ជំពូកទី ៦ : អំពី Inheritance

១. សញ្ញាណ Inheritance

Java ផ្តល់នូវលក្ខណៈ inheritance ដែលអាចអោយ class មួយទទួលលក្ខណៈពី class មួយទៀតបាន ។ នេះធ្វើឡើងដោយប្រើពាក្យ extends ។ class ដែលទទួលលក្ខណៈពីគេហៅថា subclass ។ ហើយ class ដែលអោយគេទទួលលក្ខណៈពីវាហៅថា superclass ។

ទំរង់ទូទៅនៃការប្រកាស class មួយដែលបានទទួល លក្ខណៈពី superclass មួយមានដូចខាងក្រោមនេះ :

```
class subclass-name  extends superclass-name {
    // body of class
}
```

សារៈសំខាន់នៃ inheritance គឺ កាលណាយើងបានបង្កើត superclass មួយដែលបានកំនត់លក្ខណៈរួមទៅអោយ objects ហើយពេលនោះវាអាចប្រើសំរាប់បង្កើត subclass ជាច្រើនដែលមានលក្ខណៈកាន់តែលំអិតជាង ។

ឧទាហរណ៏ :

```
class TwoDShape {
  double width, height;
  void showDim(){
     System.out.println("Width and"
         + " height are " + width +
         " and " + height);  }
  }
  class Triangle extends TwoDShape{
    String style;
    double area(){
      return width * height/2;
    }
    void showStyle(){
      System.out.println("Triangle"
            + " is " + style);   }
  }
```


២. Constructors និង Inheritance

នៅក្នុងលំដាប់ថ្នាក់ គេអាចអោយ superclass និង subclass មាន constructor ដោយខ្លួន ។ Constructors សំរាប់ superclass បង្កើតនូវ object នៃផ្នែក superclass ហើយ constructor សំរាប់ subclass បង្កើតនូវ object នៃផ្នែក subclass ។

ឧទាហរណ៏ :

២.១ ការប្រើ super ដើម្បីហៅ constructor នៃ superclass មកប្រើក្នុង subclass

Subclass មួយអាចហៅ constructor មួយដែលកំណត់ដោយ superclass តាមទំរង់ super ខាងក្រោម :

super (*parameter-list*);

ក្នុងនេះ *parameter-list* បញ្ជាក់នូវប៉ារ៉ាម៉ែត្រណាមួយដែល constructor ត្រូវការនៅក្នុង superclass ។ super() ត្រូវតែស្ថិតនៅឃ្លាទីមួយជានិច្ចដើម្បីប្រតិបត្តិការ នៅក្នុង constructor នៃ subclass ។

ឧទាហរណ៍ :

២.២ ការប្រើ super ដើម្បីចូលទៅប្រើ members របស់ superclass

ទំរង់ទីពីរនៃ super មានលក្ខណៈដូច this លើកលែងតែវាជានិច្ចជាកាលត្រូវបានគេប្រើនៅក្នុង subclass ជាមួយនឹង members របស់ superclass ។ ទំរង់ទូទៅរបស់វា គឺ :

super.*member*

ក្នុងនេះ member អាចជា method ឬ អញ្ជាត ។

ឧទាហរណ៏ :

៣. Superclass References និង Objects នៃ subclass

អញ្ជាតនៃ superclass ដែលមានលក្ខណៈ reference អាចកំណត់តំលៃទៅអោយអញ្ជាត reference នៃ subclass ទាំងឡាយណាដែលទទួលបានលក្ខណៈពី superclass នោះ។

ឧទាហរណ៏ :

៤. អំពី Method Overridings

កាលណា method មួយនៅក្នុង subclass មានឈ្មោះ និង ប្រភេទទិន្នន័យដូចគ្នានឹង method នៅក្នុង superclass របស់វា នោះ method នៅក្នុង subclass ត្រូវបានគេ និយាយថា លុបលើ method នៅក្នុង superclass ។ កាលណា method មួយដែលត្រូវលុបលើនោះ បានហៅប្រើ នៅក្នុង subclass ជានិច្ចជាកាលវាបញ្ជាក់ទៅអោយកំណែ ប្រែថ្មីនៃ method នោះ ។ រីង method ដែលកំនត់ដោយ superclass នឹងត្រូវបំពាំង ។ ឧទាហរណ៏ :

៥. Overridden Method ផ្តល់នូវលក្ខណៈ Polymorphism

ការលុបលើ method បង្កើតនូវមូលដ្ឋានសំខាន់ចំពោះទស្សនៈដ៏មានឥទ្ធិពលមួយរបស់ Java គឺ : dynamic method dispatch ។ វាជា mechanism មួយ ដែលការហៅប្រើចំពោះ method លុបលើ ត្រូវបានដោះស្រាយនៅក្នុងពេលដំណើរការជាជាងដោះស្រាយនៅក្នុងពេលបំលែង code ។ វាមានសារៈសំខាន់ ព្រោះនេះជារបៀបដែល Java អនុវត្តនូវលក្ខណៈ polymorphism ក្នុងពេលដំណើរការ ។

ឧទាហរណ៍ :

៦. ការប្រើ Abstract classes

Class មួយដែលមាន abstract method មួយឬច្រើន ត្រូវតែប្រកាសដោយប្រើពាក្យ abstract នៅពីមុខឈ្មោះ class របស់វា។ ដោយសារ abstract class ពុំមានការអនុវត្ត ពេញលេញនាំអោយវាគ្មាន object នៃ abstract class។ ហេតុនេះការព្យាយាមបង្កើត object នៃ abstract class មួយ ដោយប្រើ new នោះនឹងនាំអោយ error កើតឡើងនៅពេល យើងធ្វើការបំលែង code។ ដើម្បីប្រកាស abstract method មួយ យើងប្រើទំរង់ទូទៅដូចខាងក្រោមនេះ :

abstract *type name* (*parameter-list*);

ឧទាហរណ៏ :

```
abstract class Shape {
    double width, height, radius;
     double PI = 3.14159;
     double getWidth(){
       return width;
     }
     double getHeight(){
       return height;
     }
     double getRadius(){
       return radius;
     }
     abstract double area();
}
```

៧. ការប្រើ final

final អាចប្រើបានបីយ៉ាង គឺ :

- ការពារមិនអោយមាន overriding method ។

ឧទាហរណ៍ :

```
class A{
   final void meth(){
   System.out.println("This is a final method.");
 }
 }
 class B extends A{
   void meth(){ // Error! Can't override.
      System.out.println("Illegal!");
   }
 }
```

- ការពារមិនអោយមានលក្ខណៈ inheritance ។

ឧទាហរណ៏ :

```
final class A {
    // …
}
// The following class is illegal.
class B extends A{
     // Error! Can't subclass A
     // …
}
```

- ការពារមិនអោយមានការប្តូរតំលៃនៃអញ្ជាត(រក្សាតំលៃថេរ)

ឧទាហរណ៏ :

សំនួរ និង លំហាត់

១ - ចូរពន្យល់ពីការប្រើ super.member និង super() ។

២ - តើ subclass អាចមាន superclass ច្រើនឬទេ?

៣ - តើ method overriding ជាអ្វី?

៤ - ចូរសរសេរ code បង្ហាញពីការប្រើ abstract class និង abstract method ។

៥ - តើ final អាចប្រើបានប៉ុន្មានយ៉ាង? ចូរសរសេរ code បង្ហាញ ។