Q&A for Lecture 08

CS2040 AY2022/2023 Sem 1

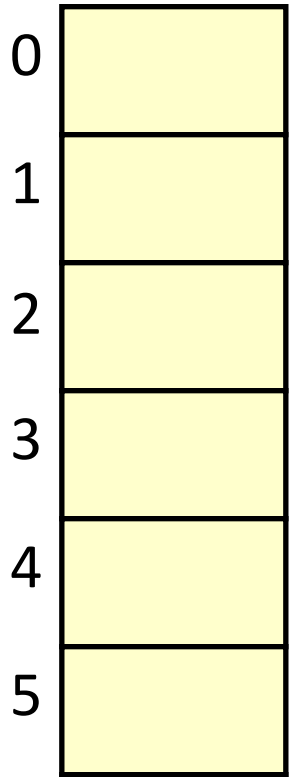
- Can you please explain slide 39. I am quite confused about what α represents. In the slide it says that α is the average length of the linked lists.
- In addition, is the capacity = 16 mentioned in slide 72 the size of the hash table, *m*?
- Lastly, if we use Java HashMap API, where does linear probing, double hashing, etc., come in?

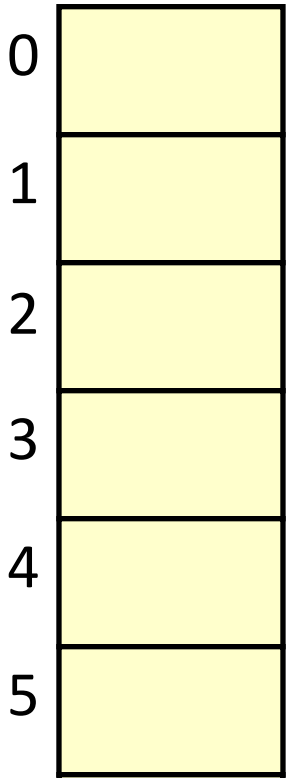
- Load factor: $\alpha = n/m$
  - n: how many slots are filled
  - m: table size (# of slots)

- *Why is α the same as the average chain length?*

- Notes:
  - It is always true that: $\alpha \geq 0$.
  - In Separate Chaining it is possible that: $\alpha > 1$.
  - For the 3 other methods (Linear Probing, Quadratic Probing, Double Hashing) it is always: $\alpha \leq 1$.
  - If the load factor exceeds a certain threshold, we may need to create a bigger table. All keys need to be rehashed.

Ex.: Separate Chaining, m = 6





- Let’s say you have a singly-linked list, which has a head pointer only. How would you efficiently check if the list contains a cycle?

- Let’s say you have a singly-linked list, which has a head pointer only. How would you efficiently check if the list contains a cycle?

- ```
public boolean hasCycle (ListNode node) {
        Set<ListNode> nodesSeen = new HashSet<>();
        while (node != null) {
                if (nodesSeen.contains(node)) {
                        return true;
                } else {
                        nodesSeen.add(node);
                }
                node = node.next;
        }
        return false;
}
```
```

- What is MD5?

- MD5 is a hash function that outputs a 128-bit hash value. The hash value is often called a *digest*.
- Commands:
  - On Unix:
    md5sum <key-file>
  - On Mac:
    md5 <key-file>
  - On Windows:
    certutil -hashfile <key-file> MD5

```
$ cat smallfile
This is a very small file with a few characters

$ cat bigfile
This is a larger file that contains more characters.
This demonstrates that no matter how big the input
stream is, the generated hash is the same size (but
of course, not the same value). If two files have
a different hash, they surely contain different data.

$ ls -l empty-file smallfile bigfile linux-kernel
-rw-rw-r--    1 steve    steve           0 2004-08-20 08:58 empty-file
-rw-rw-r--    1 steve    steve          48 2004-08-20 08:48 smallfile
-rw-rw-r--    1 steve    steve         260 2004-08-20 08:48 bigfile
-rw-r--r--    1 root     root      1122363 2003-02-27 07:12 linux-kernel

