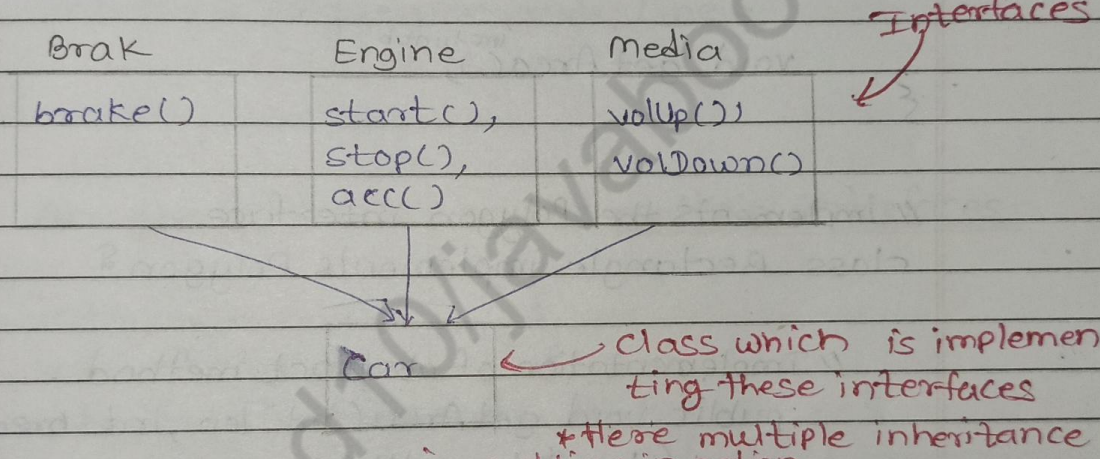
Interfaces

- An interfaces is a fully abstract class which contains a group of abstract methods (methods without a body.

- Interfaces can work as abstract class but abstract class can't work as interface, bec in interfaces multiple inheritance is allowed while in classes it is not allowed.



- Two classes are unrelated to each other can implement same interface.

- All the methods inside an interface are implicitly public and all fields are implicitly public static final.

```
interface Language {
    // by def public static final
    String type = "Programming lang.";
    // by def Public.
    void getName();
}
```

* Interface helps us to achieve abstraction like ~~for~~ abstract classes.

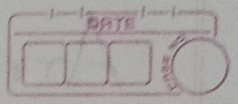
- Interfaces provides specifications that a class (which implements it) must follow.

Interface example

```
interface Polygon{

    void getArea(int l, int b);
}

// implements the Polygon interface
class Rectangle implements Polygon{

    // Implementation of abstract method.
    public void getArea(int len, int breadth){
    sout(len * breadth);
    }
}

class Main{
    psvm(String[] args){
        Rectangle rn = new Rectangle();
        rn.getArea(2,3);
    }}
```

O/P 6

* Annotations are interfaces.



* Extending an interface.
- The extends keyword is used to extend interfaces.

Ex. interface line {
    // members of line interface.
}

// extending interface.
interface Polygon extends Line {
    . . . .
}

- An interface can extend mutliple interfaces.

interface A { . . . . }
interface B { . . . . }
interface C extends A, B { . . . . }

* Default methods in interfaces.
- we can add methods with an implementation in interfaces these methods are called as default methods.

public default void getSides() {
    // body of getsides.
}

- If we not implement them in child class that is fine bec they have body we can directly reffer them.