Abstract:

The abstract keyword is used to define an abstract class. An abstract class cannot be instantiated directly.

```
abstract class Animal{
  void makeSound();
}
class Dog extends Animal{
  void makeSound(){
    print("bark");
  }
}
class Cat extends Animal{
  void makeSound(){
    print("meow");
  }
}
void main(){
  Dog d1=Dog();
  d1.makeSound();
  Cat c1=Cat();
  c1.makeSound();
}
```

Interface:

Dart does not have a special interface keyword. Any class can act as an interface.

Other classes implement the interface using the implements keyword. While implementing, a class must override all methods and properties declared in the interface.

```
abstract class Animal{
  void makeSound();
}

class Dog extends Animal{
  void makeSound(){
    print("bark");
  }
}

class Cat implements Animal{
  void makeSound(){
    print("meow");
  }
}
void main(){
  Dog d1=Dog();
  d1.makeSound();
  Cat c1=Cat();
  c1.makeSound();
```

Factory:

```
class banking {
```

```
  String? accnumber;
  String? accholder;
  double? balance;
banking._({this.accnumber,this.accholder,this.balance});
factory banking.toCheck({String? accnumber,String? accholder,double? balance})
 {
  if(balance!<100)
  {
   throw new Exception("Balance is insufficient");
  }
  else{
     return banking._(accnumber:  accnumber, accholder:accholder, balance: balance);
  }
 }
 double deposit(double amt)
 {
  return balance=balance!+amt;
 }

 double withDraw(double amt)
 {
  if(balance!>amt)
  {
   balance=balance!-amt;
  }
  else
  {
   throw new Exception("Less balance !");
```

```
    }
    return balance!;
  }
}
void main()
{
   var b1=banking.toCheck(accnumber:"b1",accholder:  "Tarunika",balance:125000);
   print(b1.deposit(25000));
   print("Balance is ${b1.withDraw(10000)}");
}
```