ជំពូកទី ៥ : អំពី Class និង Methods

១. មូលដ្ឋានគ្រឹះនៃ Class

Class គឺជាលំនាំមួយដែលកំនត់នូវទំរង់របស់ object មួយ ។

Objects គឺជា instances នៃ class មួយ ។

ទំរង់ទូទៅរបស់វា គឺ :

By Mr. Chi Kuong

```
class classname {
    type instance-variable1;
    type instance-variable2;
    // ..
    type methodname1(parameter-list) {
      // body of method
    }
    type methodname2(parameter-list) {
      // body of method
    }
      // …
}
```

Methods និង អញ្ញាត ដែលមាននៅក្នុងclass

មួយ ហៅថា members នៃ class ។

ឧទាហរណ៍ : Vehicle class បានកំនត់នូវ instance variables ចំនួនបី គឺ : passengers, fuelcap និង mpg ។

```
class Vehicle {
   int passengers; // number of passengers
   int fuelcap; // fuel capacity in gallons
   int mpg; // fuel consumption
            // in miles per gallon
}
```

ដើម្បីបង្កើត object ប្រភេទ Vehicle មួយ គេត្រូវសរសេរ :

```
Vehicle minivan = new Vehicle();
```

ឃ្លាខាងលើនេះ គេអាចសរសេរជា :

```
Vehicle minivan;
minivan = new Vehicle();
```

៣. អញ្ជើញ reference និងការកំណត់តំលៃ

(Reference Variables and Assignment)

ចូរពិនិត្យឃ្លាខាងក្រោមនេះ :

```
Vehicle car1 = new Vehicle();
Vehicle car2 = car1;
```

ក្រឡេកមើលដំបូង វាហាក់ដូចជាងាយគិតថា car1 និង car2 បញ្ជាក់អោយ objects ពីរផ្សេងគ្នា តែតាមពិតនោះវាមិនដូច្នោះទេ។ car1 និង car2 បញ្ជាក់អោយ object ដូចគ្នា។ ការកំនត់តំលៃនៃ car1 ទៅអោយ car2 ធ្វើអោយ car2 បញ្ជាក់ទៅអោយ object ដូចគ្នា ដូចនឹង car1 ដែរ។

ឧទាហរណ៏ : បន្ទាប់ពីការកំនត់តំលៃ : car1.mpg = 26;

ក្រោយពីប្រតិបត្តិការឃ្លា println() ទាំងពីរនេះ គឺ :

```
System.out.println(car1.mpg);
System.out.println(car2.mpg);
```

វានឹងបង្ហាញតំលៃដូចគ្នា គឺ : 26 ។

៤. អំពី Method

ទំរង់ទូទៅនៃ method មានដូចខាងក្រោមនេះ :

ret-type name(*parameter-list*) {
  // body of method
}

ឧទាហរណ៍ :

```
void range(){
   System.out.println("Range is "
                + fuelcap * mpg);
}
```

គេអាចបញ្ជូនតំលៃទៅក្នុង method បានតាមរយៈ ប៉ារ៉ាម៉ែត្ររបស់វា ។

ឧទាហរណ៍ :

```
double fuelneeded(int miles) {
    return (double) miles / mpg;
}
```

៥. អំពី Constructor

Constructor ប្រើសំរាប់កំនត់តំលៃដំបូងទៅអោយ object ពេល object បង្កើតឡើង ។ វាមានឈ្មោះដូចទៅនឹងឈ្មោះ class របស់វា ហើយពុំមានបញ្ជាក់ប្រភេទទិន្នន័យឡើយ ។

លើសពីនេះ វាក៏ពុំមានការប្រើពាក្យ void ដាក់ពីមុខដែរ ។

ឧទាហរណ៍ :

```
class MyClass{
        int x;
        MyClass(){
            x = 10;
        }
}
```

៦. ការប្រើពាក្យ this

កាលណា method មួយត្រូវបានហៅប្រើ នោះវាបញ្ជូន argument មួយដោយស្វ័យប្រវត្ត ដែលជាការបញ្ជាក់មួយទៅអោយ object បង្កើតឡើង ។

ឧទាហរណ៍ :

```
class Pwr {
    double b;
    int e;
    double val;
    Pwr(double base, int exp){
      this.b = base;
      this.e = exp;

      this.val = 1;
      if(exp == 0)  return;
      for( ; exp>0; exp--)
        this.val=this.val*base;
    }
    double getPower(){
      return this.val;
    }
}
```

៧. ការប្រមូលសំណល់ និង finalizers

Java មានលក្ខណៈបំផ្លាញចោលនូវ object ឬ អញ្ញាតណា ដែលឈប់ប្រើចេញពី memory ដោយស្វ័យប្រវត្ត ។

គេអាចប្រើ finalize() method ដើម្បីធ្វើសកម្មភាពមុននឹង object មួយត្រូវបំផ្លាញ ដោយធានាបាននូវ object មួយត្រូវបញ្ចប់ទាំងស្រុង ។ វាមានទំរង់ទូទៅដូចខាងក្រោម :

```
protected void finalize(){
   // finalization code here
}
```

៨. អំពី Method Overloadings

នៅក្នុងភាសា Java វាអាចមាន methods ពីរ ឬច្រើន ក្នុង class មួយដែលមានឈ្មោះដូចគ្នា តែមានការប្រកាសប៉ារ៉ាម៉ែត្រផ្សេងគ្នា ឬ មានចំនួនប៉ារ៉ាម៉ែត្រមិនស្មើគ្នា។

ឧទាហរណ៍ :

```
class A {
    int area(int w, int h){
        return w * h;
    }
    double area(double r){
        return 3.14 * r * r;
    }
}
```

