OOPJ Notes Day-5 (Date: 02/05/2023)

Session-2 (4:00PM to 6:00PM)

Eclipse introduction

- Kindly use seperrate workspace for your entire OOPJ Module
- make use of packages where ever applicable.
- Apply the Access Modifier whenever needed.
- See the source of EmployeeMgtSystem

implementing toString() method

- It is method of Object class and it can be overridded into any class

```
public class Employee
{
        private int EmpId;
        private String EmpName;
        private long MobileNo;
        private String Address;
        private String EmailId;

        public Employee() {
                super();
        }

        public Employee(int empId, String empName, long mobileNo, String address, String emailId) {
                super();
                EmpId = empId;
                EmpName = empName;
                MobileNo = mobileNo;
                Address = address;
                EmailId = emailId;
        }

        public int getEmpId() {
                return EmpId;
        }

        public void setEmpId(int empId) {
                EmpId = empId;
        }

        public String getEmpName() {
                return EmpName;
        }

        public void setEmpName(String empName) {
                EmpName = empName;
        }

        public long getMobileNo() {
                return MobileNo;
        }

        public void setMobileNo(long mobileNo) {
                MobileNo = mobileNo;
        }

        public String getAddress() {
                return Address;
        }

        public void setAddress(String address) {
                Address = address;
        }

        public String getEmailId() {
                return EmailId;
        }

        public void setEmailId(String emailId) {
                EmailId = emailId;
        }

        @Override
        public String toString() {
                return "Employee [EmpId=" + EmpId + ", EmpName=" + EmpName + ", MobileNo=" + MobileNo + ", Address=" + Address
                                + ", EmailId=" + EmailId + "]";
        }
}
```

Array concept

- It is used to store the homogenous type of elements.
- by using array we get contigous memory allocation. which help us to access the data fastly.

```
public class ArrayDemo {

        public static void main(String[] args) {
                Scanner sc = new Scanner(System.in);
                int arr[]=new int[5]; //Decalartion of Array of int type with fixed size of 5 elements
        for(int i=0;i<arr.length;i++)
        {
                System.out.println("Enter a Int value");
                arr[i]=sc.nextInt();
        }
        System.out.print("Array is:   ");
        for(int i=0;i<arr.length;i++)
        {
                System.out.print(arr[i]+"       ");
        }
        }

}
```

Array using Java

NegativeArraySizeException

-

ArraayIndexOutOfBoundsException

```
public class ArrayDemo {

        public static void main(String[] args) {
                Scanner sc = new Scanner(System.in);
                int arr[]=new int[5]; //Decalartion of Array of int type with fixed size of 5 elements
        for(int i=0;i<arr.length;i++)
        {
                System.out.println("Enter a Int value");
                arr[i]=sc.nextInt();
        }
        System.out.print("Array is:   ");
        for(int i=0;i<6;i++)
        {
                try
                {
                        System.out.print(arr[i]+"       ");
                }
                catch(Exception ex)
                {
                        System.out.println("Exception cathed"+ex.getMessage());
                }
        }
        System.out.println("Program executed Successfully");

        }

}
```

foreach loop

```
public class ArrayDemo {

        public static void main(String[] args) {
                Scanner sc = new Scanner(System.in);
                int arr[]=new int[5]; //Decalartion of Array of int type with fixed size of 5 elements
        for(int i=0;i<arr.length;i++)
        {
                System.out.println("Enter a Int value");
                arr[i]=sc.nextInt();
        }
        System.out.print("Array is:   ");
        for(int i:arr )
        {
        System.out.println(i);
        }
        }
}
```

Major and minor elements of oops

1. Major Elements
    1. Abstraction
    2. Encapsulation
    3. Modularaty (Polymorphism)
    4. Hierarchy (Inheritance and Association)

Association

1. Composition: Kindly do study on that
2. Aggregation: Kindly do study on that

Passing array as a argument

Creating array of primitive values/reference/instances

java.util.Arrays

Multi Dimensional array

Jagged/Ragged array

Enum

Writing menu driven code using Enum

Major and minor elements of oops