#C-3.40
logb f(n) = log f(n)/logb;
logb is a constant so the core of the function is log f(n), which is Θlogf(n)

#C-3.50
Something like this…

a) list of len i with all values x
```
      for i in n:
      for j<i j++:
       list[i]*=x
      sum(list)
```

b) list of len i with all values x
```
      for i in n:
       while i>1:
            if n%2:
                  list[i]*=list[i]
                  i=i/2
            else:
                  list[i]*=x
                  i-=1
      sum(list)
```
c) O(n)

#R-3.17
Show that (n+1)^5 is O(n^5)
The expanded form of (n+1)^5 would have a highest degree of n^5 and as n gets very big the other terms become relatively insignificant. As n becomes very big, there exist positive constant c that makes cn^5 greater than n^5.

#C-4.20
```
def sort(s,k):
      if s[0]<=k:
            return s[0]+sort(s[1:],k)
      else:
            return sort(s[1:],k)+s[0]
```
Θ(n)

#C-4.10
find log n using division and addition
```
def flog(n):
      if n == 1:
            return 0
      else:
            return flog(n//2)+1
```