Drives -dependency

Car

-color:text

-model:text

Employee

-name:text

-age: int

Has a

Association

EmpClient

UI

Business Layer

Employee

Employee sam = **new** Employee();

stack

sam

name age

Heap memory

Car car1 = new Car(“blue”,”Blue Alto”);

Create a class Car with color and model as attributes.

Generate constructors,getters and setters

George has and drives to ABC Ltd,Bangalore in his blue alto car.

Smith has and drives to XYZ Ltd ,Hyderabad in his red santro car

Create a class Company with attribute (name,location)

company - Employee

company - Client

reference

ABC Ltd

Bangalore

Static members

Employee -- empid,name,age,dob -- instance variables -- sam,peter,john

Class Employee

{

private static int count;

public String getName(){ …} // instance method

}

public static void printCount(){ ..}

Static methods can call /access static members within the same class or with the class name in a different class.

Integer.parseInt();

Date d=new Date();

d.getYear();

Method Overloading : 2 or more methods with same name but different signatures.

Static /compile time polymorphism

Inheritance – A family of Classes – Reusability

Person

name,address

Not supported

C

B

A

Privileged Customer

Is -a

Teacher

Student

extends

Customer

public class Pearl

{

//color size shape

}

public class Necklace

{

// array of Pearl

}

Day 2

Object class is the super class in java library.

Java.lang --- java.lang.Object

class Person extends Object

{

}

class Employee

Teacher

dept , subjects

Student

regNo

Person

name,address

Method Overloading

Drives()

Drives(Car car)

Inheritance:

1. Private members are not inherited
2. Constructors are not inherited
3. Static members are not inherited.

MethodOverriding : An existing method of the base class is recreated/redefined in the child class.

final – create constants, prevent overriding , prevent inheritance

toString()

1. Protoypes have to be same :
   1. String toString()
2. overridden method cannot be restrictive (Access specifiers)

Employee – drives() drives(Car car)

Employee sam = new Employee(….);

Car car1=new Car(..);

sam.drives(); // binding of object with method drives during compilation

sam.drives(car1); // binding of object with method drives during compilation

Polymorphism – Compile time / static -- Method Overloading

Class Demo

User

name

giveOffers

Student

regno

Employee

empId

Customer

email

* Abstract method is a specification in a class that does not have any body for the method.
* Abstract class is a class that has one or more abstract methods.
* Abstract method must and should be overridden in the child class wo which the class will not compile
* Abstract class/method helps in dynamic/runtime polymorphism through method overriding
* Abstract class cannot be instantiated
* Abstract class can have concrete methods also.
* Abstract method/class cannot be final
* Ab method/class can not be static

Abstract Class User

{

public abstract void a();

}

Class B extends Thread

{

}

interface Bird

{

public void fly();

}

Interface Animal

{

public void walks();

}

Class Employee extends Person implements Bird,Animal

{

public void fly(){ }

public void walks(){ }

PaymentFactory

}

PaymentClient

Payment

Payment (I)

makePayment(double)

UPI payment

CashPayment

CardPayment

Paymen

Client

Day 3

Throwable

Exception

ArrayIndexOutOfBoundsException

ArithmeticException

NumberFormatException

NullPointerException….

RuntimeException

(Unchecked Exceptions)

1. Try block is followed by a catch block or finally block
2. Try block can have multiple catch blocks
3. Catch blocks should follow proper hierarchy of Exception classes
   1. ie.. generic block follows other catch blocks
4. Unchecked exceptions are not checked during compilation time. Program compiles
5. Checked Exceptions are those that have to be handled ,wo which program will not compile
   1. Accessing a database
   2. Accessing a file – IO
   3. Socket

Collections API/framework

Array – int a[]=12,23,44,12,56

String names[] = new String[10];

Array limitations

1. Size is fixed.
2. Cumbersome operations

Collection (I)

Set (I)

Unordered

No duplicates, no index

Not indexed

List(I)

Ordered [7,8,4,3]

Duplicates [7,8,7,4,8]

Indexed list.get(3)

LinkedHashSet

Ordered – [15,7,8]

No duplicates

TreeSet

HashSet

SortedSet (I)

sorted

LinkedList

ArrayList

Vector

Map(I) – Key,Value

Searching is easier

Keys are unique, unordered and no duplicates

TreeMap

HashMap

LinkedHashMap -ordered

HashTable - legacy

SortedMap(I)

Sorted keys

BookStore Case Study

Book

BookStore

Comparable – compareTo(Object o)

Comparator – compare(Object o1,Object o2)