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

Coding conventions:

1. Camel Case

- Camel case is a naming convention in which the first letter of each word in a compound word is capitalized, except for the first word. Example: showInputDialog

2. Pascal Case

- Pascal case is a naming convention in which the first letter of each word in a compound word is capitalized. Example: ArrayIndexOutOfBoundsException

3. Snake Case

- Snake case combines words by replacing each space with an underscore (_). Example:MAX_VALUE

4. Kebab Case

- Kebab case is the way to write compound words separated by hyphens (-) instead of using space. Generally, everything is written in lowercase. Example: “what-is-kebab-case”

Final modifier

- Final Variable: A variable whose value cant be changed once initialized
- Final Feild: A Feild of class whose value cant be changed once initialized
- Final Method: A method of class that can't be overridden
- Final Class: A class which can't be inherited.

Absolute path and relative path

- Absolute Path: complete path of the file from its root directory
- Relative Path: path info w.r.t. to current, next and previous directory
- Reference: https://www.redhat.com/sysadmin/linux-path-absolute-relative (https://www.redhat.com/sysadmin/linux-path-absolute-relative)

Path and classpath

- path: used by OS to locate the application in the system.
- classpath: used by javac, javap or javadoc or any other application to locate
- Reference: https://docs.oracle.com/javase/tutorial/essential/environment/paths.html (https://docs.oracle.com/javase/tutorial/essential/environment/paths.html)

Demo of Classes (Scanner, Date, Calendar, LocalDate, LocalTime, LocalDateTime and SimpleDateFormat)

- Scanner class is used to get input from keyboard into your program.
- Date: This is oldest class in java from jdk1.1.
- LocalDate: This class is present in java.time package.

```
import java.util.InputMismatchException;
import java.util.Scanner;
class DemoOfScanner {

    public static void main(String[] args) {
        Scanner sc= new Scanner(System.in);
        int a=0;
        System.out.println("Enter a int value");
        try{
            a=sc.nextInt();
        }
        catch(InputMismatchException ex)
        {
            System.out.println("Exception Catched"+ex.getMessage());
        }
        System.out.println("Program closed normally");

        System.out.println("Entered value is:   "+a);
    }

}
import java.util.Scanner;
class Student
{
    int RollNo;
    String Name;
    String Address;

    void Setdata()
    {
        Scanner sc= new Scanner(System.in);
        Scanner scstr=new Scanner(System.in);
        System.out.println("Enter the Roll No:  ");
        RollNo=sc.nextInt();
        System.out.println("Enter the Name:     ");
        Name= scstr.nextLine();
        System.out.println("Enter the Address:     ");
        Address= scstr.nextLine();
    }
    void Display()
    {
        System.out.println("Student Information");
        System.out.println(RollNo+" "+Name+" "+Address);
```

```
    }

}
class DemoOfSc2Test {
public static void main(String[] args) {
    Student s1=new Student();
    s1.Setdata();
    s1.Display();
}



}
import java.time.LocalDate;
class Date{
    int day;
    int month;
    int year;

    Date()
    {
        LocalDate ld= LocalDate.now();
        this.day=ld.getDayOfMonth();
        this.month=ld.getMonthValue();
        this.year=ld.getYear();
    }
}

class DemoofDate {
    public static void main(String[] args) {
        Date dt=new Date();
        System.out.println(dt.day+" "+dt.month+" "+dt.year);



    }

}
```

java.lang.Object class introduction

- In Java, parent class is called as super class and child class is called as sub class.
- Object is a concrete class declared in java.lang package.

- java.lang.Object do not extend any class or do not implement any interface. In other words, it is super class of all the classes(not interfaces) in core Java.
- It is also called as ultimate base class / super cosmic base class / root of java class hierarchy.
- Reference: https://docs.oracle.com/javase/8/docs/api/java/lang/Object.html (https://docs.oracle.com/javase/8/docs/api/java/lang/Object.html)

constructor chaining

- To reuse body of exisiting constructor we can call constructor from another constructor. It is called as constructor chaining.
- For constructor chaining we should use this statement.
- this statement must be first statment inside constructor.

```
class DemoOfConsChain {
    int a;

    DemoOfConsChain()
    {
        this(10); //Valid
        System.out.println("No Argu Cons");
        //this(10); //Not Valid
    }
    DemoOfConsChain(int p)
    {
        this(100, 200);
        System.out.println("One Argu Cons");
    }
    DemoOfConsChain(int q, int r)
    {
        System.out.println("Two Argu Cons");
    }
}
class DemoOfConsChainTest
{
    public static void main(String[] args) {

        DemoOfConsChain d1=new DemoOfConsChain();
    }
}
```

NullPointerException

- It may comes when there is no instance ref available in ref variable

```
class DemoOfNPE {
    int a;
    void Show()
    {
        System.out.println(a);
    }
}
class DemoOfNPETest
{
    public static void main(String[] args) {

        DemoOfNPE d1=new DemoOfNPE();
        d1.Show();
        d1=null;
        d1.Show();

    }
}
```

Package, import and static import

```
package com.cdac.p1;
import java.util.*; //Non-static import of java.util package
import static java.lang.Math.PI; //Static import for the value of pi
class DemoPack {
    public static void main(String[] args) {
        System.out.println("Hello World");
        int area=(int)PI*22;
    }

}
```