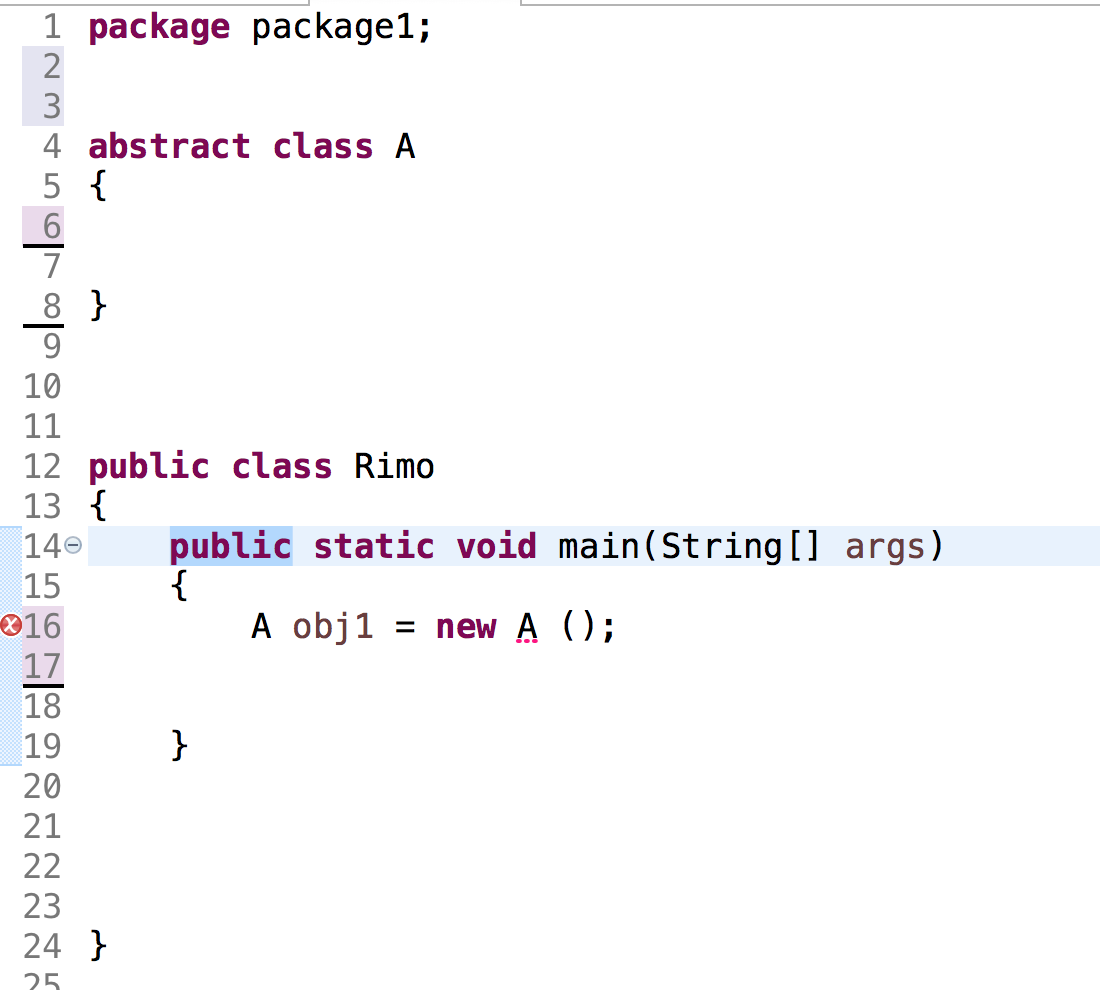
**ABSTRACT CLASS**

1. **Can’t create object of abstract class**

****

1. **A class is called abstract when**
2. **When it contains only abstract methods**
3. **When it contains only concrete methods**
4. **When it contains only abstract methods and concrete methods both**

