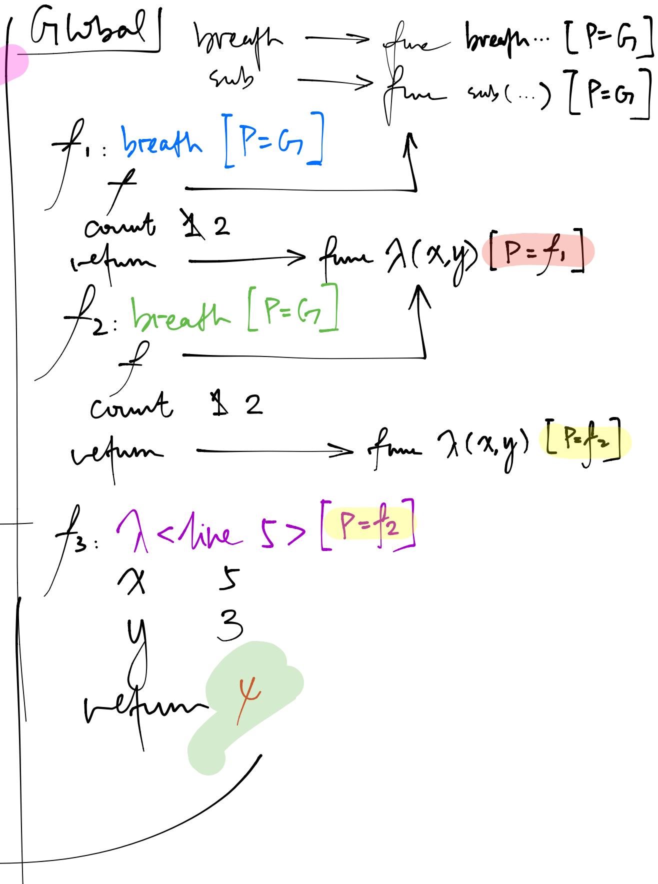
Q1-3.

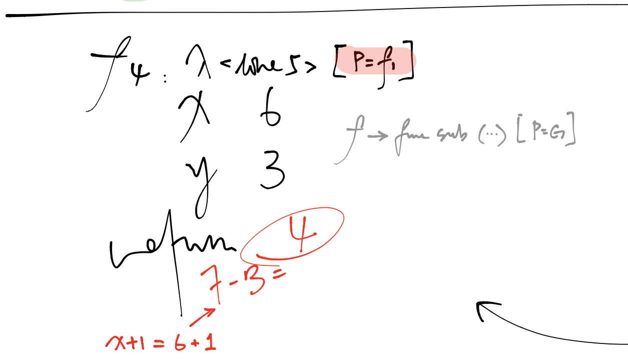
```
1 def breath(f, count=1):
2     if count > 1:
3         print(count)
4     count += 1
5     return λ x,y: f(x+1, y).
```

→ breath(breath(sub))(5,3)

→ 4



Global
breath → func breath... [P=G]
sub → func sub(...) [P=G]
f1: breath [P=G]
f
count ~~1~~ 2
return → func λ(x,y) [P=f1]
f2: breath [P=G]
f
count ~~1~~ 2
return → func λ(x,y) [P=f2]
f3: λ <line 5> [P=f2]
x 5
y 3
return 4



Q3-(c)

Key:

- smallest:
  min of past digits
  that couldn't be surpassed.
- d:
  Digit to ignore.
  i.e. last digit
- Going backwards
  ↓
- Tree Recursion.
  Take the curr last digit
  or not.

```
def near(n, smallest=10, d=10)
    if n==0:
        return 0
    no = near(n//10, smallest, d)
    if smallest > n%10:
        yes = 10*near(n//10, min(smallest, d), n%10) + n%10
        return max(yes, no)
    return no.
```