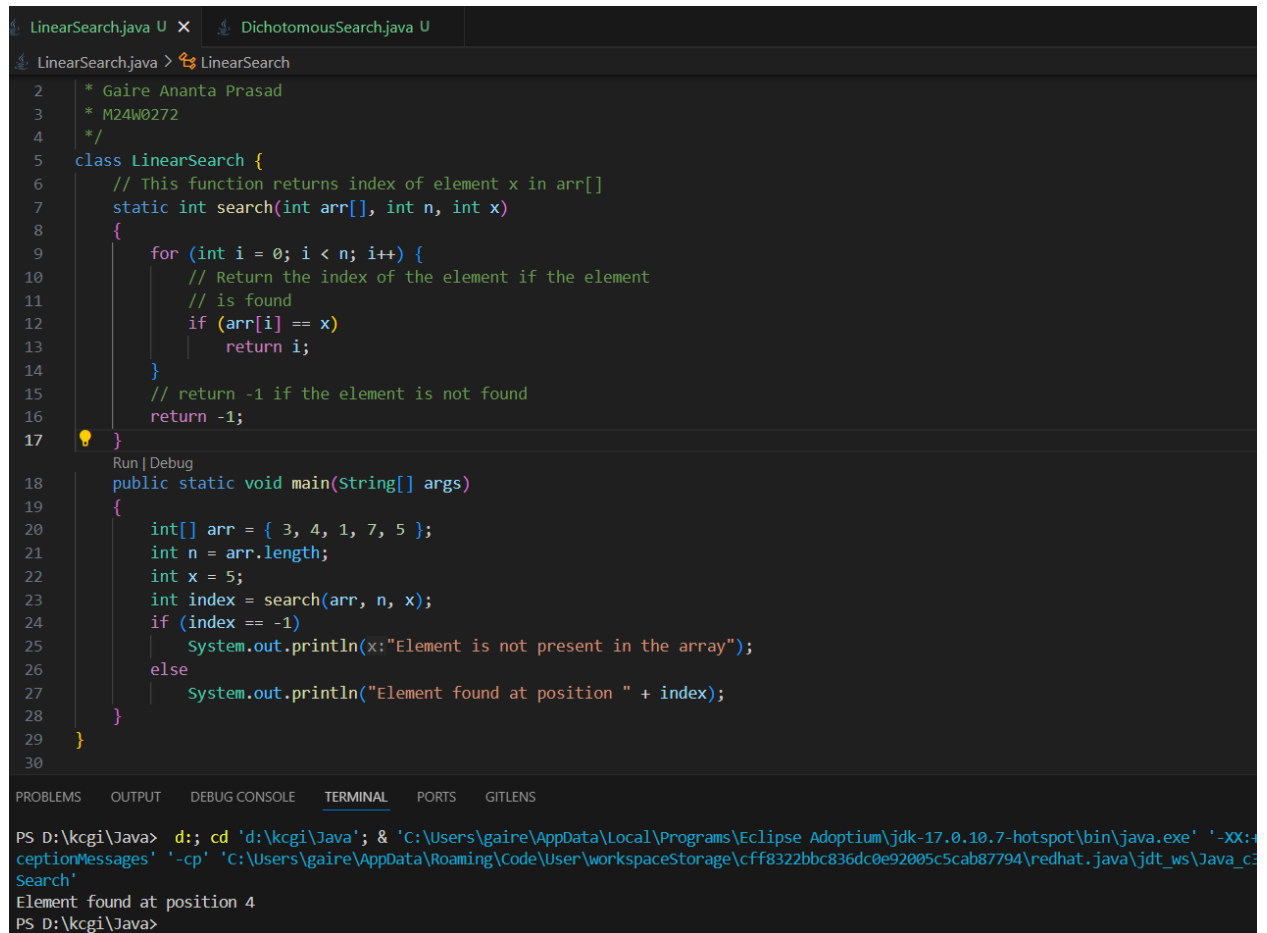


GAIRE ANANTA PRASAD  
M24W0272  
LEANER SEARCH METHOD



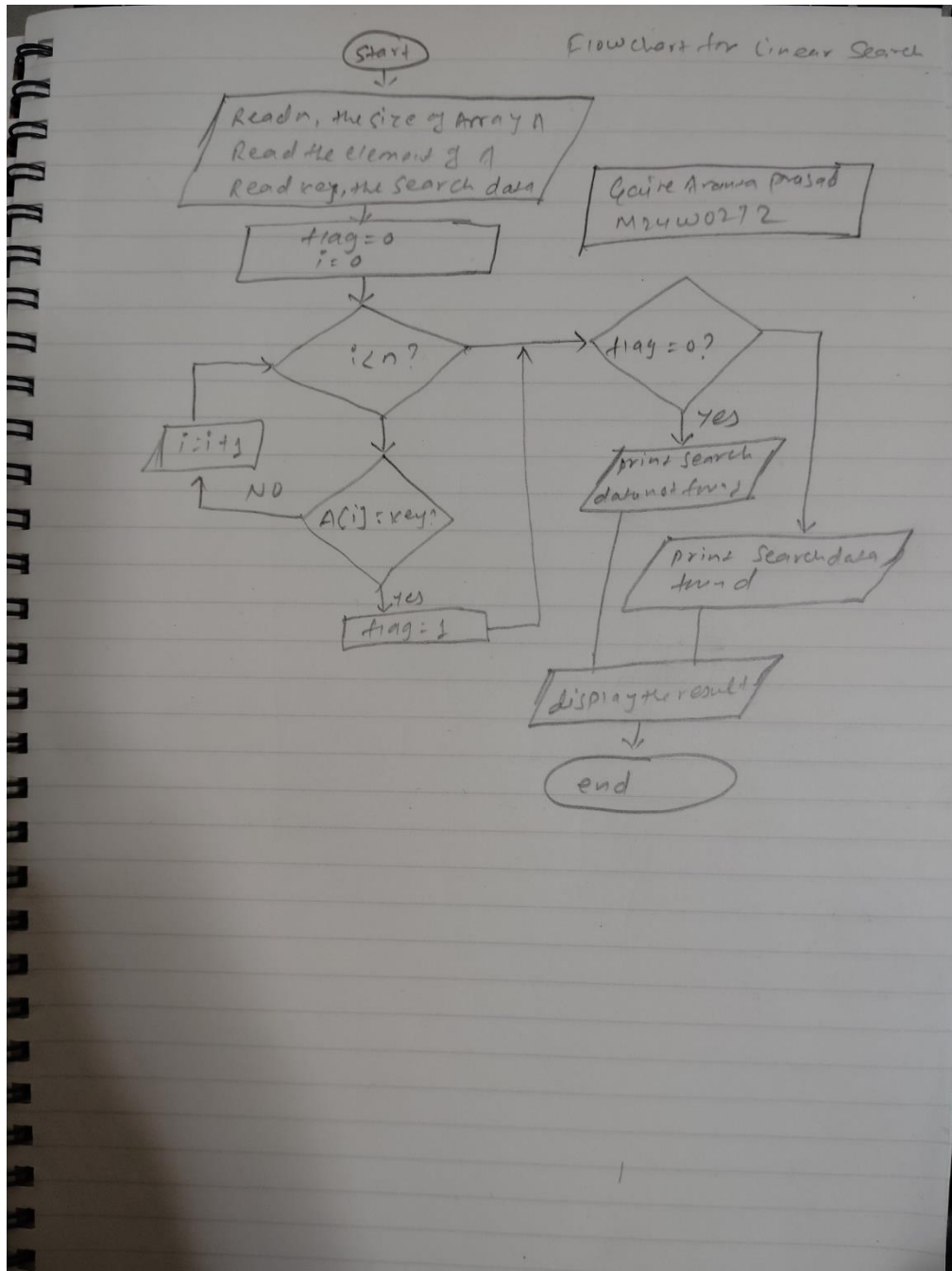
```
LinearSearch.java U X  DichotomousSearch.java U
LinearSearch.java > LinearSearch
2  * Gaire Ananta Prasad
3  * M24W0272
4  */
5  class LinearSearch {
6      // This function returns index of element x in arr[]
7      static int search(int arr[], int n, int x)
8      {
9          for (int i = 0; i < n; i++) {
10             // Return the index of the element if the element
11             // is found
12             if (arr[i] == x)
13                 return i;
14         }
15         // return -1 if the element is not found
16         return -1;
17     }
18
19     Run | Debug
20     public static void main(String[] args)
21     {
22         int[] arr = { 3, 4, 1, 7, 5 };
23         int n = arr.length;
24         int x = 5;
25         int index = search(arr, n, x);
26         if (index == -1)
27             System.out.println(x+"Element is not present in the array");
28         else
29             System.out.println("Element found at position " + index);
30     }
31 }
