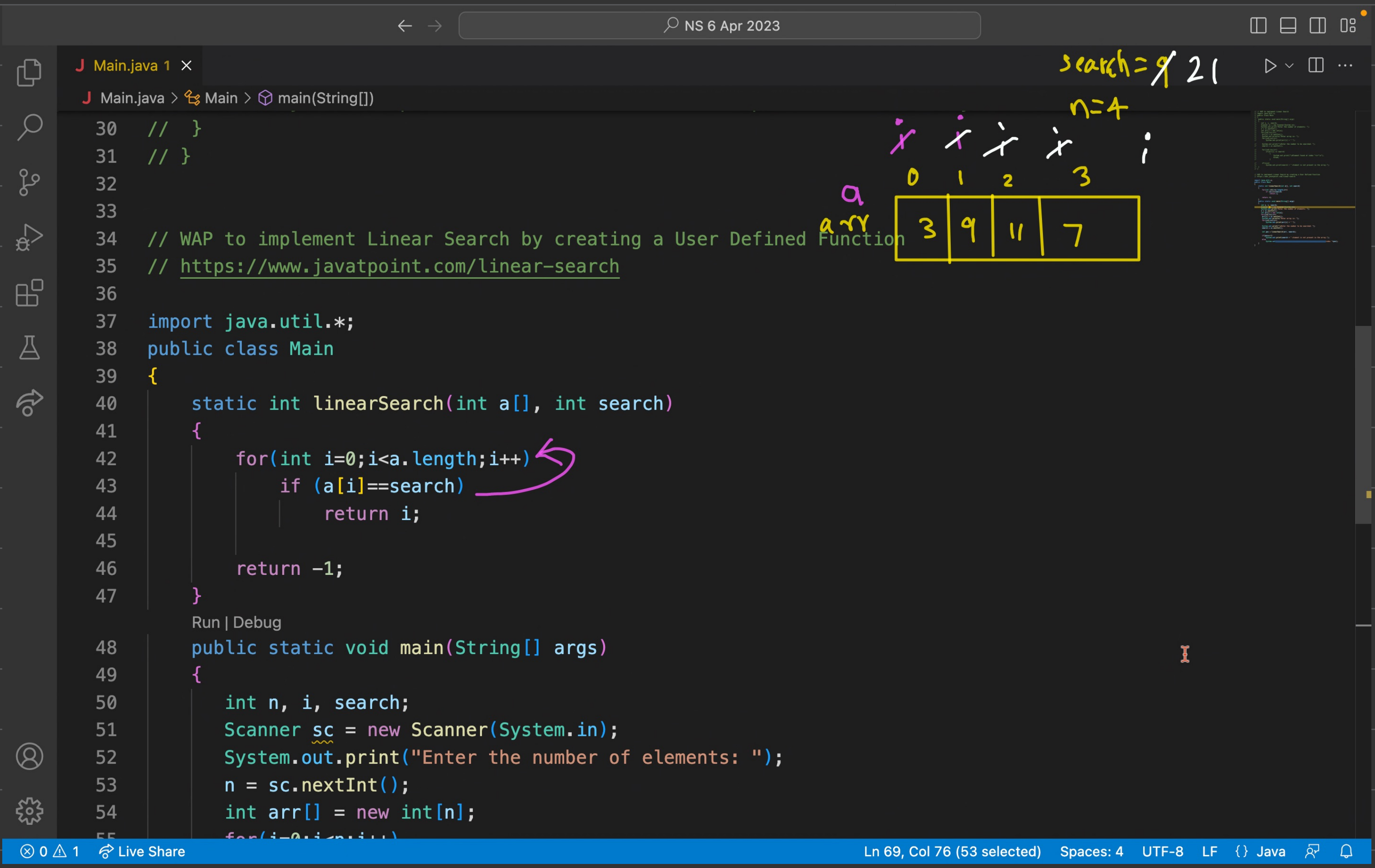
06 Apr 2023

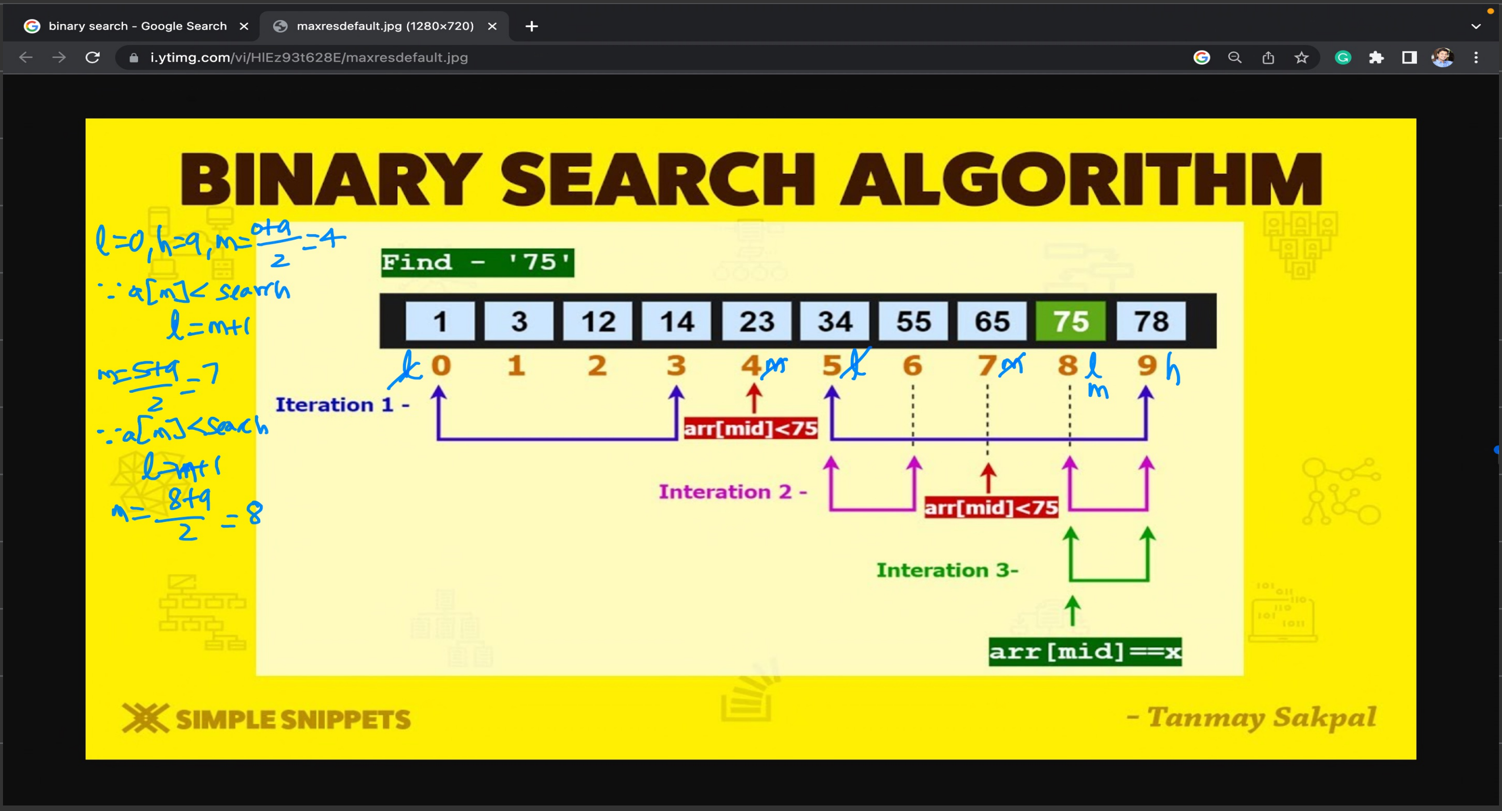


```java
//  }
// }


// WAP to implement Linear Search by creating a User Defined Function
// https://www.javatpoint.com/linear-search

import java.util.*;
public class Main
{
    static int linearSearch(int a[], int search)
    {
        for(int i=0;i<a.length;i++)
            if (a[i]==search)
                return i;

        return -1;
    }
    public static void main(String[] args)
    {
       int n, i, search;
       Scanner sc = new Scanner(System.in);
       System.out.print("Enter the number of elements: ");
       n = sc.nextInt();
       int arr[] = new int[n];
       for(i=0;i<n;i++)
```



