CHARUSAT
CHAROTAR UNIVERSITY OF SCIENCE AND TECHNOLOGY

CE251: Java PROGRAMMING

July – November 2020

Chapter – 4

OOPs Inheritance

DEPSTAR
Devang Patel Institute of Advance Technology and Research

What are the OOPs Concepts?

1. Inheritance
2. Polymorphism
3. Abstraction
4. Encapsulation
5. Class
6. Object

Class Vs Object??

What do you mean by Inheritance?

Parent class is providing properties and child class is acquiring properties

Parent & child relationship is called inheritance

Why important??

```
class A
{
void m1(){}
void m2(){}
}
```

```
class B
{
void m1(){}
void m2(){}
void m3(){}
void m4(){}
}
```

```
class C
{
void m1(){}
void m2(){}
void m3(){}
void m4(){}
void m5(){}
void m6(){}
}
```

Code Duplication- length of code increases

Important Advantages of Inheritance

- Reduce the length of code
- Reduce the redundancy of the application

How??

By using extends keyword

Example

```
class A
{
void m1(){}
void m2(){}
}
```

```
class B extends A
{
void m3(){}
void m4(){}
}
```

```
class C extends B
{
void m5(){}
void m6(){}
}
```

IS it allow in Java?

```
class C extends A, B
{
void m5(){}
void m6(){}
}
```

NO

Java do not support Multiple inheritance

How to execute all methods?

```
A a = new A();
a.m1();
a.m2();
```

```
B   b = new B();
b.m1();
b.m2();
b.m3();
b.m4();
```

```
C   c = new C();
c.m1();
c.m2();
c.m3();
c.m4();
c.m5();
c.m6();
```

Which one is better?

Child class object can access parent as well as child class properties

Types of Inheritance

1. Single Inheritance
2. Multi Level Inheritance
3. Multiple Inheritance
4. Hierarchical Inheritance
5. Hybrid Inheritance

Java Support only 3 types

1. Single Inheritance
2. Multi Level Inheritance
3. Hierarchical Inheritance

How to prevent Inheritance?

For security reason I want to prevent Inheritance

By using final keyword

Example

```
final class A
{
void m1(){}
void m2(){}
}
```

```
class B extends A
{
void m3(){}
void m4(){}
}
```

Not possible to create child class

How to represent Parent class member?

By using super keyword

Example-1

```
class Parent
{
int a =100;
int b =200;
}
```

```
class Child extends Parent
{
 int x =10;
 int y =20;
void add(int i, int j)
{
    S.O.P(i+j);
    S.O.P(x+y);
    S.O.P(a+b);
}
P.S.V.M()
{
new Child().add(1,2);
}}
```

Example-2

```
class Parent
{
int a =100;
int b =200;
}
```

```
class Child extends Parent
{
 int a =10;
 int b =20;
void add(int a, int b)
{
    S.O.P(a+b);
    S.O.P(a+b);
    S.O.P(a+b);
}
P.S.V.M()
{
new Child().add(1,2);
}}
```

How can I print Parent class as well as Child class instance variable?

Using this to print current class variable

Using super to print Parent class variable

```
S.O.P(a+b);
S.O.P(this.a + this.b);
S.O.P(super.a + super.b);
```

Example-3 (Parent Class Method)

```
class Parent
{
void m1()
{
S.O.P("Parent class
method");
}
}
```

```
class Child extends Parent
{
 void m1()
{
S.O.P("Child class method");
}
void m2()
{
m1();
m1();
}
P.S.V.M()
{
new Child().m2();
}}
```

How to call parent class method?

```
void m2()
{
    this.m1();
    super.m1();
}
```

Parent class constructor

```
class Parent
{
Parent()
{
S.O.P("Parent class
constructor");
}
}
```

```
class Child extends Parent
{
 Child()
{
  this(1);
S.O.P("Child class constructor");
}
Child(int a)
{
  super();
S.O.P("1-argument child class
constructor");
}
P.S.V.M()
{
   new Child();
}}
```

Small modification in Child class constructor

```
Child(int a)
{
   S.O.P("1-argument child class constructor");
   super();
}
```

Will it compile or not?

Compiler error

One more modification in Child class constructor

```
Child()
{
  this(1);
  super();
   S.O.P("Child class constructor");
}
```

Will it compile or not?

Compiler error

What is the output?

```
class Parent
{
Parent()
{
S.O.P("Parent class
constructor");
}
}
```

```
class Child extends Parent
{
 Child()
{
S.O.P("Child class
constructor");
}
P.S.V.M()
{
   new Child();
}
}
```

What is the output?

```
class Parent
{
   Parent()
   {
     S.O.P("Parent class constructor");
   }
}
class Child extends Parent
{
   P.S.V.M()
   {
      new Child();
   }
}
```

Program it.

MotorVehicle // class name

String modelName;
int modelNumber;
float modelPrice;

MotorVehicle(mName, mNumber, mPrice) //constructor

void display() // print values

Car // class name

int discountRate;

Car(name, mNumber, mPrice, discountRate) // constructor
void display()
void discount()

public static void main(){
Call constructor by creating object of Car class
Call display method;
Call discount method;}

Any Question??

Q.01

```
class Top {

public Top(String s)
{ System.out.print("B"); }
}

public class Bottom2 extends Top {

public Bottom2(String s)
{ System.out.print("D"); }

public static void main(String [] args) {

new Bottom2("C");

}}
```

What is the result?

A. BD

B. DB

C. BDC

D. DBC

E. Compilation fails

Q.02

```
class Top {

public Top()
{ System.out.print("B"); }
}

public class Bottom2 extends Top {

public Bottom2(String s)
{ System.out.print("D"); }

public static void main(String [] args) {

new Bottom2("C");

}}
```

What is the result?

A. BD

B. DB

C. BDC

D. DBC

E. Compilation fails

Q.03

```
class Building {

Building()
{
      System.out.print("b ");
}

 Building(String name) {
  this();
  System.out.print("bn " + name);
}
}
```

What is the result?

A. h hn x B. hn x h

C. b h hn x D. b hn x h E. bn x h hn x

```
public class House extends Building {
  House()
  {
       System.out.print("h ");
  }
 House(String name) {
  this();
  System.out.print("hn " + name);
}
 public static void main(String[] args)
 {
       new House("x");
 }
 }
```

Q-4

```
class A
{
   {
      System.out.println(1);
   }
}

class B extends A
{
   {
      System.out.println(2);
   }
}

class C extends B
{
   {
      System.out.println(3);
   }
}
```

Output :

1

2

3

```
public class MainClass
{
   public static void main(String[] args)
   {
      C c = new C();
   }
}
```

Q-5

```
class A
{
   String s = "Class A";
}

class B extends A
{
   String s = "Class B";
    {
      System.out.println(super.s);
   }
}

 class C extends B
{
   String s = "Class C";
    {

      System.out.println(super.s);
   }
}
```

```
public class MainClass
{
   public static void main(String[] args)
   {
      C c = new C();

      System.out.println(c.s);
   }
}
```

Output :

Class A

Class B

Class C