1. تعدد الاشكال

مفهوم Polymorphism او تعدد الأشكال، ويحدث عندما يكون لدينا العديد من الكلاسات التي ترتبط ببعضها البعض عن طريق الوراثة.

كما حددنا في الفصل السابق؛ تتيح لنا الوراثة وراثة Attributes وmethods من الكلاس الاساسي الى الكلاس الفرعي. يستخدم Polymorphism هذه methods لأداء مهام مختلفة. هذا يسمح لنا بأداء عمل واحد بطرق مختلفة.

على سبيل المثال، فكر في كلاس اساسي اسمه Animal و يحتوي method تسمىanimalSound(). يمكن أن تكون الكلاسات الفرعية من كلاس Animal عبارة عن كلاس Cat و كلاس Dog ولديهم أيضًا method خاصة بنفس الاسم تسمى animalSound():

```
class Animal{
    public void  animalSound(){
        System.out.println("The animal makes a sound");
    }
}

class Dog extends Animal{
    @Override
    public void animalSound() {
        System.out.println("The dog says: bow wow");
    }
}

class Cat extends Animal{
    @Override
    public void animalSound() {
        System.out.println("The cat says: meow meow");
    }
}
```

يتم استخدام Override@ للإشارة ان هذه method موجودة مسبقاً في الكلاس الأساسي، ولكن تم تغييرها بداخل الكلاس الفرعي.

تكمن قوة مفهوم Polymorphism عند تعريف object لهذه الأنواع كما في المثال التالي:

```
public class Main {
    public static void main(String[] args) {
        Animal animal = new Animal();
        Animal cat = new Cat();
        Animal dog = new Dog();
        animal.animalSound();
        cat.animalSound();
        dog.animalSound();
    }
}
```

قمنا بتعريف 3 متغيرات من نوع Animal ولكن القيم لبمعطى لها تختلف:

- المتغير الأول تم اسناد له قيمة من نوع Animal
- المتغير الثاني تم اسناد له قيمة من نوع Cat
- المتغير الثالث تم اسناد قيمة له من نوع Dog

وهنا تم استخدام مفهوم Polymorphism بحيث انه نوع Animal يقبل اشكال متعددة للقيم و لكن يجب ان تكون كل القيم عبارة عن كلاسات فرعية لكلاس نوع المتغير.

عند القيام بمناداة دالة animalSound() سوف نلاحظ اختلاف القيمة المطبوعة و السبب يرجع الى نوع القيمة المسندة لكل متغير.

```
The animal makes a sound

The dog says: meow meow

The dog says: bow wow
```