$l=0, h=9, m=\frac{0+9}{2}=4$

$\because a[m] <$ searrh

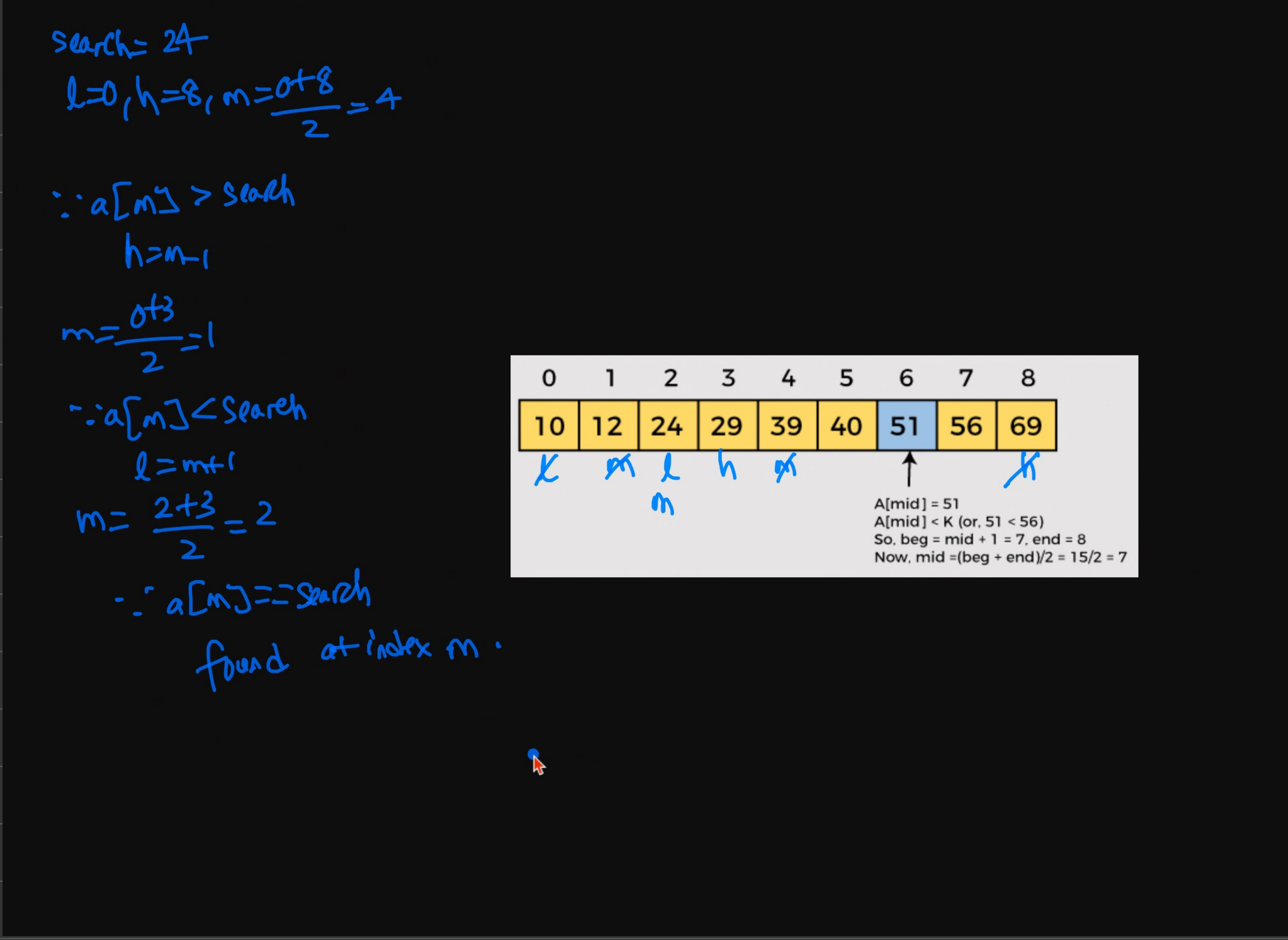
$l=m+1$

$m=\frac{5+9}{2}=7$

$\because a[m] <$ search

$l=m+1$

$m=\frac{8+9}{2}=8$



search = 24

$l=0, h=8, m=\frac{0+8}{2}=4$