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS GITLENS

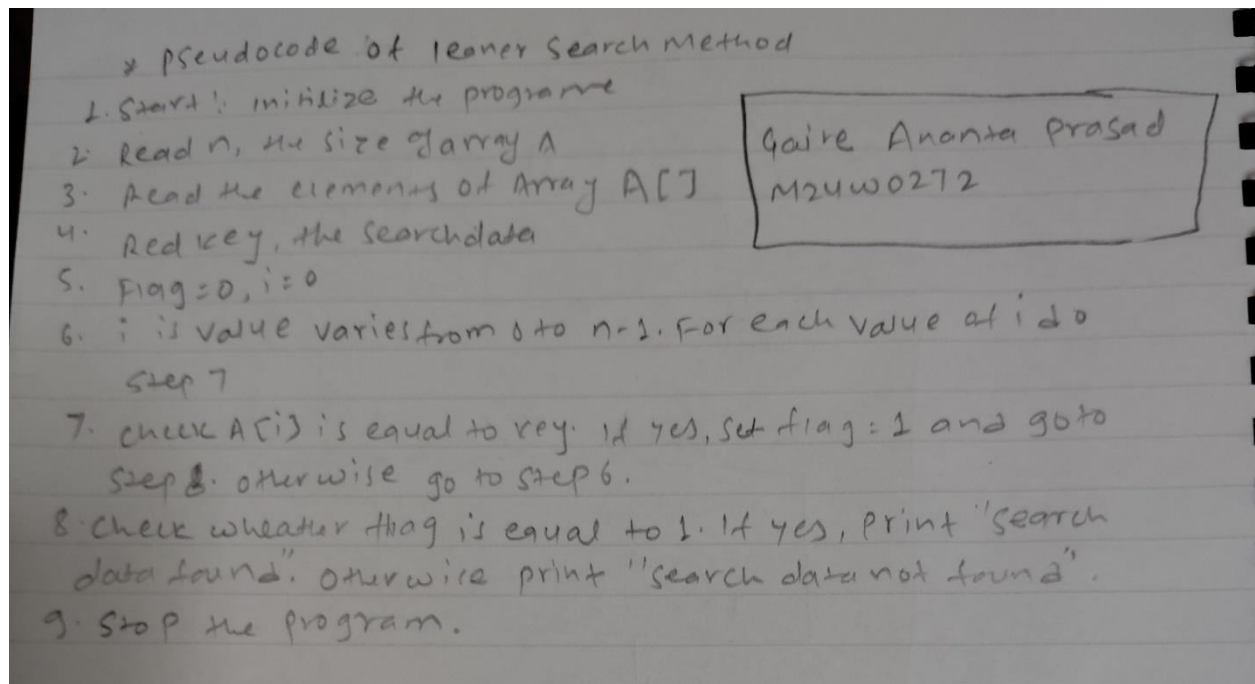
```
PS D:\kcgi\Java> d:; cd 'd:\kcgi\Java'; & 'C:\Users\gaire\AppData\Local\Programs\Eclipse Adoptium\jdk-17.0.10-hotspot\bin\java.exe' '-XX:HeapDumpPath=C:\Users\gaire\AppData\Local\Temp\jvmti\jvmti.log' '-cp' 'C:\Users\gaire\AppData\Roaming\Code\User\workspaceStorage\cff8322bbc836dc0e92005c5cab87794\redhat.java\jdt_ws\Java_c3' 'Search'
Element found at position 4
PS D:\kcgi\Java>
```

Gaire Ananta Prasad  
M24W0272

## Flowchart



## Pseudocode



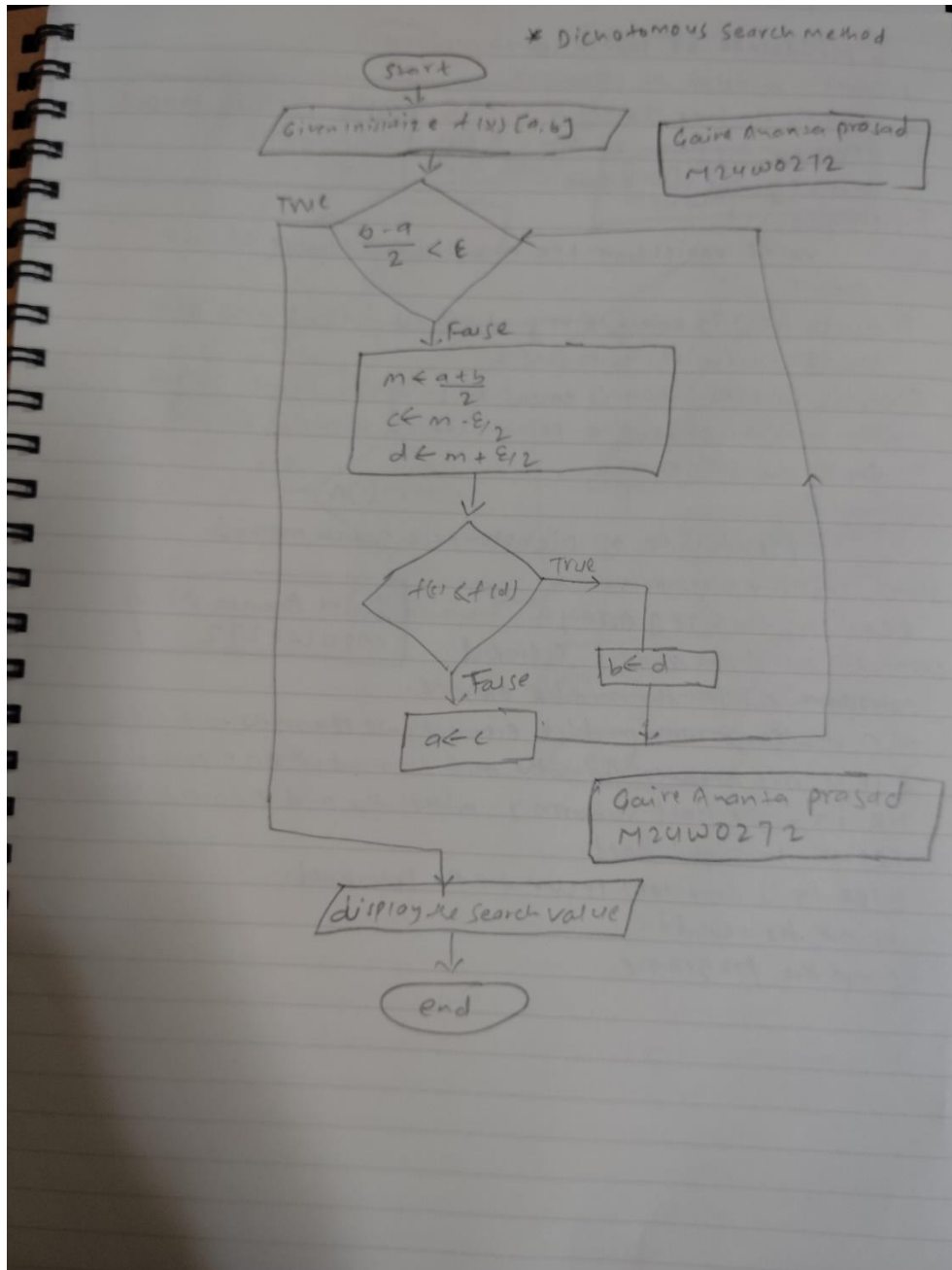
## DICHOTOMOUS SEARCH METHOD

```
LinearSearch.java U  DichotomousSearch.java U X
DichotomousSearch.java > DichotomousSearch > binarySearch(int[], int)
1  /*
2   * GAIRE ANANTA PRASAD
3   * M24W0272
4   */
5  public class DichotomousSearch {
6
7      public static int binarySearch(int[] array, int target) {
8          int left = 0;
9          int right = array.length - 1;
10
11         while (left <= right) {
12             int middle = left + (right - left) / 2;
13
14             // Check if the target is present at mid
15             if (array[middle] == target) {
