JAVA OOP

INHERITANCE

2 INTRODUCTION

- Inheritance គឺជា key feature មួយរបស់ OOP ដែលអនុញ្ញាតិអោយយើងបង្កើត Class ថ្មី ចេញពី Class ដែលមានស្រាប់។
- Class ថ្មីនោះហៅថា subclass (Child ឬ Derived class) ហើយ Class ដែលមានស្រាប់នោះហៅថា superclass (parent ឬ base class)
- Keyword extends ត្រូវបានប្រើដើម្បីអនុវត្ត inheritance នៅក្នុងភាសា Java

3 INTRODUCTION

```
class Animal {
  // methods និង fields
}
class Dog extends Animal {
  // methods និង fields របស់ Animal រួមជាមួយ methods និង fields របស់ Dog
}
```

Dog គឺជា subclass ហើយ Animal គឺជា superclassฯ

4 INTRODUCTION

```
class Animal {
  // field និង method របស់ parent class
  String name;
  public void eat() {
    System.out.println("I can eat");
  }
}
```

```
class Dog extends Animal {
  // method ថ្មីនៅក្នុង subclass
  public void display() {
    System.out.println("My name is " + name);
  }
}
```

5 INTRODUCTION

```
class Main {
  public static void main(String[] args) {
    Dog dog= new Dog(); // បង្កើត object របស់ subclass
    dog.name = "Kitty"; // access field របស់ superclass
    dog.display();
    dog.eat(); // ហៅ method របស់ superclass ដោយប្រើ object របស់ subclass
  }
}
```

OUTPUT:

```
My name is Kitty
I can eat
```

6 IS-A RELATIONSHIP

Inheritance នៅក្នុង Java គឺជា is-a relationship។ ដូច្នេះយើងប្រើ inheritance បានតែនៅពេលណាដែលមាន is-a relationship រវាង class ២។ ឧទាហរណ៍៖

- Car is a Vehicle
- Orange is a Fruit
- Surgeon is a Doctor
- Dog is an Animal

ដូច្នេះ Car អាច inherit ពី Vehicle, Orange អាច inherit ពី Fruit ។ល។

7 METHOD OVERRIDING

យើងឃើញថា object របស់ subclass អាច access method របស់ superclass។

ចុះបើ method នោះមានវត្តមានទាំងក្នុង superclass និង subclass តើនឹងមានអ្វីកើតឡើង?

ក្នុងករណីនេះ method នៅក្នុង subclass overrides method របស់ superclass។ Concept គឺជា method overriding របស់ Java។

8 METHOD OVERRIDING

```
class Animal {
  // method របស់ superclass
  public void eat() {
    System.out.println("I can eat");
  }
}
```

```
class Dog extends Animal {
    @Override
  public void eat() { // overriding the eat() method
    System.out.println("I eat dog food");
  }
  public void bark() { // method ថ្មីរបស់ subclass
    System.out.println("I can bark");
  }
}
```

9 METHOD OVERRIDING

```
class Main {
  public static void main(String[] args) {
    Dog myDog = new Dog(); // បង្កើត object របស់ subclass
    myDog.eat(); // ហៅ method eat()
    myDog.bark();
  }
}
```

OUTPUT:

I eat dog food

I can bark

10 SUPER KEYWORD

មុននេះ យើងបានឃើញថា method នៅក្នុង subclass overrides method របស់ superclass។

ក្នុងស្ថានភាពបែបនេះ keyword super ត្រូវបានប្រើដើម្បីហៅ method របស់ class មេ ពីទីតាំង method របស់ classកូន។

SUPER KEYWORD

```
class Animal {
  // method របស់ superclass
  public void eat() {
    System.out.println("I can eat");
  }
}
```

```
class Dog extends Animal {
    @Override
  public void eat() { // overriding the eat() method
     super.eat(); // ហៅ method របស់ superclass
    System.out.println("I eat Dog food");
  }
public void bark() { // method ថ្មីរបស់ subclass
    System.out.println("I can bark");
  }
```

12 SUPER KEYWORD

```
class Main {
  public static void main(String[] args) {
    Dog myDog = new Dog(); // បង្កើត object របស់ subclass
    myDog.eat(); // ហៅ method  eat()
    myDog.bark();
  }
}
```

OUTPUT:

I can eat

I eat Dog food

I can bark

13 PROTECTED MEMBERS IN INHERITANCE

ក្នុង Java, ប្រសិនបើ class មួយដាក់ fields និង methods ជា protected នោះ fields និង methods ទាំងនោះអាច access ពី subclass របស់ class នោះបាន។

14 PROTECTED MEMBERS IN INHERITANCE

```
class Animal {
  protected String name;
  protected void display() {
    System.out.println("I am an animal.");
  }
}
```

```
class Dog extends Animal {
  public void getInfo() {
    System.out.println("My name is " + name);
  }
}
```

15 PROTECTED MEMBERS IN INHERITANCE

```
class Main {
  public static void main(String[] args) {
    Dog myDog = new Dog(); // បង្កើត object របស់ subclass
// access protected field និង method ដោយប្រើ object របស់ subclass
    myDog.name = "Davan";
    myDog.display();
    myDog.getInfo();
  }
}
```

OUTPUT:

I am an animal.

My name is Davan

16 ប្រើ INHERITANCE ដើម្បីអ្វី?

- សារៈសំខាន់ជាងគេនៃការប្រើ inheritance ក្នុង Java គឺ code reusability។ Code ដែលមានវត្តមានក្នុង class មេ អាចត្រូវបានយកមកប្រើដោយផ្ទាល់ក្នុង class កូន។
- Method overriding ត្រូវបានគេស្គាល់ថាជា runtime polymorphism។ ដូច្នេះ យើងអាចទទួលបាន Polymorphism ក្នុង Java តាមរយៈ inheritance។