$\because a[m] > \text{search}$

$h = m-1$

$m=\frac{0+3}{2}=1$

$\because a[m] < \text{search}$

$l = m+1$

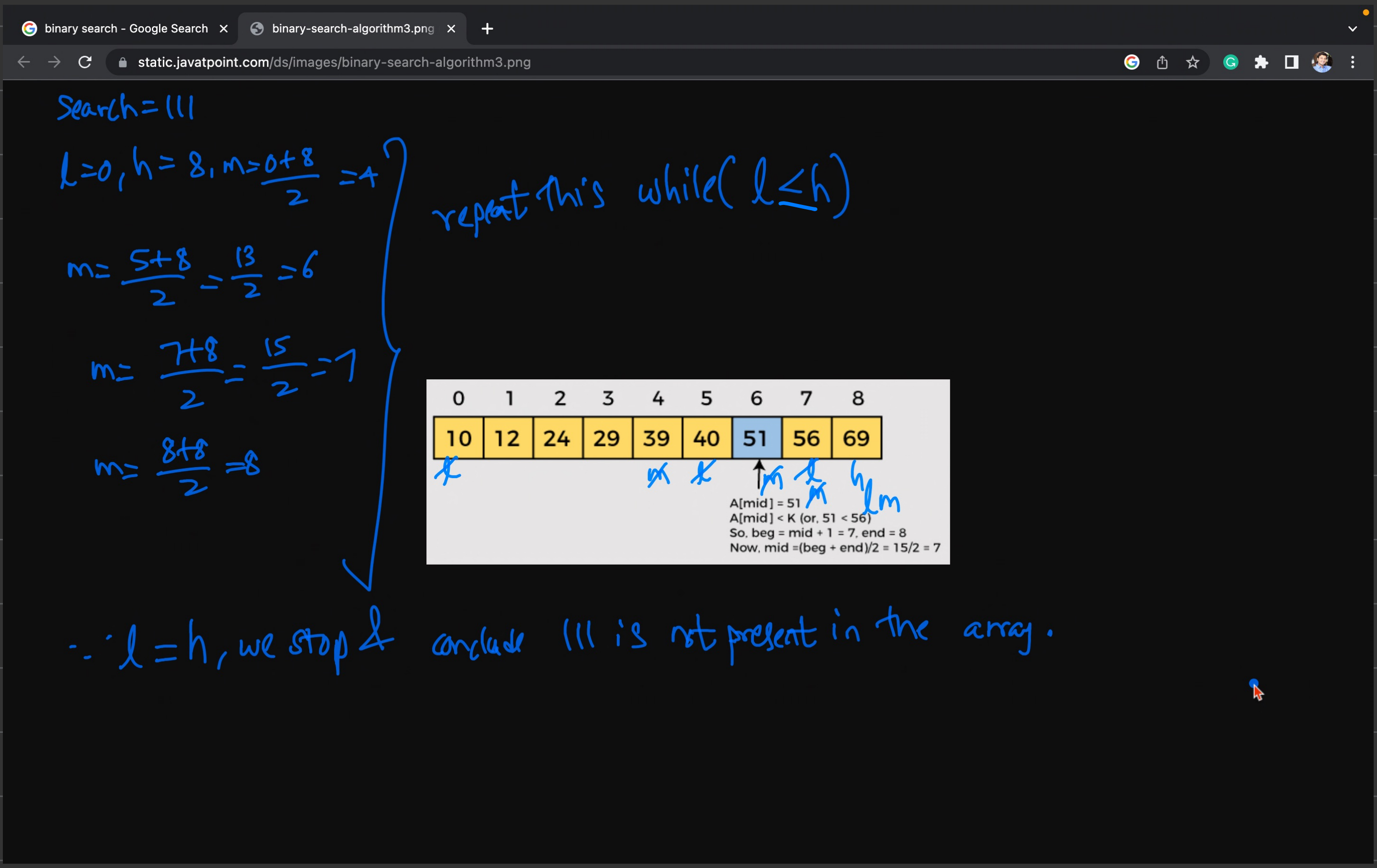
$m = \frac{2+3}{2} = 2$

$\because a[m] == \text{search}$

found at index m.

