**DAY-12**

**COLLECTION FRAMEWORK**

**16/05/2025**

Collection Framework

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_|\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | |

List Set Map

| | |

i)ArrayList i)HashSet i)HashMap

ii)LinkedList ii)LinkedHashSet ii)LinkedHashMap

iii)Vector iii)TreeSet iii) TreeMap

List,Set,Map -> these are Interfaces and rest of all are classes

* Interface contains incomplete methods and class contains complete methods
* List is ordered which can allow duplicates
* Set is unordered which wont accept duplicates
* Map contains key-value pairs

**Dutch National Flag**

i) Initialize three pointers l-at 0th index, m-at 0th index, h-at n-1 index

ii) The elements between m and h are unsorted so, we have to traverse between m and h

iii) if arr[m] == 0 then move right-> means swap between arr[m] and arr[l] increase-l and m

iv) if arr[m]==1 just stay there itself just increase m

v) else arr[m] == 2 swap m with h and decrement h

code:

import java.util.Arrays;

class Main {

public static void main(String[] args) {

int[] arr = {1, 2, 0, 0, 0, 1};

System.out.println(Arrays.toString(sorting(arr)));

}

public static int[] sorting(int[] arr) {

int l = 0;

int m = 0;

int h = arr.length - 1;

while (m <= h) {

if (arr[m] == 0) {

int temp = arr[m];

arr[m] = arr[l];

arr[l] = temp;

m++;

l++;

} else if (arr[m] == 1) {

m++;

} else if (arr[m] == 2) {

int temp = arr[m];

arr[m] = arr[h];

arr[h] = temp;

h--;

}

}

return arr;

}

}