16                 return middle; // Target found
17             }
18
19             // If target is greater, ignore left half
20             if (array[middle] < target) {
21                 left = middle + 1;
22             }
23             // If target is smaller, ignore right half
24             else {
25                 right = middle - 1;
26             }
27         }
28
29         // Target not found
30         return -1;
31     }
32
33     public static void main(String[] args) {
34         // Example array
35         int[] array = {2, 3, 4, 10, 40, 50, 60, 70, 80};
```

```
LinearSearch.java U  DichotomousSearch.java U X
DichotomousSearch.java > ...
5  public class DichotomousSearch {
6      public static int binarySearch(int[] array, int target) {
7
8      }
9
10 }
11
12 Run | Debug
13 public static void main(String[] args) {
14     // Example array
15     int[] array = {2, 3, 4, 10, 40, 50, 60, 70, 80};
16
17     int target = 10;
18
19     // Call binary search method
20     int result = binarySearch(array, target);
21
22     if (result == -1) {
23         System.out.println("Element not present in the array");
24     } else {
25         System.out.println("Element found at index: " + result);
26     }
27 }
28
29 PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS  GITLENS
PS D:\kcg\Java> d:; cd 'd:\kcg\Java'; & 'C:\Users\gaire\AppData\Local\Programs\Eclipse Adoptium\jdk-17.0.10.7-hotspot\bin\java.exe' '-Xc:
ceptionMessages' '-cp' 'C:\Users\gaire\AppData\Roaming\Code\User\workspacestorage\cfff8322bbc836dc0e92005c5cab87794\redhat.java\jdk_ws\Java_c
omousSearch'
Element found at index: 3
PS D:\kcg\Java>
```

Gaire Ananta Prasad  
M24W0272

## Flowchart



Gaire Ananta Prasad  
M24W0272

## Pseudocode

✓ Pseudocode of Dichotomous Search method.

Start Initialize the programme

Read  $n$ , the size of array  $A$ .

Compute: find the middle element

Compare  $n$  with the middle element.

If  $n$  matches with middle element, we return the mid index.

Else if  $n$  is greater than the mid element, then  $n$  can only lie in right half Subarray after the mid element. So we recur for right half.

Else ( $n$  is smaller) recur for the left half.

Print the result

End the programme.

Gaire Ananta Prasad  
M24W0272