Description

This week we will learn how to modify a class declaration using the keyword abstract. We will see how to compile and execute an abstract class, as long as we don't try to make an instance of it.

Abstract Classes

In the inheritance hierarchy, classes become more specific and concrete with each new subclass. If you move from a subclass back up to a superclass, the classes become more general and less specific. A superclass contains common features of its subclasses. Sometimes a superclass is so abstract that it cannot have any specific instances.

- Classes whose instances are meaningless.
- Classes which have one or more methods we do not know how to implement.

Such a class is referred to as an abstract class. Abstract classes are like regular classes, but we cannot create instances of abstract classes using the new operator. If a programmer tries to use the constructor to create an instance of the abstract class, it will not compile. An abstract method is defined without implementation. Its implementation is provided by the subclasses. A class that contains abstract methods must be defined abstract.

Employee is defined as a superclass for **FullTimeEmployee** and **PartTimeEmployee.** Employee models common features of Employee objects. Both FullTimeEmployee and PartTimeEmployee contain calculateSalary() method for computing the salary of full time and part time employees. Since we can compute salary for all employee objects. It is better to define calculateSalary() method in the Employee class. However, this method cannot be implemented in the Employee class, because its implementation depends on the specific type of Employee object. Such methods are referred to as abstract methods and are denoted using the abstract modifier in the method header.

```
public abstract class Employee{
public abstract double calculateSalary();
}
```

After we define abstract methods in Employee class, it becomes an abstract class.

- A class which will never be instantiated can be declared abstract.
- Methods which you do not know how to write can be declared abstract and do not have code for the body.
- Any class with an abstract method must be declared abstract.
- Subclasses must provide methods for abstract methods or they too will become abstract classes

Abstract class constructor

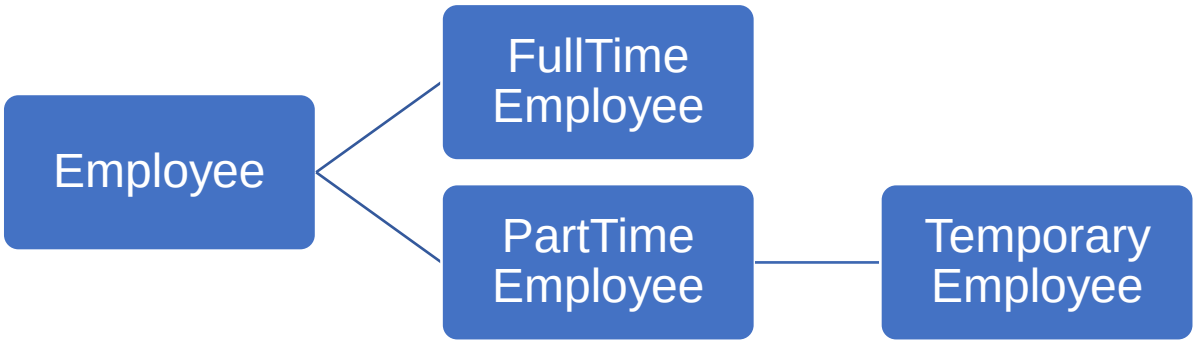
The constructor in the abstract class is defined **protected** because it is used only by subclasses. When you create an instance of a concrete subclass, its superclass's constructor is invoked to initialize data fields defined in the superclass.

```
public class FullTimeEmployee extends Employee{
    public double calculateSalary(){
        return salary / 52;
    }
}
```

```
public class PartTimeEmployee extends Employee{
    public double calculateSalary(){
        return hoursWorked * hourlyRate;
    }
}
```

There is one more category of part time employees, called a **temporary employee**. Part time employees are entitled to partial benefits but temporary employees are not. Temporary employees are like seasonal employees who are hired for a specific time, but there are chances that employment will continue beyond that time.

They have a start and end date of hiring. I have declared two LocalDate references in TemporaryEmployee class who is a child of PartTimeEmployee.



Here is the summary:

- An abstract method cannot be contained in a nonabstract class.
- The abstract methods are nonstatic.
- If a subclass of an abstract superclass does not implement all the methods, the subclass must be defined abstract.
- An abstract class cannot be instantiated using the new operator, but we can still define its constructors, which are invoked in the constructors of its

subclasses. For example, the constructor of Employee class are invoked in the FullTimeEmployee class and PartTimeEmployee class.

- A class that contains abstract methods must be abstract. However, we can define an abstract class that contains no abstract methods.
- A subclass can be abstract even if its superclass is concrete. For example, the Object class is concrete, but its subclasses such as Employee is abstract.
- A subclass can override a method from its superclass to define it abstract. However, it is very unusual.
- You cannot create an instance from an abstract class using the new operator, but an abstract class can be used as a data type. It means you can create an array whose elements are of the Employee type:
  Employee[] empList = new Employee[2];
- You can create an instance of FullTimeEmployee and assign its reference to the array as:
  empList[0] = new FullTimeEmployee();