***Interview 1 - Software Design + Logic + Algorithms***

***-15min Main Question***

***You have a 25gb file of words on each line, You only have 2gb of Ram to process it, Sort it Alphabetically, how would it be done?*** *Quicksort a fragment 1/10 of the file, using an ARRAYLIST to go both directions (linkedlist is unilateral), then use merge sort to combine the 10 files.*

***-15mins Bonus Question***

***Create a system design, that covers everything. You want to fly a plane to los-angeles with a controller to pick up a friend and bring him to home in seatac, you want to do this without leaving your house or going to los angeles. How would you create a system to do this?*** *Test and design explanation*

***Interview 2 - Javascript + Design Questions***

***How would you test, what values would you input into perameriers***

***What is javascript and what is it used for?  
What are the downsides of javascript?  
How can javascript return objects?  
Can JavaScript Be asynchronous? And how - callback functions  
What are some coding principles? Using mock objects, pre-unit testing, interface vs abstract, clean code  
Why do we use try catch? For the users, programmer, and to know where errors happened  
What is the issue with putting many parameters in one function   
Is javascript object oriented?   
Can javascript have classes and class structures?  
What kind of data structures have you used and why did you use them in your projects over each other? Hashmaps pros/cons, linked lists pro/cons, arraylist pro/cons (use array list for quick sort a file A-Z)***

***Interview 3 - Software Design, Design Patterns, & Code Problem***

***Stocks class a and b, implement a observer design pattern and tell me how it works***

***Explain inheritance***

***Explain polymorphism  
Can hashmaps hold more than one element?***

***Why use a linkedlist over an arraylist***

***What are some data structures and where did you use one in a project?***

***Explain the difference between a interface and abstract***

***Reverse a string in code+an extra -- How would you test it, edge cases\****

***OnSite Real Interview 6 rounds 30 min-45 each -***

***Round 1 Technical Javascript 30min  
>Code to extract values and parameters from a Url input in javascript  
  
Round 4, 5 Technical Javascript 30mins  
>How would observer design pattern work on a stock buy/sell problem.  
>How would run a button to two methods asynchronously in javascript on a html button  
>Design buttons and resposnes in javascript   
>How would you limit parameters if user inputs char instead of strings?  
>js object value equality checks> how do you do them? & are these right? 5 checks  
>Design and code the fastest algorithim with a random list or random size, how would you code it Effiency.  
>How would you code a html js button count that updates as two users click a and b buttons?   
>Show me how you did an observer design pattern in your app and how it updates  
  
Round 6 Technical OPP 45min > Senior Head  
>Delete node from doubly linked list  
>Code a breath first search***

***>How would a search engine work, how do you populate results   
>Design a search engine   
>How would a tag manager on a website work?  
  
Round 2, 3 30 min  
>Ecommerce & Datacollection & SQL fluff  
>What do you do in a conflict of interest/issue with a teammate***

***Technical CORE***

**What the difference between a while loop and a do while loop** - do while executes at least once, while the while loop is based on the condition

**>>what does a final and private do**

**>>Explain polymorphism?**In a program, things work out a bit the same way: you specify something at a relatively abstract level (go to the store and get supper). Each object provides its own concrete implementation of how to implement that, and in many cases provides for some variation in exactly what it's going to do (e.g., like the differences in favorite foods above).

>>Can abstract class have a constructor in Java? (Yes, detailed answer is here)

>>Which two methods is overridden by an Object, intended to be used as key in HashMap?

(equals and hashCode, read more)

**What is the difference between Overriding and Overloading?** ([detailed answer](http://javarevisited.blogspot.sg/2011/12/method-overloading-vs-method-overriding.html))

Overriding is resolved at runtime while overloading is compile time. Also, rules of overriding and overloading are different, for example in Java, method signature of the overloaded method must be different than original method, but in the case of overriding it must be exactly same as an overriding method.

**>>Difference?**

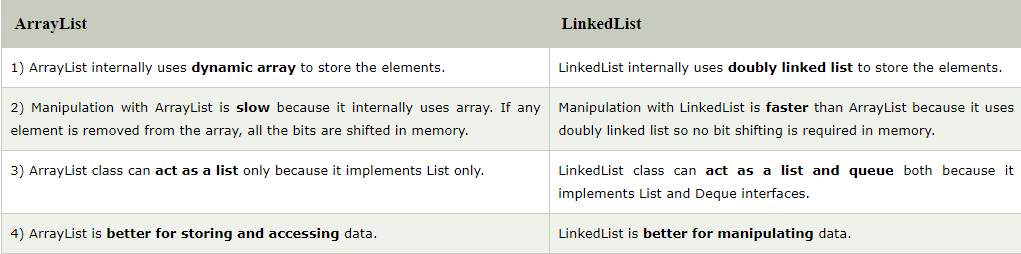
**Interface -Characters--**Allow multiple inheritance.No concrete method.No constructor.No instance variable, it only allow static final constant with the assignment.Can extend interface.Cannot be initialized. But sometime the interface can initialized by providing anonymous inner class.**Guide---**When you think that the API will not change for a while.2When you want to have something similar to multiple inheritance.3.Enforce developer to implement the methods defined in interface. Interface are used to represent adjective or behavior

