Monday, April 25, 2022 9:58 AM

java.util. Random

This is a random number generator .

Inheritance ----- IS A relation

To establish parent child relation ----extends

When we create Employee object -------- Person object is created first and then the employee object is created!!!

Constructor Calling Sequence = Object Creation Sequence

Super most class constructor is called first , followed by next sub class ,…..followed by the current subclasses

Multi LEVEL Hierarchy ------------- D isa C , C isa B , B isa A }} super most class is A

```
A
|
B
|
C
|
D
```

One class can have many subclasses

C isa B
E isa B

One class CANNOT have many super classes !!! THAT means one class can have ONLY ONLY ONLY 1 super class
That means

JAVA does not support MULTIPLE inheritance using extends KEYWORD !!!!

If B isa A then I cannot say B isa Z also (NOT ALLOWED)

One advantage of Inheritance is that -- sub class can REUSE the methods and properties of super class.

---

super() = this calls the no-parameter/default constructor of the super class
= If written ,this line MUST be the first line of the code in CHILD constructor
= If not written , the compiler will add it on its own automatically

super.show()   = this will call the show method of the super class
               = super keyword ACTS as a "this" of Person (SUPER CLASS)

---

java.lang. Object   class.

Object class is the super class of all the classes by default . ROOT of java object hierarchy!!!
We do not write extends Object but still Object becomes the super class of every class by default

Object class methods

1	getClass	
2	clone()	
3	toString()	it returns a string made up of **package qualified class name @ hex address** When we put obj in sysout System.out.println(obj) internally obj.toString() is called and the result is displayed.
4	equals()	
5	hashcode()	
6	wait()	
7	notify	
8	notifyAll	
9	finalize()	

OVERRIDE A METHOD ??????
Writing another implementation in the sub class of the same signature method already present in superclass!!!!

HW ----- Override toString in Employee class , Person class , TechnicalBook class !!!
Create the respective objects in TestObject and show them in sysout .

Overloading ----- constructor overloading , function overloading
Overriding ---- method/function overriding

POLYMORPHISM ----- POLY = many , MORPHS = forms/ implementations of methods !!!!

TWO types of POLYMORPHISM ----------------------------------

1. **Overloading** = One method name MANY forms , BUT the signatures are different

```
public   MyDate(){   …………………}

public MyDate( int d, int m, int y )
       {  ……………..}
```

2. **Overriding =** One method signature , MANY forms , Signatures are same

```
Object class
        public String toString()
             {
                    return  packagequalifiedclassname @ hex
             }

MyDate class
              public String toString()
             {
                    return  day+"/"+month+"/"+year;
             }
```

____

Overloading	Overriding
We can write all the forms/implementations in one CLASS	We can write all the forms/implementation in super and sub classes of inheritance
method name is SAME but signatures MUST be different	method signatures must be same return type, name and parameter list must be same
This is called as COMPILE time polymorphism This is also called as static polymorphism	This is called as RUN time Polymorphism !!! This is called as dynamic polymorphism

____

We create a hierarchy

```
Study.isa. TestInheritanceConcepts
{
}
```

create a hierarchy

```
Alpha   ----  fa()  ---- sysout Alpha
Beta isa Alpha   fb()  ---- sysout Beta
Theta isa Beta    fc ()  ---- sysout Theta
Gama isa Theta   fd()  ---- sysout  Gama
Delta isa Beta
```

When we create an object

The object can call all its visible super class methods

+

Its own methods

HW --- complete the Alpha hierarchy assignment discussed in lecture

HW ---- add a class study.isa.Patient in the Person hierarchy

    Patient is a Person
        Properties ---- bloodGroup , bp , heartRate
        Write 2 constructors getters and setters and override toString

    Write a class User2 in same package
        Create patient objects using parameterized constructors
        Show all patient details using toString
        Show patient name and bp only using getters

HW ---------- add a class study.isa.PartTimeEmployee in Person Hierarchy

    PartTimeEmployee **isa Employee**
        Numbers of hours
         write 2 constructors getters setters and override tostring

    Write a class User3
            Main
                create ParttimeEmployee objects using parametrized constructors
                 show all details using toString
                show name , department and number hours using getters