Singleton Design Pattern
1. Make the ctor as private
2. provide a static getInstancce method that will return a new instance of the class
3. cretae a static reference of the same class as a field inside it.
4. check for if reference ois null then only create the object of the class and return the reference from the getInstance Method

Hirerachy
1. Has-a -> Association
    - Composition

    - Aggegration

2. Is-a -> Inheritance

```
class Apple{
Apple(){
super("Apple");
}
}

class Mango{

}
```

```
class Fruit{
String name;
Fruit(String name){
this.name = name;
}
}
```

Method Overriding

```
class Subclass {
accept(){
super class
}

@override
accept(){
super.accept();
sub class
}
}
```

Upcasting
-

```
Person p
accept(){
person::accept
}

display(){
person::display
}
```

```
Person p = new Person();
p.accept(); // person
p.display();//person
```

```
Employee e
accept(){
person::accept
}

display(){
person::display
}

accept{
super.accept();
Employee::accept
}

display{
Employee::display
}

calculateTax(){
}
```

```
Employee e = new Employee();
e. accept(); // employee
e.display(); // employee
```

```
Person p = new Employee(); //upcasting
p.accept(); // employee -> Late Binding
p.display();//employee -> Late Binding
```

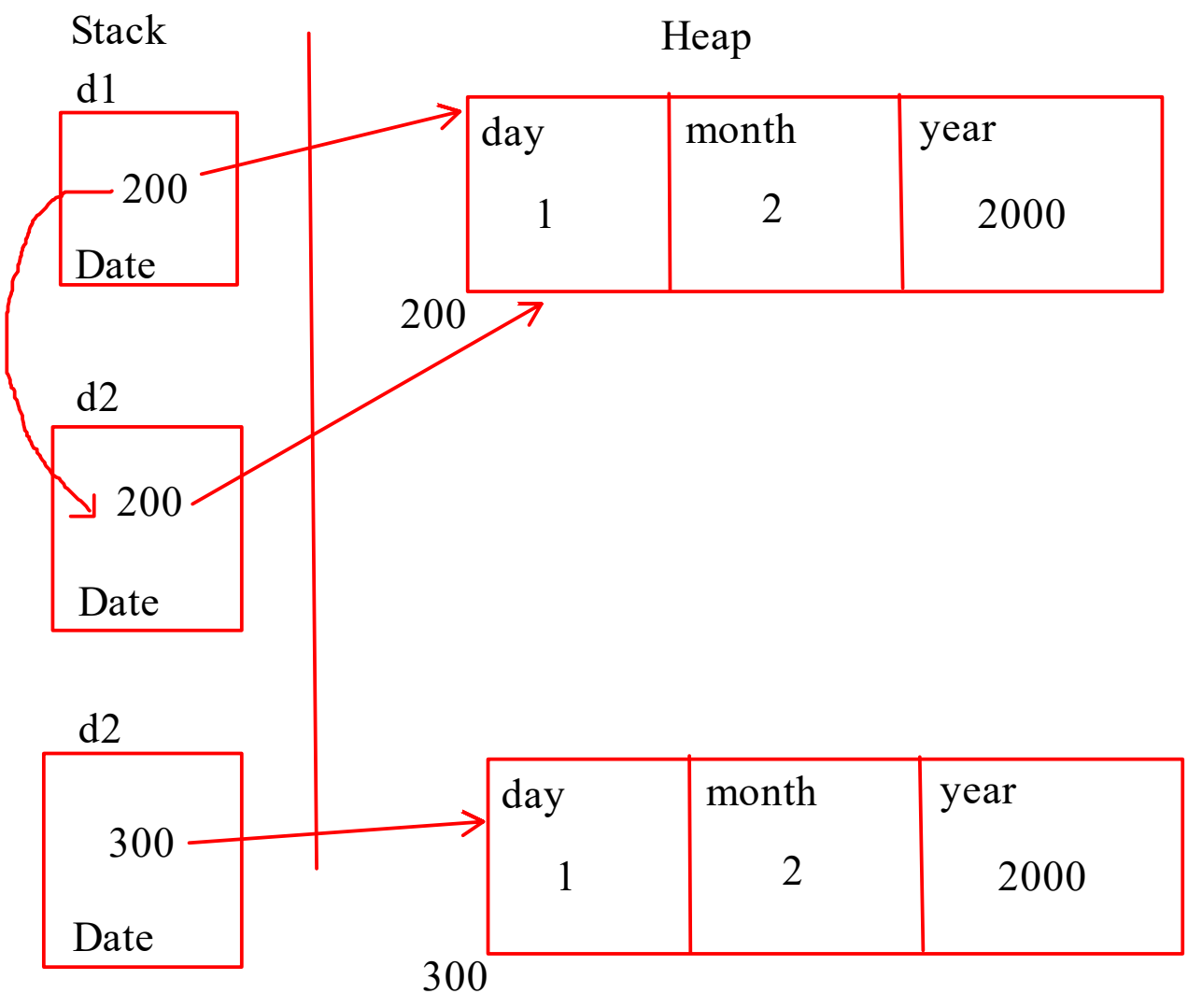
```
Employee e = (Employee)p; // Downcasting
e.calculateTax();
```

