<https://medium.com/javarevisited/java-understanding-list-set-map-and-queue-4330cf54596d>

1.

You want to keep track of all the unique users who have logged on to your system.

**Set**: keep track of unique users

**Similar to List, but it does not accept duplicated elements and does not keep a defined order. We can find the following implementations:**

You are creating a version control system and want to associate each file name with a Blob.

**Map** : let you pair a key and value

We are running a server and want to service clients in the order they arrive.

**Queue**

We can push clients to the front of the queue as they arrive, and pop them off the queue as we service them.

**when you need your elements are added/removed in a specific order. Similar to the real world, a Queue is the structure you use when you want to process elements in some order,**

We have a lot of books at our library and we want our website to display them in some sorted order. We have multiple copies of some books and we want each listing to be separate

List

**list is an ordered collection of items.**

**2.**

**Worst N \* M**

**Best N \* log M**

**3.**

**a.**

**Θ N**

**b.**

**O log N**

**c.**

**Θ(N log(N))**

**4.**

**a.**

1 public static boolean findSumFaster(int[] A, int x){

2 int left = 0;

3 int right = A.length - 1;

4 while (left <= right) {

5 if (A[left] + A[right] == x) {

6 return true;

7 } else if (A[left] + A[right] < x) {

8 left++;

9 } else {

10 right--;

11 }

12 }

13 return false;

14 }

Examprep07

**1**

**a**

**N; N^2**

**b**

**N 2^N**

**N N^N**

**2**

**4**

1. Storing all the Students in Shreyas’s first section in alphabetical order

**List**

1. Storing all the Students by their section, where Students within a section are sorted alphabetically.

**Map, key = section, value = List**

1. Storing the Students in all of Shreyas’s sections. There shouldn’t be duplicates

**Set**

1. Determining all the Students that attend more than one of Shreyas’s sections

**2 Sets**

. Iterate through the Students in each Section and, if the first set contains the Student, add it to the second set. Otherwise, add the Student to the first set.

1. Quickly getting a Student by sid.

**Map**

1. Quickly getting a Student by name or sid. Names aren’t necessarily unique

**2 Maps: sid+name; name + Set of student**

1. Cycling through the Students in one discussion section.

**List**