1. نماذج البرمجة Programming Paradigms

1.1 Object-oriented programming

في البرمجة الكائنية، يتم تمثيل المعلومات كفئة تصف مفاهيم مجال المشكلة ومنطق التطبيق. تحدد الفئات الطرق التي تحدد كيفية معالجة المعلومات.

1.2 Procedural programming

بينما في البرمجة الشيئية، يتم تشكيل هيكل البرنامج من خلال البيانات التي يعالجها ، في البرمجة الإجرائية ، يتم تشكيل هيكل البرنامج من خلال الوظيفة المطلوبة للبرنامج: يعمل البرنامج كدليل خطوة بخطوة للوظيفة المطلوب أداؤها .

1.3 Static keyword

الفرق بين Non-Static vs. Static في الدروس السابقة قمنا بتعريف دوال باستخدام static كما في المثال التالي:

```
static void printHelloWorld(){
    System.out.println("Hello World");
}
```

الفرق بين Static vs. Non-Static هو انه في الدوال من نوع static يمكننا استخدامها بدون انشاء object لهذا الكلاس بينما دوال Non-Static يجب علينا انشاء object لهذا الكلاس ثم استخدام هذه الدالة كما في المثال التالي:

```
class Animal{
    private String name;
    private int age;


    Animal(String name, int age) {
        this.name = name;
        this.age = age;
    }

    void hunt(){
        System.out.println("We Are Hunting Now ");
    }
    static void staticHunt(){
        System.out.println("StaticHunt: We Are Hunting Now ");

    }
}
    Animal cat = new Animal("Cat",4);
    cat.hunt();
    Animal.staticHunt();
```

Java Access Modifiers 1.4

قبل البدء بشرح مفهوم Modifiers لاحظنا سابقا تكرار كلمة public كثيرا وهي احدى خيارات Modifiers في لغة Java، تنقسم Modifiers الى ثلاثة يمكن تعريفها في الجدول التالي:

نوع Modifiers	التعريف
public	يمكن الوصول الى العناصر الخاصة بالكلاس من اي مكان
private	لا يمكن الوصول الى العناصر الخاصة بالكلاس الا من داخل الكلاس
protected	يمكن الوصول الى العناصر الخاصة بالكلاس من الكلاسات الموجوة بنفس المجلد والكلاسات التي ترث منه
default	يمكن الوصول الى العناصر الخاصة بالكلاس من الكلاسات الموجوة بنفس المجلد

لشرح الفكرة بشكل مبسط سوف نقوم بالتجربة على object من نوع Animal في الوقت الحالي جميع العناصر الخاصة بهذا class لم يتم تعريف Modifiers خاصة بها اي انها الان من نوع default ويتم الوصول لها من الكلاسات الموجودة بنفس المجلد.

```
class Animal{
    String name;
    int age;

    Animal(String name, int age) {
        this.name = name;
        this.age = age;
    }
}

public class Main {
    public static void main(String[] args) {
    Animal cat = new Animal("Cat",4);
    System.out.println(cat.name);
}
}
```

في المثال الأعلى يمكننا طباعة خاصية name الخاصة ب cat لكن لو قمنا بتغيير نوع Modifiers الخاصة بعناصر هذا الكلاس فسوف يصبح من غير الممكن طباعتها.

```
class Animal{
    private String name;
    private int age;
```

```
    Animal(String name, int age) {
        this.name = name;
        this.age = age;
    }
}


public class Main {
    public static void main(String[] args) {
    Animal cat = new Animal("Cat",4);
    System.out.println(cat.name);

}
}
```

وسوف نحصل على خطأ.

```
java: name has private access in Animal
```