Upcasting
- Keeping object of sub class into super class reference is called as upcasting
- During upcasting the super class reference can point only at the methods of the super class inherited into the sub class or the methods overriden by the sub class.
- It cannot point to the methods of the sub class.
- This is called as object slicing
- To call the methods of subclass we have to do the downcasting

Downcasting
- Converting the referennce of super class into sub class reference is called as downcasting
- At the time of downcating if the downcast fails jvm throw an exception ClassCastException.
- To avoid this exception check whether the reference of super class is storing object of sub class.
- For this use instanceof operator.

Object
- It is super class of all the classes in java
- Object class has 11 methods
    - toString()
    - equals()

toString()
- It is method of object class that represent state of an object in string format
- The object class toString represents the state in the format of FullyQualifiedClassName@Hashcode
- to represent the state of an object in human redable format it is recommended that all the subclass should override toString();



Date d2 = d1;

d1==d2
200==200 -> true

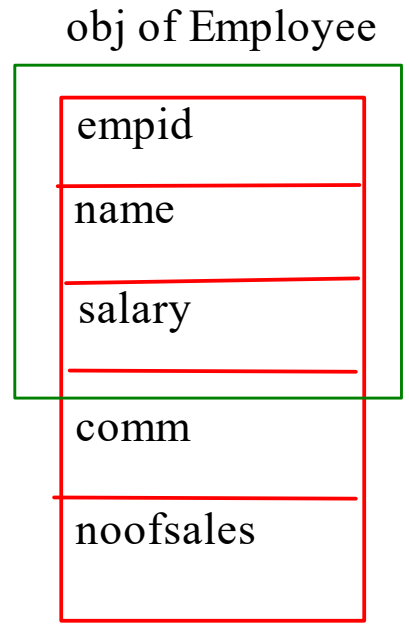
d1==d2
200 == 300 -> false

Employee
id,name,salary

Manager
bonus

Salesman
commission
noofproductssold

Abstract
- Abstract class is a class that contains abstract as well as non abstract methods.
- We cannot define abstact methods inside non abstract class.
- To declare abstract methods we have to make the class as abstract.
- We cannot create object of an abstract class, only references are allowed.
- All the sub classses that are derived from an abstract class have to compulsary override the abstract methods
- Abstract classes are used to group related types together.
- Abstact classes helps to keep the method design same across such related classes.
- Abstract classes can contain fields as well as constructor.



Fragile Base class Problem
- Any changes don in the super class can have corresponding effect on the sub classes.
- The sub class might need to implement new methods or need to change the method design and requires recompilation
- To overcome this problem use interface

Interface (Java 7 Interface)
- Interface is a set of protocols/specifications provided for your classes
- Interfaces are immutable i.e once declared should not be changed
- Interfaces are used to group related as well as unrelated types together
- Interfaces are used to keep the method design same across the types

```
shape{
virtrual void accept()=0;
virtual void calculatearea() = 0;
}
```