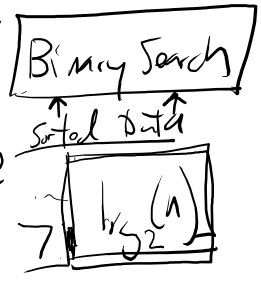


1-100
 Window size
 Yes → Too High
 No → Too Low
 You Pick
 Yes → Too High
 No → Too Low
 find middle



Sorting

ArrayList
 $O(n)$
 $O(n)$
 $O(n)$
 $O(h)$
 $O(n)$
 $O(n)$

Map $\langle K, V \rangle$
 put (k, v)
 get (k)
 boolean \leftarrow contains (K)
 boolean \leftarrow isEmpty $()$
 int \leftarrow size $()$
 delete (K)

Map
 ArrayList
 ArrayList values
 connected by index
 for $i = 0, 1, 2$
 $s[2] = 4$
 "Hello", "Lemur"
 "T"
 \leftarrow get (i)
 removed (i)

Magic ???
 Efficient
 Time - space
 Bad ----- Better
 find

Java Collections
 Stack → Queue
 List → Map
 Sorted
 efficiently
 $O(n^2)$
 $O(n \cdot \log n)$
 No more

Keys are Unique!

d = {}
 Efficient
 $d[5] = \text{"Hello"}$
 $d[21] = \text{"Lemur"}$
 key value str
 $d["b"] = 7$
 print $(d[5])$
 put $(4, \text{"T"})$
 get (21)
 delete (21)

HashMap $\langle \text{Integer}, \text{String} \rangle$ d =
 new HashMap $\langle \text{Integer}, \text{String} \rangle$ ();
 d.put $(5, \text{"Hello"})$;
 d.put $(21, \text{"Lemur"})$;
~~d.put $(\text{"b"}, 7)$;~~
 sOP $(d.get(5))$;
 =