9/5/25 (Day 5)

ARRAYS:

Simpler kind of data stored in same place.

collection of simpler kind of data in contious manner is contious allocation

in C/C++ we have reference, pointer(address) but where we can store in java?

it will be stored in heap memory but in heap memory it will store in

- Collection of simpler kind of data in contious allocation.
- In C/C++ we have pointers definetely elements will be stored in continous format
- In java we don't have pointers then their is a chance of elements might store in random places

Syntax

datatype[ ] array name;

Premitive or non-premitive

- Premitive data types can't be divided (eg. int, float, char) , non-premitive data type can be divided(eg. string and all the objects that are created by users).

example:

```
class Node{
   int data;
   node next;
}
```

- All the objects will be stored in heap memory all the references will be stored in stack memory.
- Stack memory having less space compare to heap memory.

```
class Main {
   public static void main(String[] args) {
      datatype[] array name;
      int a=10;
      int b=12;
      int c=14;
      int d=16;
```

```
        int[] arr = {10,12,14,16}
    }
}
```

All the variables are stored in same data type

- instead of creating multiple variables create and array.
- If we create multiple variables to get the data simply we can bring the variable.
- When it comes to array, if we want to print the data we can print using index values and by default index values will start with zero.

Q. in 4 bit memory we have 10,12 and address of second value is address + 4 but where the first value is stored?

It will store in random place and we can’t say where it is stored in heap memory (dyanmic memory)

```
class Main {
    public static void main(String[] args) {
        int[] arr ={1,2,3,4,5);
        int []arr; //declaration
        arr={1,2,3,4,5); //intialization
    }
}
```

Declaration and intialization happens in same line.

in intialization only we use square brackets.

```
class Main {
    public static void main(String[] args) {
        int[] arr ={1,2,3,4,5};
        for(int i=0;i<arr.length;i++){
            System.out.println(arr[i]);
        }

    }
}
```

- To get the length of the array we have a method called arr.length
- in array it is array_name.length and
- in string it is string_name.length()

CAMEL CASE AND SNAKE CASE:

- camel case : arrayName
- snake case : array_name
- in java it is prefered to use camel case to write production level code

in production level we have to write it standare (eg . [void add()] but in production level we have to write void addition_of_two_numbers() but we have to write in camel case so [void addtionOfTwoNumbers()]).

```
void additionOfTwoNumbers(){
} //camel case for production level code.
```

TYPE OF PRINTING FORMAT:

```
for(int i=0;i<arr.length;i++){
        System.out.println(arr[i]= " ");
    }
```

we can write the array values using simple for loop

```
class Main {
    public static void main(String[] args) {
        int[] arr = {1,2,3,4,5};
        for(int i : arr){
          System.out.print(i+" ");
        }

    }
}
//for each loop
```

- in for each loop we don’t need index values

```
import java.util.*;
class Main {
    public static void main(String[] args) {
        int[] arr ={1,2,3,4,5};
        // for(int i=0;i<arr.length;i++){
        //     System.out.println(arr[i]);
        // }

        System.out.print(Arrays.toString(arr));
    }
}
```

- toString() is an inbuild method to print an array.
- toString can accept only one augument and that to array(arr)

Declaring an size of array intilization the values later:

```
int[] arr= new int[5]
        System.out.println(arr[10]);
```

- array having fixed sized if we declared an array size it can’t be changed

```
int[] arr= new int[5];
```

- before=array reference was created(int[ ] array) with the help of new keyword actual object will be created in with the help of heap memory for that particular data type and size.
- if the array type is int all the default values are zero.
- if it is a string it’s default values are null and null is a literal (create null but not assign).

```
int[] arr = new int[5];
arr[0] = 1;
arr[1] = 2;
[2] = 3;
arr[3] =4;
arr[4] = 5;
```

```
import java.util.*;
class Main {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int[] arr= new int[5];
        System.out.println("Enter the values:-");
        for(int i =0;i<arr.length;i++){
            //take the values from user..
        }
        for(int i =0;i<arr.length;i++){
            System.out.print(arr[i]);
        }
        for(int i =0;i<arr.length;i++){
            System.out.print(i);
        }
          System.out.print(Arrays.toString(arr));
      }
      }
```

- in standard form sc is writen as in

```
import java.util.*;
class Main {
    public static void main(String[] args) {
        Scanner in = new Scanner(System.in);
        int[] arr= new int[5];
        for(int i =0;i<arr.length;i++){
            System.out.println("Enter the values:-");
            //take the values from user..
            arr[i]=in.nextInt();
        }
        for(int i =0;i<arr.length;i++){
            System.out.print(arr[i]);
        }
        for(int i =0;i<arr.length;i++){
            System.out.print(i);
        }
            System.out.print(Arrays.toString(arr));
        }
        }
```

Q. Java code to find the count of odd number and even number in array

```
class Main {
    public static void main(String[] args) {
          int[] arr= {1,2,3,4,5,6,7,8};

          int evenCount= 0;
          int oddCount =0;

          for(int num:arr){ // for(int i=0;i<arr.length;i++)
              if(num%2==0){
                  evenCount++;
              }
              else{
                  oddCount++;
              }
```

```
         }
         System.out.println("Even numbers: " + evenCount);
         System.out.println("Odd numbers: " + oddCount);

    }
}
```
\\\\we have index value than use prefered is for loop not for each loop

Q. Print the highest value in array

```
class Main {
    public static void main(String[] args) {
        int arr[]={1,3,6,7,34};
        int heightValue=arr[0];
        for(int i=1;i<arr.length;i++){
            if(arr[i]>heightValue){
                heightValue = arr[i];
            }
        }

        System.out.println(heightValue);
    }
    }
```

Q. Print Second highest value

```
class Main {
    public static void main(String[] args) {
        int arr[] = {1, 3, 6, 7, 34};

        int highest = Integer.MIN_VALUE;
        int secondHighest = Integer.MIN_VALUE;

        for (int i = 0; i < arr.length; i++) {
            if (arr[i] > highest) {
                secondHighest = highest;
                highest = arr[i];
            } else if (arr[i] > secondHighest && arr[i] != highest) {
                secondHighest = arr[i];
            }
        }

        if (secondHighest == Integer.MIN_VALUE) {
            System.out.println("No second highest value found.");
        } else {
            System.out.println(secondHighest);
        }
    }
}


for(int i=0;i<arr.length;i++){
    if(arr[1]>heightValue){
        heightValue=arr[i];
    }
}

for(int i=0;i<arr.length;i++){
    if(arr[1]>secondHeightValue && arr[i]!==heightValue && secondHeightValue !=heightValue){
        secondHeightValue = aa[i];
    }
```

Q. Check wheather the array is sorted or unsorted

Q. Check wheather the given element is there or not

Q. Given array we have elements only 1’s and 0’s, Move all Zero to the right side