0	1	2	3	4	5	6	7	8
10	12	24	29	39	40	51	56	69

A[mid] = 51
A[mid] < K (or, 51 < 56)
So, beg = mid + 1 = 7, end = 8
Now, mid =(beg + end)/2 = 15/2 = 7



binary search - Google Search
binary-search-algorithm3.png
static.javatpoint.com/ds/images/binary-search-algorithm3.png

search = 111

$l=0, h=8, m=\frac{0+8}{2}=4$

$m=\frac{5+8}{2}=\frac{13}{2}=6$

$m=\frac{7+8}{2}=\frac{15}{2}=7$

$m=\frac{8+8}{2}=8$

repeat this while($l \leq h$)

0	1	2	3	4	5	6	7	8
10	12	24	29	39	40	51	56	69

A[mid] = 51
A[mid] < K (or, 51 < 56)
So, beg = mid + 1 = 7, end = 8
Now, mid =(beg + end)/2 = 15/2 = 7

$\because l = h$, we stop & conclude 111 is not present in the array.

n=6

0	1	2	3	4	5
8	7	5	11	15	2
	s				
7	8	5	11	15	2
		s			
7	5	8	11	15	2
7	5	8	11	15	2
7	5	8	11	15	2
					s
7	5	8	11	2	15

Pass I

after Pass I, the largest no in the array i.e. 15 is at its right position

	s				
5	7	8	11	2	15
5	7	8	11	2	15
5	7	8	11	2	15
				s	
5	7	8	2	11	15
5	7	8	2	11	15

Pass II

After Pass II, 2nd largest no in the array is at its right position i.e. second last

5	7	8	2	11	15
5	7	8	2	11	15
			s		
5	7	2	8	11	15
5	7	2	8	11	15
5	7	2	8	11	15

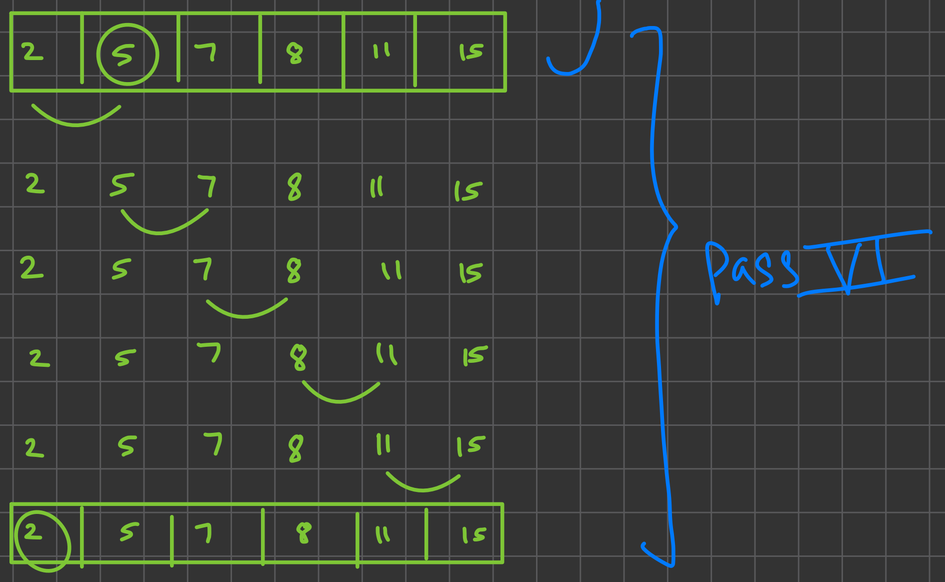
Pass III

5	7	2	8	11	15
		s			
5	2	7	8	11	15
5	2	7	8	11	15
5	2	7	8	11	15
5	2	7	8	11	15

Pass IV

	s				
2	5	7	8	11	15
2	5	7	8	11	15
2	5	7	8	11	15
2	5	7	8	11	15

Pass V



If $n=6$, # passes reqd $= 6$

i.e. we we give a sorted array of 20 values, it will take 20 passes, everytime comparing these values even when it is just not reqd

That is the biggest disadvantage of Bubble sort & hence it is considered as one of the worst Sorting Algorithms.

H/W: Just as shown above, sort the below elements using Bubble Sort.

0	1	2	3	4	5	6	7	8
6	7	8	9	5	4	3	2	1

$n=9$

———— X ————