Abstraction

Abstract Class

- A class which contains the **abstract** keyword in its declaration is known as abstract class.
  - Abstract classes may or may not contain *abstract methods* ie., methods with out body (public void get();)
  - But, if a class have at least one abstract method, then the class **must** be declared abstract.
  - If a class is declared abstract it cannot be instantiated.
  - To use an abstract class you have to inherit it from another class, provide implementations to the abstract methods in it.
  - If you inherit an abstract class you have to provide implementations to all the abstract methods in it.

```java
/* File name : Employee.java */
public abstract class Employee
{
   private String name;
   private String address;
   private int number;
   public Employee(String name, String address, int number)
   {
      System.out.println("Constructing an Employee");
      this.name = name;
      this.address = address;
      this.number = number;
   }
   public double computePay()
   {
     System.out.println("Inside Employee computePay");
     return 0.0;
   }
   public void mailCheck()
   {
      System.out.println("Mailing a check to " + this.name
       + " " + this.address);
   }
   public String toString()
   {
      return name + " " + address + " " + number;
   }
   public String getName()
   {
      return name;
   }
   public String getAddress()
   {
      return address;
   }
   public void setAddress(String newAddress)
   {
      address = newAddress;
   }
   public int getNumber()
   {
     return number;
   }
}
```

```
/* File name : AbstractDemo.java */
public class AbstractDemo
{
   public static void main(String [] args)
   {
      /* Following is not allowed and would raise error */
      Employee e = new Employee("George W.", "Houston, TX", 43);

      System.out.println("\n Call mailCheck using Employee reference--");
      e.mailCheck();
    }
}
```

When you compile the above class, it gives you the following error:

```
Employee.java:46: Employee is abstract; cannot be instantiated
      Employee e = new Employee("George W.", "Houston, TX", 43);
                   ^
1 error
```

Inheriting Abstract class

```
/* File name : Salary.java */
public class Salary extends Employee
{
   private double salary; //Annual salary
   public Salary(String name, String address, int number, double
      salary)
   {
       super(name, address, number);
       setSalary(salary);
   }
   public void mailCheck()
   {
       System.out.println("Within mailCheck of Salary class ");
       System.out.println("Mailing check to " + getName()
       + " with salary " + salary);
   }
   public double getSalary()
   {
       return salary;
   }
   public void setSalary(double newSalary)
   {
       if(newSalary >= 0.0)
       {
           salary = newSalary;
       }
   }
   public double computePay()
   {
      System.out.println("Computing salary pay for " + getName());
      return salary/52;
   }
```

```
   public static void main(String [] args)
   {
      Salary s = new Salary("Mohd Mohtashim", "Ambehta, UP", 3, 3600.00);
      Employee e = new Salary("John Adams", "Boston, MA", 2, 2400.00);

      System.out.println("Call mailCheck using Salary reference --");
      s.mailCheck();

      System.out.println("\n Call mailCheck using Employee reference--");
      e.mailCheck();
    }
}
```

This produces the following result:

```
Constructing an Employee
Constructing an Employee
Call mailCheck using  Salary reference --
Within mailCheck of Salary class
ailing check to Mohd Mohtashim with salary 3600.0

Call mailCheck using Employee reference--
Within mailCheck of Salary class
ailing check to John Adams with salary 2400.
```

Abstract Methods

- If you want a class to contain a particular method but you want the actual implementation of that method to be determined by child classes, you can declare the method in the parent class as abstract.
  - **abstract** keyword is used to declare the method as abstract.
  - You have to place the **abstract** keyword before the method name in the method declaration.
  - An abstract method contains a method signature, but no method body.
  - Instead of curly braces an abstract method will have a semi colon (;) at the end.

Example

```
public abstract class Employee
{
   private String name;
   private String address;
   private int number;

   public abstract double computePay();
   //Remainder of class definition
}
```

```
/* File name : Salary.java */
public class Salary extends Employee
{
   private double salary; // Annual salary

   public double computePay()
   {
      System.out.println("Computing salary pay for " + getName());
      return salary/52;
   }

   //Remainder of class definition
}
```