$ md5sum empty-file smallfile bigfile linux-kernel
d41d8cd98f00b204e9800998ecf8427e  empty-file
75cdbfeb70a06d42210938da88c42991  smallfile
6e0b7a1676ec0279139b3f39bd65e41a  bigfile
c74c812e4d2839fa9acf0aa0c915e022  linux-kernel
```

- Why do we call the complexity of the Java HashMap methods **insertion** and **deletion** to be O(1) on average?

- Do you think you can use a hash function to manage the passwords on a computer (server)? I.e., we store <key, value> = <password, username> where we use hash(password) to hash the key (and where password is a String). What are the advantages and disadvantages of such a method?

- Comparison. Note: linear probing, quadratic probing and double hashing are also sometimes called Open Addressing.

S.No.	**Separate Chaining**	**Open Addressing**
1.	Chaining is Simpler to implement.	Open Addressing requires more computation.
2.	In chaining, Hash table never fills up, we can always add more elements to chain.	In open addressing, table may become full.
3.	Chaining is Less sensitive to the hash function or load factors.	Open addressing requires extra care to avoid clustering and load factor.
4.	Chaining is mostly used when it is unknown how many and how frequently keys may be inserted or deleted.	Open addressing is used when the frequency and number of keys is known.
5.	Cache performance of chaining is not good as keys are stored using linked list.	Open addressing provides better cache performance as everything is stored in the same table.
6.	Wastage of Space (Some Parts of hash table in chaining are never used).	In Open addressing, a slot can be used even if an input doesn't map to it.
7.	Chaining uses extra space for links.	No links in Open addressing

Q: May I know why the time complexity for this is O(n) instead of O(n^2)?

Problem 1.b. The running time of the following code, as a function of n:

```
public static int loopy(int n){
  for (int i=0; i<n; i++) {
    for (int k=i; k<50; k++) {
      for (int j=0; j<100; j++) {
        System.out.println("Boss!");
      }
    }
  }
  return i;
}
```

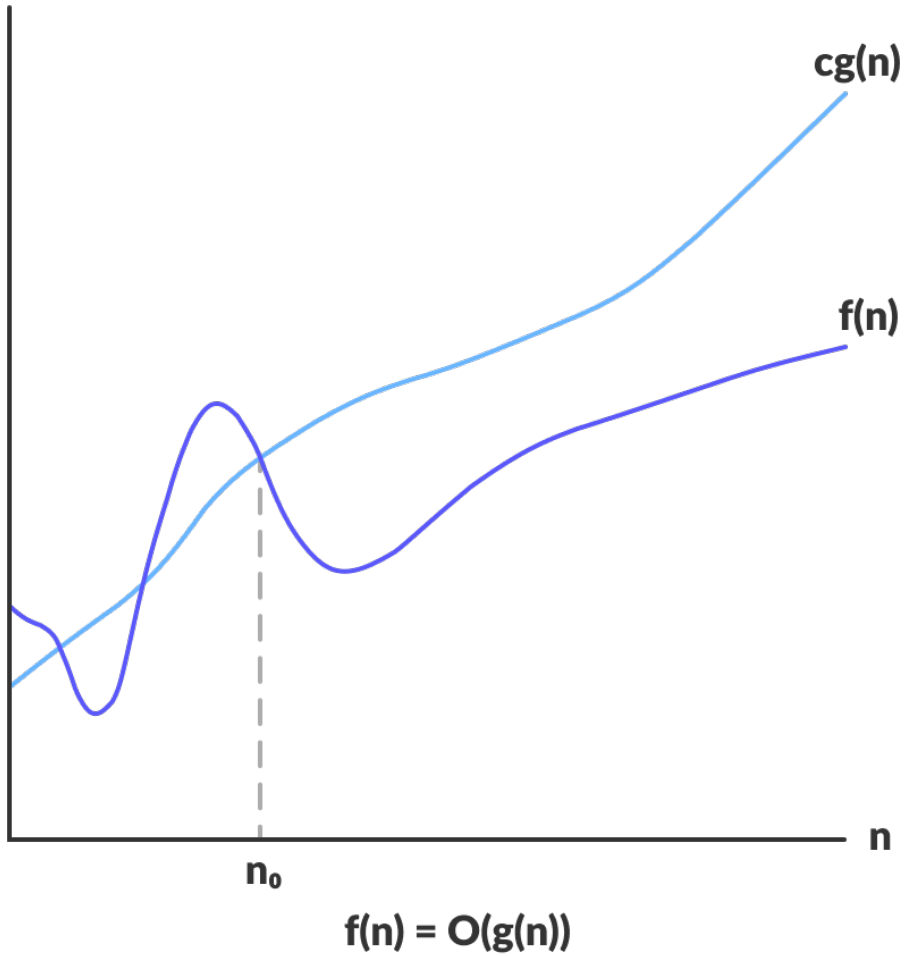
$T(n) =$ [ ]

Hint: Read the code carefully.

A Few Words about Big-O Notation

- Won't go into a very formal definition.
- What we need to know:
- The big-O represents an upper bound such that f(n) is "below" c * g(n) for all $n > n_0$. Then we say f(n) is O(g(n)).
- g(n) is one of: $\log n$, n, $n \log n$, n^2, n^3, $n!$, …

Asymptotic notation: $n > n_0$



”What does the growth rate look like when n becomes large?”