៩. អំពី Overloading Constructors

ដូចនឹង method ដែរ នៅក្នុង class មួយអាចមាន constructors ច្រើនបាន។

ឧទាហរណ៍ :

```
class A {
        int x;
        A(){
           x = 0;
        }
        A(int i){
           x = i;
        }
}
```

១០. ការបញ្ជូនតំលៃជា object ទៅកាយ method

គេអាចបញ្ជូនតំលៃជា object ទៅក្នុង method បានដូចនឹងប៉ារ៉ាម៉ែត្រដែរ ។

ឧទាហរណ៍ :

វាមាន methods ពីរប្រភេទ គឺ :

- មានលក្ខណៈ call-by-value
- មានលក្ខណៈ call-by-reference

១១. ការអោយតំលៃជា objects ពី methods

Method មួយអាចអោយតំលៃជាប្រភេទទិន្នន័យណាមួយ

រួមទាំងប្រភេទទិន្នន័យជា class ផង ។

ឧទាហរណ៏ :

១២. អំពី recursion

Method មួយអាចហៅប្រើដោយខ្លួនវា ។ Method ដែល

មានលក្ខណៈបែបនេះហៅថា recursive method ។

ឧទាហរណ៍ទី ១ :

```
int factR(int n) {
   int result;
   if (n == 0 || n == 1)
      return 1;
   else result = factR(n-1) * n;
   return result;
}
```

ឧទាហរណ៍ទី ២ :

```
int fibo(int n) {
   if ((n==1)||(n==2))
      return 1;
   else
      return fibo(n-1)+fibo(n-2);
}
```

១៣. លក្ខណៈ static

Member នៃ class មួយ ដែលមានលក្ខណៈ static

- អាចប្រើបានជាមួយឈ្មោះ class ដោយមិនចាំបាច់មានការកកើត object ជាមុនឡើយ ។
- មានលក្ខណៈចំលងតំលៃ (share value)
- method ដែលមានលក្ខណៈ static អាចប្រើបានតែជាមួយនឹង data ដែលមានលក្ខណៈ static ប៉ុណ្ណោះ ។ រាងុំអាចប្រើជាមួយពាក្យ this ទេ ។

លើសពីនេះ គេប្រើ static block ដើម្បីកំនត់តំលៃដំបូងមួយចំនួន មុននឹង objects បង្កើតឡើង ។

១៤. Class មួយនៅក្នុង class មួយទៀត

នៅក្នុង Java គេអាចកំនត់អោយមាន class មួយនៅក្នុង class មួយទៀតបាន ។ លក្ខណៈនេះមានចំពោះ Java version 1.1 ឡើងទៅ ។ members នៃ class ក្រៅ អាចចូលប្រើដោយ class ក្នុង តែ members នៃ class ក្នុង ពុំអាចចូលប្រើដោយ class ក្រៅដោយផ្ទាល់បានឡើយ ។

ឧទាហរណ៍ :

```
class Outer {
  int outer_x = 100;

  void test() {
    for(int i=0; i<10; i++) {
      class Inner {
        void display() {
          System.out.print("display: "
            +" outer_x = "+ outer_x);
        }
      }
      Inner inner = new Inner();
      inner.display();
    }
  }
}
```

កាលណាគេបង្កើត class មួយនៅក្នុង class មួយទៀត បើយ class ក្នុងពុំបានបង្កើត object នៅក្នុង class ក្រៅទេនោះ គេអាចចូលទៅប្រើ members របស់ class ក្នុងតាមរយៈ object នៅកំរិតនៃ class ក្នុងនោះ ។

ឧទាហរណ៍ : កម្មវិធីខាងក្រោមបង្ហាញពីការប្រើលក្ខណៈបែបនេះ ។

សំនួរ និង លំហាត់

១ - តើ constructor ជាអ្វី? តើវាប្រតិបត្តិនៅពេលណា ?

២ - ចូរពន្យល់ពីភាពខុសគ្នារវាង class និង object ។

៣ - ចូរនិយាយពីលក្ខណៈ static ដែលប្រើជាមួយនឹង members

របស់ class ព្រមទាំង static block ។

៤ - ចូរសរសេរ recursive method ដែលអាចគណនាផលបូក 2

+ 4 + ... + 2N ដោយអោយតំលៃជាចំនួនគត់ ។

៥ - ចូរបង្កើត class មួយឈ្មោះ Rectangle ។ Class នេះ មានលក្ខណៈពីរគឺ បណ្តោយ (width) និង ទទឹង (height) ហើយតំលៃ default របស់វា គឺស្មើនឹង 1 ។ វាមាន method ដែលអាចគណនាបរិមាត្រ និងក្រឡាផ្ទៃនៃចតុកោណកែង ។ ហើយវាក៏មាន methods ឈ្មោះ set និង get សំរាប់ កំណត់តំលៃ និងទទួលតំលៃរបស់ width និង height ដែរ ។ methods ទាំងពីរនេះមានលក្ខខ័ណ្ឌផ្ទៀងផ្ទាត់ជាលេខ ទសភាគ ដោយធំជាង 0.0 និង តូចជាង 20.0 ។

៦ - ចូរសរសេរកម្មវិធីមួយដើម្បីបង្ហាញពីការចូលប្រើ member នៃ inner classes ទាំងអស់ដែលមានទំរង់ដូចខាងក្រោម :

```
class AA {
    int x = 10;
    class BB {
        int y = 20;
        int xy( ){  return x+y;   }
        class CC {
            int z = 30;
            int xyz( ){  return x+y+z;   }
        }
    }
}
```