// only abstract method

**abstract** **class** A

{

**abstract** **void** sum ();

**abstract** **void** run ();

**abstract** **void** cun ();

}

//only concrete method

**abstract** **class** B

{

**void** area ()

{}

**void** volume ()

{}

**void** square ()

{}

}

// both abstract and concrete

**abstract** **class** C

{

**void** area ()

{}

**abstract** **void** team ();

}

1. **Extensions**
2. **A class can extends abstract class**

**Condition >> has to give body to the abstract methods**

**abstract** **class** A

{

**abstract** **void** sum ();

**abstract** **void** run ();

**void** area ()

{

System.***out***.println(“parent”);

}

}

**class** B **extends** A

{

**void** sum () // abstract method of A

{

}

**void** run () // abstract method of A

{

}

**void** volume () // own specialized method class B

{

}

**void** area () // overriding method (optional)

{

}

}

1. **An abstract class can extends another abstract class**

**>> it is not necessary to give body to its abstract method of abstract parent class**

**(but can give if we want to)**

**abstract** **class** A

{

**abstract** **void** sum ();

**abstract** **void** run ();

**void** area ()

{

System.***out***.println("parent");

}

}

// abstract class(B) extends

// abstract class (A)

**abstract** **class** B **extends** A

{

}

**abstract** **class** D **extends** B

{

**void** run ()

{

}

}

**Note : If above parent classes already provide body to the abstract then the extend child class need not to give**

**body abstract methods of parent classes**

**abstract** **class** A

{

**abstract** **void** sum ();

**abstract** **void** run ();

**void** area ()

{

System.***out***.println(“grand parent”);

}

}

**abstract** **class** B extends A

{

**abstract** **void** fun ();

**void** sum () // abstract method of A

{

System.***out***.println(“parent”);

}

}

**class** C **extends** A // **( multi-level )**

{

**void** run () // abstract method of A

{

}

**void** fun () // abstract method class B

{

}

// class C don’t have to give body to sum method

// as already given by above parent class(B)

}

**class** D **extends** A // **( heirharchy )**

{

**void** run () // abstract method of A

{

}

**void** sum () // abstract method class A

{

}

1. **A Class can extends abstract class**
2. **Abstract class can extends class**

**abstract** **class** A

{

**abstract** **void** sum ();

**abstract** **void** run ();

**void** area ()

{

System.***out***.println("parent");

}

}

**abstract** **class** B **extends** A

{

}

**abstract** **class** D **extends** B

{

**abstract** **void** tum ();

**void** run ()

{

}

}

// class(E) extends abstract class (D)

**class** E **extends** D

{

**void** sum ()

{

}

**void** tum ()

{

}

}

// abstract(G) class extends class (E)

**abstract** **class** G **extends** E

{

**abstract** **void** jump ();

}

1. **A class cannot extends 2 abstract classes**

**abstract** **class** A

{

}

**abstract** **class** B **extends** A

{

**abstract** **void** run ();

}

// A class (C) cannot extends 2 abstract

**class** C **extends** B,A >> Error

{

**void** run ()

{

}

**void** fun ()

{

}

}

**A class cannot extends 2 non abstract class**

**class** A

{

}

**class** B **extends** A

{

**abstract** **void** run ();

}

// A class (C) cannot extends 2 classes

**class** C **extends** B,A >> Error

{

}

**\*\* Diamond Problem**

**package** abs;

**abstract** **class** A

{

**int** x = 5;

}

**abstract** **class** B

{

**int** x = 10;

}

**class** C **extends** A,B // Error (cannot extend 2 classes)

{

}

**public** **class** Fun {

**public** **static** **void** main(String[] args)

{

C obj = **new** C();

System.***out***.println(obj.x);

// ambiguity occurs.

