Hash Table

1. What is a Hash Table?

A Hash Table is a data structure that maps keys to values for highly efficient lookup.

2. How does Hash Table work?

Given a (key, value) pair, hash table is data structure which converts the key to an index and then store the value somewhere using that index. It can be implemented by an array of a linked list and a hash code function.

3. How does open addressing work?

Open Addressing (close hashing) is method of collision resolution in hash tables. It’s resolved by probing. Probing includes linear probing, quadratic probing and double hashing.

4. What's the problem of linear probing?

Deletion will be a problem. If one key involves a chain of several probes, it will be lost if somewhere along the chain, one of the other keys is removed, leaving an empty slot. Thus, you can't find the value stored after probing.

5. What is separate chaining?

The idea is to make each cell of hash table point to a linked list of records that have same hash function value.

6. Hash Table insertion, deletion, search, and collision

- Constant, amortized O(1)
- worst case O(n) for search & deletion (add flag to fix it)

7. Hash Table is slow, why?

- poor hash function
- bad open addressing strategy

8.Use hash table to store data, but there is much more data than the machine's RAM, how to deal with that?

add one more machine, rehash and reconstruct the hash table

Quick Sort VS. Merge Sort

1. Difference between them

- Merge Sort requires extra space, and Quick Sort is in place
- They both deploy the idea of divide-and-conquer. In merge sort, the divide step does hardly anything, and all the real work happens in the combine step. Quick Sort is the opposite: all the real work happens in the divide step. In fact, the combine step in Quick Sort does absolutely nothing.

2. Time Complexity

- worst case : Quick Sort is O(n ^ 2), Merge Sort is O(n * log n)
- average case : Both Quick Sort and Merge Sort O(n * log n)

3. When is Quick Sort and Merge Sort used?

Quick Sort:

- unstable sort because of time complexity is between O(n ^ 2) and O(n * log n)
- in-place sort

Merge Sort: stable sort but with extra space

4. Using External Merge Sort when there is a huge amount input.

Throughput & Latency

1. Definition of throughput and latency

Throughput: Throughput is the amount of work we finished in a unit time. Throughput is a measure of how many units of information a system can process in each amount of time.

Latency: Latency is the amount of time to finish an operation. Latency is the delay from input into a system to desired outcome. Unit would be second granule.

2. Example:

-Latency is the delay between the sender and the receiver decoding it, this is mainly a function of the signals travel time, and processing time at any nodes the information traverses.
-Throughput is: the actual rate that information is transferred.

3. The relationship between Throughput and Latency

latency up, throughput down
latency down, throughput up

throughput = bandwith / rtt
rtt = latency * 2

Latency and throughput are not necessarily correlated, because of data density. A system that has high latency but can transfer a huge amount of data can still have higher throughput than a low-latency system.
Throughput is affected by latency if protocol requires synchronous acknowledges of received data.

Processes VS. Thread

1. Definition of processes and thread

Processes: a process is an instance of a computer program that is being executed. A process contains one or more threads.

Thread: a thread of execution is the smallest sequence of programmed instructions that can be managed independently by a scheduler, which is typically a part of the operating system.

2. Difference between processes and thread

-Process runs in separate memory spaces, and threads run in shared memory spaces
-Process is independent from other processes, but threads are depend on other threads.
- Process is controlled by operating system, and threads are controlled by programmer
in a program

3. Difference between Inter-Process Communication(IPC) and Inter-Thread Communications(ITC)

-IPC:

1. A process cannot access memory inside another process, although you can communicate

between processes through various means like: 1. Network packeges 2. Files 3.Pipes 4. Shared memory 5. Semaphores 6. Corba messages 7.RPC calls

2. process to process communication is that the communication must be managed through the operating system

-ITC:
1. A thread can access memory inside a process, even memory that could be manipulated by another thread within the same process. Since all threads are internal to the same running process, they can communicate more quickly (because they don't need the operating system to referee).

4.Example of Process communication and Thread Communication
IPC : X applications communicate with the X server through network protocols.
ITC : Synchronized threads to communicate with each other. ???

5. Methods of IPC:
-File:
A record stored on disk, or a record synthesized on demand by a file server, whichcan be accessed by multiple processes.

-Socket:
A data stream sent over a network interface, either to a different process on the same computer or to another computer on the network. Typically byte-oriented, sockets rarely preserve message boundaries. Data written through a socket requires formatting to preservemessage boundaries.

-Message Queue:
A data stream similar to a socket, but which usually preserves message boundaries. Typically implemented by the operating system, they allow multiple processes to read and write to the message queue without being directly connected to each other.

- Pipe:
A unidirectional data channel. Data written to the write end of the pipe is buffered by the operating system until it is read from the read end of the pipe. Two-way data streams between processes can be achieved by creating two pipes utilizing standard input and output.

- Shared memory:
Multiple processes are given access to the same block of memory which creates a shared buffer for the processes to communicate with each other.

- Semaphore:
A simple structure that synchronizes multiple processes acting on shared resources.

Design Pattern

1. What is design pattern:
A design pattern is a general reusable solution to a commonly occurring problem.

2. Three common design pattern with examples:

-Singleton Pattern: Ensure that only one instance of a class is created and Provide a global access point to the object.

-Factory Pattern: Creates objects without exposing the instantiation logic to the client and Refers to the newly created object through a common interface.

Usage: Defines an interface for creating an object, but lets classes that implements the interface decide which class to instantiate. The Factory method lets a class defer instantiation to subclasses.

