Unit 2

Unit 2 - Contents

Classes and Objects

- Class – defines the structure of an object and its functional interface known as methods
- Basic form

```
class classname[extends superclassname]{
type instance_variable1;
type instance_variable2;
type instance_variableN;
type methodname1(parameter-list)
{
method-body}
type methodname2(parameter-list)
{method-body}
type methodnameN(parameter-list)
{method-body}
}
]
```

Classes and Objects

Example

```
class Rectangle{
int length, breadth;
void getData(int a, int b)
{ length=a;
  breadth=b;
}
}
```

Classes and Objects

Creating Objects - syntax

```
class object = new class(parameter_list)
```

Example

```
Rectangle r = new Rectangle()
```

this operator

- Special reference value called this which is used inside any method to refer to the current object.
- Example Program

Constructor

- A method which initializes an object immediately upon creation.
- Same name as that of the class.
- Do not specify a return type. Returns the instance of the class itself.
- Example Program

Method Overloading

- Same name but different parameter lists and definitions.
- Shows polymorphism.
- Provide several different method definitions in the class with the same name but with different parameter lists.
- Example program

Inheritance

- Mechanism of deriving a new class from an old one.
- Old class is called the base class, parent class or super class.

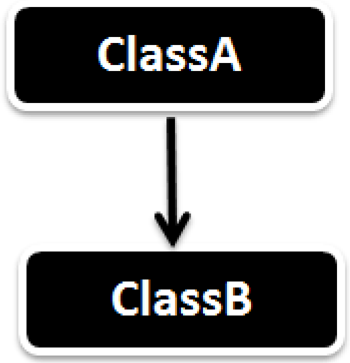
Syntax

```
class classname extends superclassname{
[variables declaration;]
[methods declaration;]
}
```

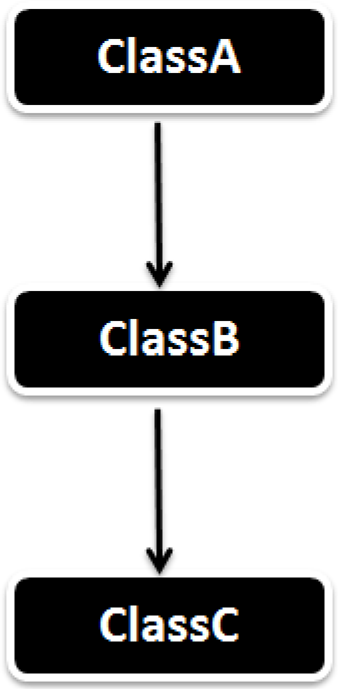
- Does not support multiple inheritance. For multiple inheritance java uses interface.

Example Program

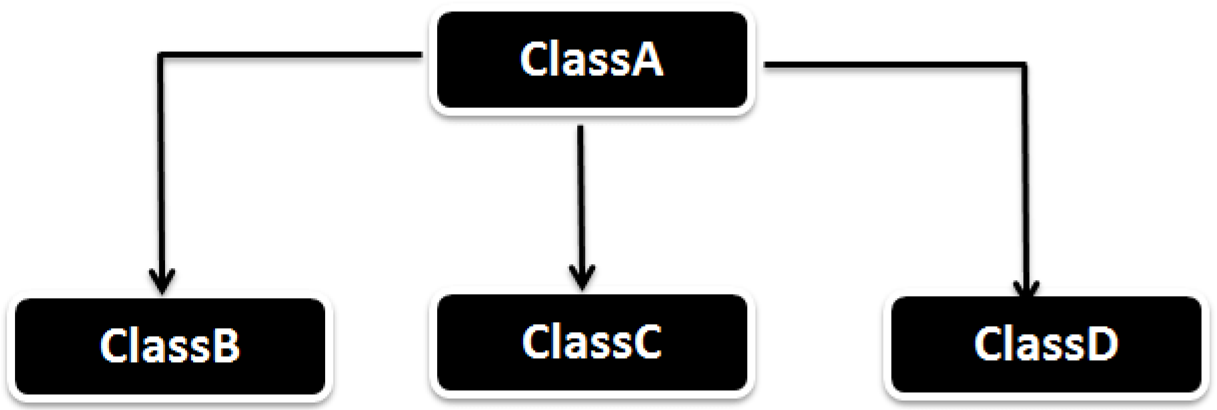
Single Inheritance



Multilevel Inheritance



Hierarchical Inheritance



Super

- Variable super refers directly to the superclass constructors.
- Example Program

Subclass constructor

- Used to construct the instance variables of both sub class and super class.
- Subclass constructor uses the keyword super to pass values to the super class.

Subclass constructor

- Super is used subject to the following conditions.

1. Only used within a subclass constructor method.
2. Call to the super class constructor must appear as the first statement within the subclass constructor.
3. Parameters in the super call must match the order and type of the instance variable declared in the super class.

Method Overriding

- Two methods with the same name, same arguments and same return type in the sub class and super class.
- When this method is called, method defined in the sub class is invoked and executed.

Example Program

Dynamic Method Dispatch

- Equivalent to virtual functions in C++
- At runtime, the object reference could be referring to an instance of some subclass.
- In this case the method in the subclass method will be invoked.

Example Program

Final

- If the methods and variables are declared final, we can prevent overriding.
- Example

```
final int  s=10;
final void showstatus(){
}
```

- If a class is declared final, it can be prevented from inheritance.
- Example

```
final class A{
}
```

finalize()

- Destructors in java.
- Java runtime frees up the memory used by objects.
- But objects may hold other non-object resources such as system fonts. Garbage collection cannot free these resources.
- finalize() can be added to any class.

static

- Static methods can be called without using the objects.
- Static methods are called using class names.
- Static methods are available for using by other classes.
- Static methods may only call other static methods directly and they should not refer to this or super.
- This is how java implements global functions.

Example Program

Abstract Classes & Abstract Methods

- Define a class without providing the complete implementation of every method.
- Any class which contains abstract methods should also be declared abstract.
- To declare a class abstract, the keyword abstract should be used in front of the class.

Abstract Classes & Abstract Methods

- Cannot have abstract constructors or abstract static methods.
- Abstract classes cannot be directly instantiated with the new operator.
- Any subclass of an abstract class must either implement all of the abstract methods in super class or itself be declared abstract.
- Example Program

Interfaces

- Java does not support multiple inheritance.
- Java does not have more than one super class.
- Provides an alternate approach known as interfaces to support multiple inheritance.

```
class A extends b extends C{
}  // not valid in Java
```

Defining Interfaces

- Interface is basically a kind of class.
- Interfaces contains methods and variables but with a difference.
- Interface defines only abstract methods and final fields.

Syntax

```
interface InterfaceName{
Variables declaration;
Methods declaration;}
```

Example

```
interface Area{
static final float pi =3.14f;
float compute(float x, float y);
}
```

Implementing Interfaces

- Interfaces are used as superclasses whose properties are inherited by classes.

Syntax

```
class classname extends superclass implements
    interface1, interface2{
Body of classname
}
```

Example Program

End of Unit 2