#### **Abstract Class-Characters-**Can extend abstract class.-Allow concrete method, provide some default behavior.-Allow abstract method.-Cannot be initialized.]**Guide**

* On time critical application prefer abstract class is slightly faster than interface.
* If there is a genuine common behavior across the inheritance hierarchy which can be coded better at one place than abstract class is preferred

***DATA STRUCTURES-***

**What is the difference between ArrayList and LinkedList in Java?**

***\***

***ALGO-  
Binary Searching - Is a recursive sort on a sorted list***

**Bubble Sort -** O(n^2): iterate through array and compare index to the one right of it, if right<index then swap them and compare new\_index to left recursively till left side of array is sorted  
**Merge Sort** - O(nlogn): recursively call merge to split array into 2 subarrays until you have size 1 arrays. Then recursively merge each sub arrays into sorted arrays by comparing left[0] to right[0] and inserting the smaller element back in place. Use this when memory isn’t an issue, when dealing with small arrays or when you think you have a worst case scenario for quicksort.  
External merge sort for sorting array > memory: first sort each memory sized chunk using an in-place sort (quicksort), then use the “merge” part of merge sort to compare ordered sub-chunks to one another  
http://www.ashishsharma.me/2011/08/external-merge-sort.html   
**Quick Sort** - O(nlogn) avg, O(n^2) worst: pick center index as pivot then iterate both sides finding elements smaller than pivot and replacing them with elements greater than pivot so left of pivot is smaller and right is > pivot. Call recursively on both sides. Use when you have to sort in place, large arrays or when you think average case is likely.

**>>What is loose-coupling?** allows one part of the software to modify without affecting another part of the software. For example, in a loosely coupled software, a change in UI layout should not affect the back-end class structure. Use MVC

UNIT TESTING & BASE CASES

**Check- base cases - Biceps**

Right: Are the results right?  
B: Are all the boundary conditions correct?  
I: Can we check inverse relationships?  
C: Can we cross-check results using other means?  
E: Can we force error conditions to happen?  
P: Are performance characteristics within bounds?

**SCRUM**

My experience in scrum teams comes from my databases class where I learned it and ran with a few teams to finish small scale projects. -A product owner creates a prioritized wish list called a **product backlog.**  
-During sprint planning, **the team pulls a small chunk from the top of that wish list**, a sprint backlog, and decides how to implement those pieces.  
-The team has a certain amount of time — a sprint (usually two to four weeks) — **to complete its work, but it meets each day to assess its progress (daily Scrum)**.  
-Along the way, the **ScrumMaster** keeps the team focused on its goal.  
-At the end of the sprint, the work should be potentially shippable: ready to hand to a customer, put on a store shelf, or show to a stakeholder.  
The sprint ends with a **sprint review** and retrospective.  
-As the next sprint begins, the team chooses **another chunk of the product backlog and begins working again. ---I like it because**

* Figures out how to do the work
* Identifies what's getting in its way
* responsibility to resolve all the difficulties within its scope\*
* Works with other parts of the organization to resolve concerns outside their control

##### 

##### **Design Principles**

##### **Open for extension but closed for modifications**

The design and writing of the code should be done in a way that new functionality should be added with minimum changes in the existing code. The design should be done in a way to allow the adding of new functionality as new classes, keeping as much as possible existing code unchanged.

**Questions for Them**

-What kind of company culture like?

-What’s the company's projected direction in a few years, i'm curious about the expanding?

-Ask them why they decided to join the company, what was the most enjoyable project

-Ask them what they wish someone would have told them before they joined.

-question about the company's engineering strategy (testing, Scrum, etc)

***Resume Projects***

**Tetris-**

**\*H**ere I built Tetris the game from java, it was done using a Observer Design pattern to connect to a pre built abstraction. \***T**he power of loosely coupled code and choosing the correct design pattern based on the project in the planning phase, it showed me the power of being able to code for longevity of the software, while allowing for additions and updates. \***I** enjoyed flexibility of java and being able to play the game in the end.

