We have an array of Friend objects, each storing the name and age of a friend. We need to find Jenny’s age. What’s the easiest way to do it?

**Name:**

**Algorithm:**

**Code:**

public int findFriend(String name)

{

**How long will it take?**

**Some Variations** - What does this code do?

int myArray = new int [5000];

Random rand = new Random();

for (int i = 0; i < myArray.length; i++)

myArray[i] = 1 + rand.nextInt(1000);

target = 479;

timesFound = 0;

for (int i = 0; i < myArray.length; i++)

if (myArray[i] == target)

timesFound++;

System.out.println (timesFound);

**We can actually find data faster than that**

**Name:**

**Strategy**:

* 1. If the array isn’t empty, look at the element in the
  2. If you found the target, stop
     + otherwise, eliminate ½ of the elements (left or right) and return to

**STRONG PRE\_CONDITION TO USE A BINARY SEARCH:**

**Example**



**Code**

This method will only sort arrays of ints

// Find the index of target in array. Return -1 if target is not in array.

// Assume the array is totally filled with values (at indices 0…array.length-1)

public static int binarySearch( int [ ] array, int target )  
{  
 int startIndex = 0; // first index of portion we’re examining  
 int endIndex = array.length - 1; // last index of portion we’re examining   
 int middle;  
  
 while ( endIndex >= startIndex ) // there actually is an array  
 {  
 middle = ( startIndex + endIndex )/ 2; // index of element in middle of array  
  
 if ( array[middle] ==   
 return middle; // target found at middle—so bail out  
 else if ( array[middle]   
 endIndex = middle - 1; // search left half of array  
 else  
 startIndex = middle + 1; // search right half of array  
 }

return -1; // if we reached this line….

}

**How would the code change with objects?**

**How do we get the data into ascending order?**