Programming Concepts Using Java

Week 4 Revision

Abstract classes

- Sometimes we collect together classes under a common heading
- Classes Swiggy, Zomato and UberEat are all food order
- Create a class FoodOrder so that Swiggy, Zomato and UberEat extend FoodOrder
- We want to force every FoodOrder class to define a function

  `public void order() {}`
- Now we should force every class to define the public void order();
- Provide an abstract definition in FoodOrder
- public abstract void order();

Interfaces

- An interface is a purely abstract class
- All methods are abstract by default
- All data members are final by default
- If any class implement an interface, it should provide concrete code for each abstract method
- Classes can implement multiple interfaces
- Java interfaces extended to allow static and default methods from JDK 1.8 onwards
- If two interfaces has same default/static methods then its implemented class must provide a fresh implementation
- If any class wants to extend another class and an interface then it should inherit the class and implements interface

private classes

- An instance variable can be a user defined type

```
public class BookMyshow{
     String user;
     int tickets;
     Payment payement;
}
public class Payment{
    int cardno;
    int cvv;
}
```

- Payment is a public class, also available to other classes
- Payment class has sensitive information, so there is a security concern.

private classes

- We cannot declare Payment class as private outside the BookMyshow class
- You can declare Payment class as private inside the BookMyshow class

```
public class BookMyshow{
     String user;
     int tickets;
     Payment payement;
     private class Payment{
        int cardno;
        int cvv;
     }
}
```

- Now Payment class is a private member of the BookMyshow class
- Now Payment class only available to the BookMyshow class

Interaction with State(Manipulating objects)

- Consider the class student below.
- Student class is encapsulated by private variables.

```
public class Student{
     private String rollno;
     private String name;
     private int age;
     //3 mutator methods
     //3 Accessor methods
}
```

- Consider Student class has student1,student2.....student60 objects
- Update date as a whole, rather than individual components

Interaction with State(Manipulating objects)

```
public class Student{
     private String rollno;
     private String name;
     private int age;
     public void setStudent(String rollno,String name,int age){
     }
}
```

- Now public void setStudent(String rollno, String name, int age) update the Student object as a whole.

Java Call back methods.

- what is call back method?

```
interface Notification{
void notification();//should be overridden in WorkingDay and Weekend
}
class WorkingDay implements Notification{
}
class Weekend implements Notification{
}
class Timer{//Timer will decide which call back function should be call
}
public class User {
    public static void main(String[] args) {
        Timer timer=new Timer();
        timer.start(new Date());
    }
}
```

Iterators

- what is Iterator?
- You can loop through any data structure using an Iterator.

```
public interface Iterator{
public abstract boolean has_next();
public abstract Object get_next();
}
```