**PowerPaint-**

**\*T**his was a very large application that is similar to microsoft word. It paints images on a plane, opens and saves images. This project was my first large project built from scratch, and showed me what could be done with all the java libraries.**\*T**he way I did this project was through use planning with pseudo code and using a MVC standard. It showed me how to design and test an efficient large scale project when there are many classes to manage. \***I** enjoyed how well java worked for abstraction in this project, everything I learned about java was utilized from encapsulation to data-structures.

**Android App - Music X Match**

**\*T**his was an android application that was designed by me and executed with a team of 2 two people. We created this app in android studio, while also following agile design values to increase our time efficiency. The app is a music discovery app. The goal of the application was to find music by using combinations of strings to find patterns and iterating through lyrical API’s to find songs. This is similar to a standard search engine, but instead based on very specific patterns in language that the end-user inputs. \***T**o create this search engine we needed to implement an API that worked with the java code we have already designed in our initial planning phases. Finding the best API to use took trial and error, as some outdated ones did not work. (Difficulty finding the best one to implement the features we wanted). \***I** enjoyed seeing the end product on a device, but also enjoyed the process of discovery and research my team had to do for this app, which meant understanding the spotify API and bending it to what we wanted to do, which gave us even more possibilities.

**Behavioral Questions - Star pattern**

**What’s a time of Conflict between a teammate to implement a pattern**

**Why do you want to work here?** I’ve lived in washington for 19 years and have seen t-mobile grow to a massive impressive company it is today. The growth has always been impressive to me, from the networking to the management, and even as a customer. I’ve also met John legere multiple times in here in downtown Bellevue, it’s motivating to see him every time as he’s so positive, passionate, and motivated and really moved the company forward. Being part of t-mobile would be something I would be proud of and proud to work for.

**Why this position? (Why Software Engineer?)** Because love engineering, love creative work, Coding is an art or craft, not just engineering. It’s why I choose computer science as my major.

**What was a time you faced a group issue and how did you solve it?**>During my mobile applications class, we were connecting to API’s to fetch lyrics, based on a combination of strings as a project app in android studio. The API we were using could only grab lyrics one at a time, we had to go back start over and look for a new API to use with the code we wrote, while also trying to discover the bug. This was a setback, as there was time constraints, so I used agile principles as well as seducing a meeting to talk out the best way forward. We discussed and compared trade offs, in the end we found a way to divide our work evenly based on our skill set. In the end we found a solution through research, looking for a new API(spotify), and unit testing.

**Where would you like your career at X company to take you?** **What would you want to acquire from this position?** Lifelong successful career, Knowledge, learn more about cutting-edge technology, talk about you like learning., Sense of achievement, conquer those tough problem., Improvement on personal ability on multiple aspects.

**What should you do if you have different opinion with your colleague?**

Talk about your opinion, do not hide your idea. Make sure your idea is reasonable, has enough resource to support it. But the way to talk should be careful. Show your opinion and discuss or compare the trade-off on different idea. Think about another side, consider if you were that person.

**How would you convince other people to adopt your suggestion?**Think wider, diversity of solution.Evidence search.

**What would you do if your boss did something 100% wrong?**Depend on the character of your boss,Different way for different boss But final purpose is point it out. Because you cannot veil the wrong thing.  
**Have you ever made a mistake? How did you handle it?**Whenever I make personal mistakes or with people I make sure to investigate the reason it happened in the first place to prevent it from happening again.

**What's your most challenge project or achievement?** Creating my first large scale project and proving my knowledge alone. (power paint)Background of where you were at the start of this project. General nature of the project. Results from the project, so that this is concluding the basics of the story. Explain where I had problems and difficulties in getting this done, what did I do to conquer them and what other changes I'd make if I was in a similar situation now.

**What would you do if you're facing something impossible?** For example, a challenge thing but near the deadline.Communication, don't avoid difficulties, don't beat around the bush

**Have you did more than was required during doing a project?**Yes, but not too much than required. You shouldn't forget this is a team project.

**Tell me about how you worked effectively under pressure?** I remember to focus as hesitation, fear, and stress caused by pressure never have led to good results. There’s a lot of self trust.

**How do you handle a challenge/STRESS?** Give an example.

With good time management, and planning for things days before they are finished. At UW I saved myself from a lot of stress unlike my peers during big projects. Even though I was confident I could finish a long programming assignment if pulled more time, I always prefer starting on assignments early so I would have more control of the assignment if I didn’t know how to do something. By being effective with my time, I was able to have more control of my assignments, thus reducing the amount of bugs and being able to balance and prioritize areas of the project that need more work.