Sort/Search

1. The sorting algorithm: Bubble Sort, Binary search
2. Starting with the Bubble Sort it is the simplest algorithm that works by repeatedly swapping the adjacent elements if they are in the wrong order. The Binary search method: a searching algorithm sed n a sorted array by repeatedly diving the search interval in half.
3. The bubble sort is where the algorithm goes through an array and compares each item in the array and swaps them. It will continue to do this until there is no more items in the array can be swapped. A good way to see this is in phases

Phase 1: here would be the list of times in the array. (7,1,4,2,8)

Now the algorithm will begin to go through the array and compare the data in each element.

Since 7 > 1 it will swap them and now the array will look like this (1,7,4,2,8) and this pattern will continue until it reaches the end.

(1,**7,4**,2,8) ----> (1,**4,7,**2,8)

(1,4,**7,2,**8)-----> (1,4,**2,7**,8)

(1,4,2,**7,8**)-----> (1,4,2,**7,8**): and because 7< 8 that means that it is already sorted

Phase 2: the algorithm will now start from the beginning again and compare the data again

(**1,4**,2,7,8)-----> (**1,4**,2,7,8) 1 is < 4 so no change needed

(1**,4**,**2**,7,8)-----> (1,**2,4**,7,8)

(1**,**2,**4,7**,8)-----> (1,2,**4,7**,8) 4 < 7 no change

(1**,**2,4,**7,8**)-----> (1**,**2,4,**7,8**) 7 < 8 no change

Phase 3: now the algorithm will go back through the array 1 last time as it made a change it the last phase so even though we know that the array is sorted the algorithm doesn’t until there are no more changes made in a phase and I don’t think I need to show this in a diagram

After an array has been sorted, we can use another algorithm to search the array for specific data points that we want and that is where the binary search comes in:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 3 | 5 | 10 | 15 | 24 | 34 | 45 | 46 | 51 |

Here is our array and as you can see it is already sorted and lets say we are searching for the number 24 the algorithm it will then break down the array from both endpoint and the middle of the endpoints ex)

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 3 | 5 | 10 | 15 | 24 | 34 | 45 | 46 | 51 |
| L=0 |  | lower |  | M=4 |  |  | higher |  | H=9 |

As you can see it takes the low endpoint L and the high endpoint H and finds the middle M and since we start at 0 in any given data structure like an array the index number of 4 would be the middle, it then breaks the array into 2 half’s the lower half and higher half. It will then compare the data in index number 4 and see if it is higher or lower to 24 and since 15 < 24 it will take the higher half and break it up in half’s again while also ignoring the midpoint of the last half as we now know that it is not what we are searching for

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 3 | 5 | 10 | 15 | 24 | 34 | 45 | 46 | 51 |
|  |  | Black = | Ignore |  | L = 5 | Lower | M = 7 | Higher | H=9 |

Now since 45 > 24 it will take the lower half

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 3 | 5 | 10 | 15 | 24 | 34 | 45 | 46 | 51 |
|  |  | Black = | Ignore |  | L = 5, M = 5 | H = 6 |  |  |  |

And now that the midpoint is the index number of 5 which is the data point that we were searching for the algorithm will return the number 5 as to tell us where 24 is located int the array

1. Data Structures: Arrays, Linked-List
2. Ex code of a Bubble sort:

class BubbleSort {

void bubbleSort(int arr[])

{

int n = arr.length;

for (int i = 0; i < n - 1; i++)

for (int j = 0; j < n - i - 1; j++)

if (arr[j] > arr[j + 1]) {

int temp = arr[j];

arr[j] = arr[j + 1];

arr[j + 1] = temp;

}

}

void printArray(int arr[])

{

int l = arr.length;

for (int i = 0; i < l; ++i)

System.out.print(arr[i] + " ");

System.out.println();

}

public static void main(String args[])

{

BubbleSort ob = new BubbleSort();

int arr[] = { 64, 34, 25, 12, 22, 11, 90 };

ob.bubbleSort(arr);

System.out.println("Sorted");

ob.printArray(arr);

}

}

Example of a binary search:

class BinarySearch {

int binarySearch(int arr[], int x)

{

int l = 0, r = arr.length - 1;

while (l <= r) {

int m = l + (r - l) / 2;

if (arr[m] == x)

return m;

if (arr[m] < x)

l = m + 1;

else

r = m - 1;

}

return -1;

}

public static void main(String args[])

{

BinarySearch ob = new BinarySearch();

int arr[] = { 2, 3, 4, 10, 40 };

int n = arr.length;

int x = 10;

int result = ob.binarySearch(arr, x);

if (result == -1)

System.out.println("Element not present");

else

System.out.println("Element found at "+"index " + result);

}

}