-Model-View-Controller Pattern: It divides a given application into three interconnected parts. Model represents the information (the data) of the application and the business rules used to manipulate the data. View corresponds to elements of the user interface such as text, checkbox items, and so forth.Controller manages details involving the communication between the model and view.

The controller handles user actions such as keystrokes and mouse movements and pipes them into the model or view as required.

Example: Building Calculator the Form will house the view and events will be passed to the controller who will then call methods on our model such as ADD/Subtract/Number Press. The model takes care of all the work and it holds the current state of the calculator.

OOP questions:

1. the benefit of objected oriented language

-Objects created for Object Oriented Programs can easily be reused in other programs.

-Large programs are very difficult to write. Object Oriented Programs force designers to go through an extensive planning phase, which makes for better designs with less flaws. In addition, once a program reaches a certain size, Object Oriented Programs are actually easier to program than non-Object Oriented ones.

- Great software maintenance

https://www.cs.drexel.edu/~introcs/Fa15/notes/06.1_OOP/Advantages.html?CurrentSlide=3

2. what is encapsulation and polymorphism and benefits

-Encapsulation: Binding the data with the code that manipulates it. It keeps the data and the code safe from external interference.

Benefit of encapsulation :
1. It improves maintainability and flexibility and re-usability.
2. The fields can be made read-only (If we don’t define setter methods in the class) or
write-only (If we don’t define the getter methods in the class).
3. User would not be knowing what is going on behind the scene.

-polymorphism: Polymorphism is the capability of a method to do different things based on the object that it is acting upon. In other words, polymorphism allows people define one interface and have multiple implementations. There are two different polymorphisms: runtime polymorphism and compile time polymorphism

Benefit of polymorphism:
1. The basic reason behind polymorphism is that we can implement different behaviors of the same object depending upon the reference type passed to an object.
2. jdbc, servlets, jsp have come through polymorphism, if not there we have to remember all dependent classes related to DB, Servers...to use in our java coding
3. Polymorphism enables us to best core java training in Bangalore define one or more methods to have the same name but differs in number of parameters and method types.

3. the difference between Java function call and Python function call ????

-Python and Java call methods similarly: If you have an object x and method foo
, you go x.foo()
-Python also has a library of global functions that are not methods of objects.
One such function is len, which returns the length of its argument.

4. difference between interface and abstract class

Interface:
- cannot provide any code at all, just the signature
- A class may implemented several interfaces
- cannot have instance variables
- must be public or none
- cannot contains constructor
- slow

Abstract class:
- can provide complete default code and/or just the details that have to be overriden.
- in case of abstract class, a class may only extend only one abstract class.
- an abstract can have non-abstract methods
- an abstract class can have instance variables
- contain constructor
- fast

---

coding:

1. Median of data stream

Median: https://www.programcreek.com/2015/01/leetcode-find-median-from-data-stream-java/
Average and Standard Deviation:
http://obscuredclarity.blogspot.com/2012/08/running-average-and-running-standard.html

```
public void put(double value){
    count++;
    average += (value - average) / count;
    pwrSumAvg += (value * value - pwrSumAvg) / count;
    stdDev = Math.sqrt((pwrSumAvg * count - count * average * average) / (count - 1));
}
```

Follow up: How to find the largest(smallest) K% elements?
Using heap and keep tack the heap's size.

2. Weighted random number

http://www.1point3acres.com/bbs/forum.php?mod=viewthread&tid=199979&highlight=Two%2BSigma
https://stackoverflow.com/questions/6737283/weighted-randomness-in-java
https://stackoverflow.com/questions/20327958/random-number-with-probabilities

```
/**
* Design a data structure to hold objects with a corresponding integer weight. It should
support:

  Obtain an object randomly with probability equal to
  (weight of the element) / (sum of the weights).
  Set an object-weight pair. If the object is already in the structure, its weight will
  be updated. Otherwise, the object will be inserted and set to its weight. If the weight
  is zero, the object can be removed.

put("A",3)     3/33
put("B",30)  30/33
*
*/
```

3. Polish notation calculator

Leetcode #150

4. How to evaluate infix notation with parenthesis? (basic calculator)

Leetcode basic calculator I/II

5. Merge sort array

Leetcode #88

6. Best time to buy and sell stock

see Leetcode

7.等差数列，等比数列，平方和 求和及其推导公式

等差数列 Arithmetic progression

Sn=1+2+3+……+(n-1)+n
Sn=n+(n-1)+(n-2)+……+2+1
两式相加
2Sn=(1+n)+(2+n-1)+(3+n-2)+……+(n-1+2)+(n+1)=(n+1)+(n+1)+(n+1)+……+(n+1)+(n+1)
一共 n 项(n+1)
2Sn=n(n+1)

$Sn=n(n+1)/2$

等比数列 Geometric progression

$Sn=a1+a2+\dots\dots+an$

$q*Sn=a1*q+a2*q+\dots\dots+an*q=a2+a3+\dots\dots+a(n+1)$

$Sn-q*Sn=a1-a(n+1)=a1-a1*q\hat{}n$

$(1-q)*Sn=a1*(1-q\hat{}n)$

$Sn=a1*(1-q\hat{}n)/(1-q)$

平方和 Quadratic sum

$(n+1)^3-n^3=3n^2+3n+1$ ①

记 $Sn=1^2+2^2+\dots.+n^2$,　$Tn=1+2+..+n=n(n+1)/2$

对①式从 1~n 求和，得：

$\sum(n+1)^3-n^3=3\sum n^2+3\sum n+\sum 1$

$(n+1)^3-1=3Sn+3Tn+n$

这就得到了 $Sn=n(n+1)(2n+1)/6$