// as compiler doesn’t know which x to call A class or B class

System.***out***.println(A.x); // no error >> 5

System.***out***.println(B.x); // no error >> 10

}

}

**INTERFACE**

1. **Interface is not a class**
2. **We cannot create object of interface**
3. **Methods inside Interface are inbuild abstract**
4. **Variables inside Interface are inbuild final**
5. **Interface does not have concrete methods**
6. **A class implements interface**

**Condition >>**

**1)has to give body to the abstract methods**

**2)method should be declared as public**

**interface** A

{

**int** ***x*** = 5 ; // final

**void** area(); // abstract

}

**class** B **implements** A

{

**public** **void** area()

{

}

}

1. **A class can implements two interfaces**

**Condition >>**

**1)has to give body to the abstract methods of both interfaces which are inheritate**

**2)method should be declared as public**

**interface** A

{

**int** ***x*** = 5 ;

**void** area();

}

**interface** B

{

**void** volume();

}

// class C implements A and B interfaces

**class** C **implements** A,B

{

**public** **void** area()

{

}

**public** **void** volume()

{

}

}

**interface** A

{

**void** volume();

}

**interface** B

{

**void** add();

}

**Class C**

{

}

**// Interface cannot implements interface**

**interface** A **implements** B // Error

{

}

**// Interface cannot implements another class**

**interface** C **implements** C // Error

{

}

// **Interface cannot extends another class**

**interface** D **extends** C // Error

{

}

1. **Interface can extends another interfaces**

**Condition >>**

**Class which extends**

**1)has to give body to the abstract methods of interfaces which are inheritate**

**2)method should be declared as public**

**interface A**

{

**int** ***x*** = 5 ;

**void** area();

}

**interface B extends A**

{

**void** volume();

}

**// Class C extends B**

**// As class B extends A**

**// C has to give body to both abstract class**

**// of A and B**

{

**public** **void** area()

{ }

**public** **void** volume()

{ }

}

**// Class C extends A**

**// C has to give body to only abstract class**

**// of A**

**Class D extends A**

{

**public** **void** area()

{

}

}

**Interface E**

{

**void** add();

}

**// Class G extends A and E**

**// G has to give body to abstract class**

**// of A and E**

**Class G extends E,A**

{

**public** **void** area()

{ }

**public** **void** add()

{ }

}

1. **Abstract class can implements interfaces**

**interface** A

{

**int** ***x*** = 5;

**void** area();

}

**interface** B

{

**void** volume();

**void** run();

}

**abstract** **class** D **implements** B,A

{

**abstract** **void** tum ();

**void** run()

{

}

}

**// We cannot create object for A B D**

**// For class E we can create object**

**class** E **extends** D

{

**public** **void** volume()

{

}

**public** **void** area()

{

}

**public** **void** tum()

{

}

}

**class can extends class and implements interface**

**Condition >> first extends than implements**

**interface** A

{

**int** ***x*** = 5 ;

**void** tum();

//void area();

}

**interface** B

{

**void** volume();

}

**class** C

{

}

**class** D **extends** C **implements** B

{

**public** **void** volume ()

{

}

}

**class** E **implements** A **extends** C // Error

{

**public** **void** volume ()

{

}

}

**class** F **extends** C **implements** B,A

{

**public** **void** volume ()

{

}

**public** **void** tum ()

{

}

* Solving Diamond Problems

**interface** A

{

**int** ***x*** = 5;

}

**interface** B

{

**int** ***x*** = 10;

}

**class** C **implements** A,B

{

**int** x = 15;

}

**public** **class** Play

{

**public** **static** **void** main(String[] args) **throws** Exception

{

C obj = **new** C ();

System.***out***.println(obj.x);

System.***out***.println(A.***x***);

System.***out***.println(B.***x***);

}

}

1. class can extends class >> true
2. class cannot extends 2 classes >> false
3. abstract class can extends class >> true
4. class can extends abstract class >> true
5. abstract class can extends abstract class >> true
6. class can extends 2 abstract classes >> false
7. class can implement interface >> true
8. interface extends interface >> true
9. interface can extends 2 interface   >> true
10. abstract class can extend interface >> true
11. abstract class can extend 2 interface >> true
12. interface can extends class >>> false
13. class can implements class >>> false
14. interface can implements interface >>> false