returns a transformed list `l’`
- `len(l’) <= len(l)`
- order of elements in the smaller l’ is preserved
- The return value of a function `f` is always boolean (True/False)
- `list(filter(f, l)) returns a list of all elements e of l such that f(e) = True`

Filter

- You can think of filter as a sieving process. Apply function `f` to each of the element `e` in the list `l`, filter out those element who do not obey the rules in function

Trace Filter

1. What is the result of:

`list(filter(lambda x: x % 13 == 0, [1,13,39,40,21]))`?

```
a.  13
b.  [13,39]
c.  [0, 13, 29, 0, 0]
d.  [ ]
e.  [1,13,39,40,21]
f.  error
```

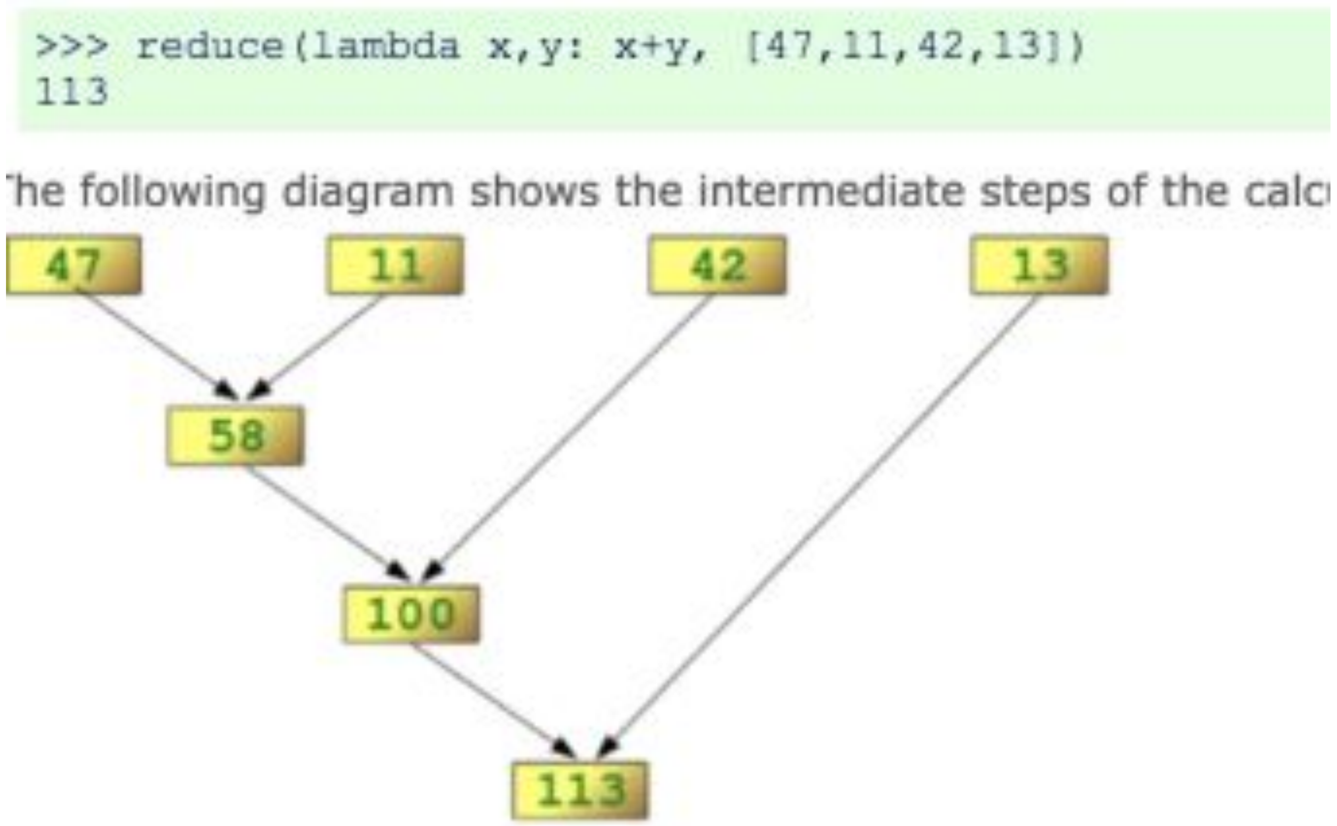
2. What is the result of is impossible with filter given you operate on the initial list `l =[1,3,10,14,15]`?

```
a.  13
b.  [13]
c.  [0,3,10,0,0]
d.  []
e.  [1,13,39,40,21]
f.  [2,6,20,24,30]
```

Reduce Review

- **reduce** is a function takes two arguments:
  - the first argument is the **function `f` that takes two arguments where the first argument is the result of reducing the list so far and the second argument is the element of the list**
  - the second argument is **the list `l` to reduce**
- **reduce** always returns a single value of the same type as the list elements
- To use reduce you need to include the following on top of your python file:
- `from functools import reduce`
- `reduce(f, [x1,x2,x3]) performs the computation f(f(x1, x2), x3)`

Reduce



```
>>> reduce(lambda x,y: x+y, [47,11,42,13])
113
```

he following diagram shows the intermediate steps of the calc

Trace Reduce

1. What is the result of:

`reduce(lambda a,b: a*b,[1,2,3])?`

a. 6
b. [6]
c. [1]
d. []
e. 1
f. error

2. What is the result of:

`reduce(lambda a,b: a if len(a) > len(b) else b,['pen','pineapple','apple','pen'])?`

a. apple
b. pen
c. pineapple
d. [pen]
e. [apple]
f. error

Problem Set

Please work on the problem set question 1-5.