↳Add **Question 5.3**

✚ Add **Question 6**

Save

Q1 How much time?

5 Points

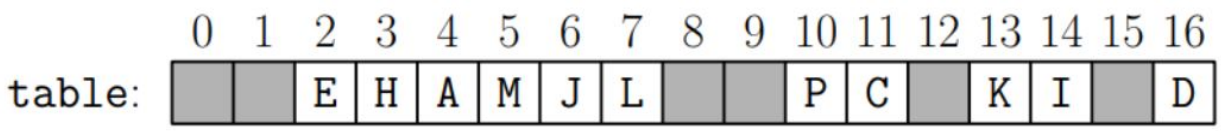
What benefits come from hash tables?

- [x] O(1) amortized insertion
- [ ] O(n) amortized deletion
- [x] O(1) amortized search
- [ ] O(n) average time deletion

Q2 The good old Insertions

30 Points

Show the result of inserting the key “V” into the hash table shown below, assuming open addressing and using each of the following collision resolution methods. Assume that `h("V") == 2`. For all problems be sure to indicate which cells were probed. Do not worry about resizing.



Q2.1 Linear Probing

15 Points

Use linear probing

No files uploaded

Q2.2 Quadratic Probing
15 Points

Use quadratic probing

No files uploaded

Q3 This one will be tough
25 Points

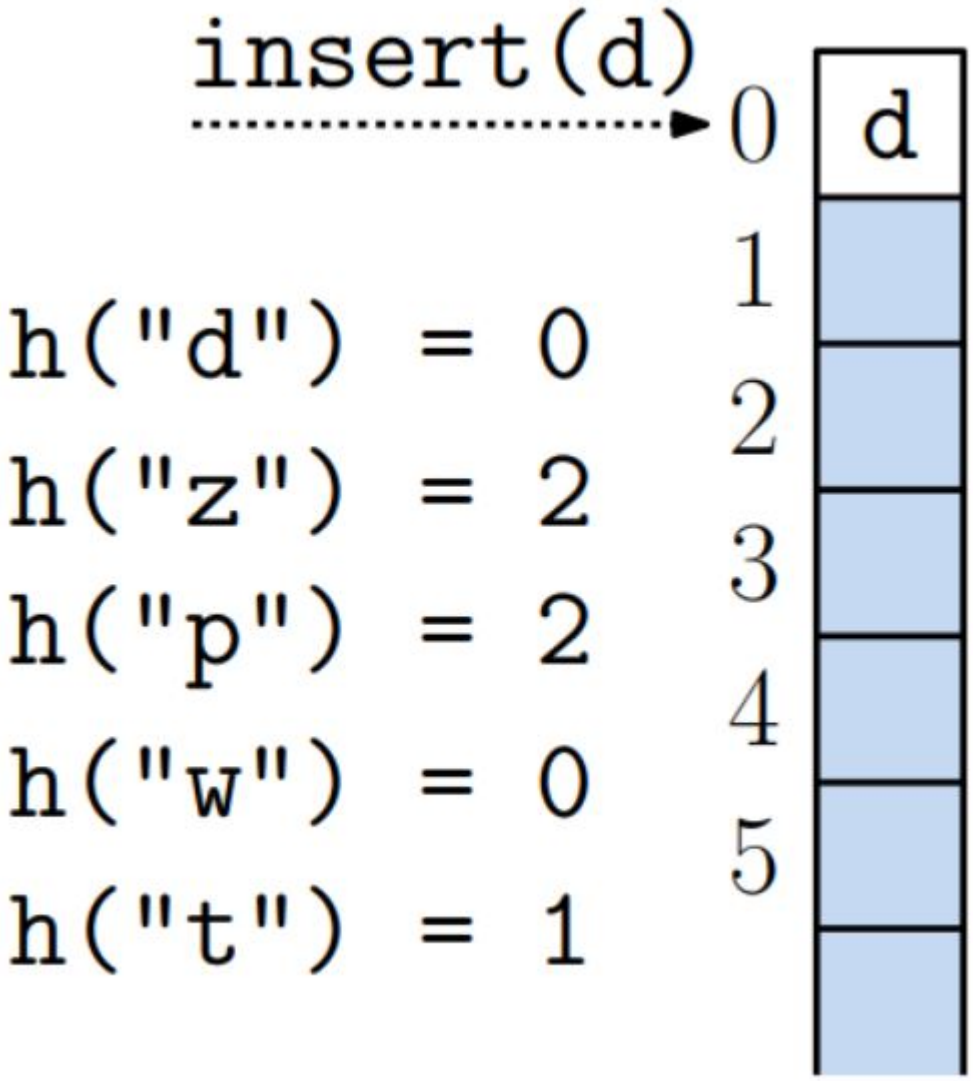
Separate chaining has many benefits (Easy to implement, can operate with limited memory etc). *BUT* it also has one major drawback: As the buckets grow we lose constant search time. This needs a solution.

Come up with a solution to this problem. The requirements are as follows.
1) Your solution must still use separate chaining.
2) It must solve the problem for both extremely long chains and for lots of empty chains.
3) Searches must be amortized constant time.

No files uploaded

Q4 More Insertions
10 Points

Use linear probing. $M = 7$, and don't resize.



No files uploaded

Q5 Deletion

30 Points

Given this table:

0	
1	
2	2
3	
4	15
5	
6	6
7	7
8	28
9	
10	
11	
12	

Resize at 50% capacity like discussed in the lecture, h(x) = x % M, where M is a capacity, initial size = 13. Show each hash table as well as number of probes. In the following hash table, using quadratic probing:

Q5.1

15 Points

insert 19

No files uploaded

Q5.2

15 Points

After inserting 19, delete 28 using *hard* deletion. (You will not be double-penalized if you got the first part incorrect.)

No files uploaded