Problem 4-1.

a: 3

b: 4

c: 5

d: In practical, m gradually increasing, but k is a constant. We don't know how to choose k. If k is small, when m becomes big and increase fast, a k-size bigger resize will be useless and frequent. If k is big, when m is small and seldom increases, $k-m$ memory will be wasted.

In theoretical, if we insert n elements, the total cost will be $O(1+k+2k+\ddots+n) = O(n^2)$ and the amortized cost will be $O(n)$, which is bad.

Problem 4-2. a: 1

b: 2

Problem 4-3. a: 6

h: 5

i: 4

j: 1 Correct:3 Reason:extra bits is the question

k: 2

l: 3

m: 4

n: 4

o: True,True,True,True,True,False Correct:False,False,True,True,True,False

p: 3

q: 2

r: 3

s: 6 Correct:8 Reason:the subtree has O(log(n)) height

t: 8

Problem 3-1.

a: intersects

b: 187590314

c: True, False, False, False, True Correct:True, True, False, False, True

d: 1

e: 2

f: 4

g: 3

h: 2

i: 2

j: list

k: 20000