**Unit 7 Cheat Sheet**

ArrayLists:

* ArrayLists are Re-sizeable arrays
* Mutable – can be changed during runtime
* In exam, might be referred to as List
* Part of the util package:

- import java.util.\*;

- import java.util.ArrayList;

Declaring and ArrayList:

ArrayList<Type> nameList = null;

Creating an ArrayList:

ArrayList<Type> name = new ArrayList<Type>();

//Type will be changed to the type of objects you will be storing in the list

* All primitive types must be wrapped in objects before they are added to an ArrayList.
* int values can be wrapped in Integer objects, double values can be wrapped in Double objects etc.
* Any object can be put in a List.

Converting Arrays to Lists:

You can convert arrays to ArrayLists using its constructor with an argument Arrays.asList(arrayname)

example:

String[] names = {"Dakota", "Madison", "Brooklyn"};

ArrayList<String> namesList = newArrayList<String>(Arrays.asList(names));

Important Methods:

.size() – get the number of items in an array list

\*an empty array list has a size of 0

\*Trying to get the size of an ArrayList that is set to null will give you a NullPointerException

.add(obj) – adds the passed object to the end of the list

.add(index,obj) – adds the passed object to the passed index

.remove(index) – remove an item form the list

\*This will move all other items over and decrease the list size by 1

.get(index) – get the object at the index

.set(index,obj) - set the object at an index

More methods at <https://docs.oracle.com/javase/8/docs/api/java/util/ArrayList.html>

Traversing Lists:

Similarly to Arrays, you can traverse ArrayLists with loops.

Example:

ArrayList<Integer> list = new ArrayList<Integer>();

Int sum=0;

for (Integer num: list)

{

sum= sum+num;

}

\*in regular for loops and while loops, remember to use .get method and not ()

**For search and Sort examples visit:**

https://exlskills.com/learn-en/courses/java-search-sort-search\_sort\_java/search-and-sort-DDyPwZvFgFhB/sorting-JDnpcSGoumka/insertion-sort-hTJsmYfBrZQP

Sequential (linear) Search:

\*only method that can be used to find a value in unsorted data

\*searches linearly

Binary Search:

\*can only be used on data that is sorted

\*It checks the middle of the data to see if that middle value is less than, equal, or greater than the desired value and then based on the results of that it narrows the search. It cuts the search space in half each time.

\* Binary search calculates the middle index as left + right / 2 where left starts out at 0 and right starts out at the array length - 1 (the index of the last element). Remember that integer division gives an integer result so 2.5 becomes 2.

**Runtime:**

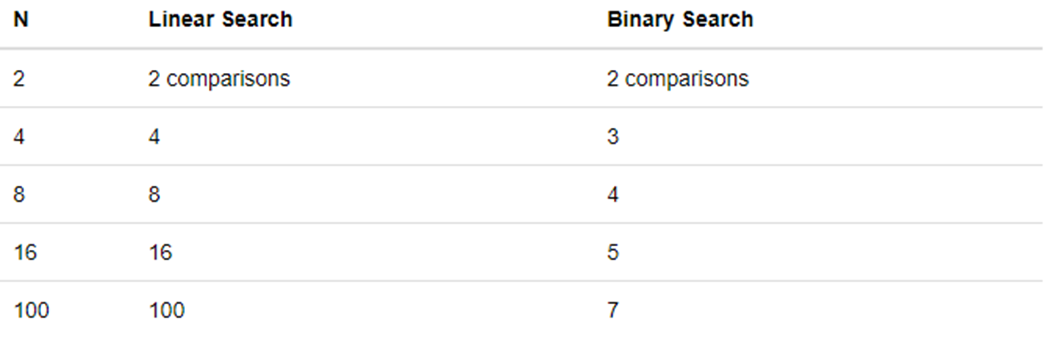
\*measures how fust the algorithm runs.

\* When considering which algorithm to use think of the worst-case behavior (if you can not find the item).

\*We use the big O notation to represent runtime.

\*Can be describes with mathematical functions.

\*for a linear search the runtime is linear, for a binary search the runtime is log base 2 of n.



Selction sort:

https://www.youtube.com/watch?v=g-PGLbMth\_g

Binary Search:

https://www.youtube.com/watch?v=g-PGLbMth\_g