Korea Software HRD Center

Software Expert Program Training

Homework 001

**ឈ្មោះ** គឹម ពន្លឺ  
**ថ្នាក់** កំពង់សោម

**មុខវិជ្ជា** JAVA

1. **What is OOP?**

OOP គឺជាវិធីសាស្រ្តក្នុងការបង្កើតកម្មវិធីតាមបែបជាលក្ខណៈទម្រង់ Class និង Object។ ដូច្នេះអ្វីៗទាំងអស់ត្រូវបានកំនត់ជា Class ហើយនិង Object ដើម្បីធ្វើការងាយស្រួល រឺ ជួយសំរួលដល់ការសរសេរកម្មវិធីព្រោះអ្វីៗទាំងអស់រចនា ឫ ផ្ដោតទៅលើObject។

OOP មានទម្រង់និងវិធីសាស្រ្តដូចជា៖

* Object
* Class
* Inheritance
* Polymorphism
* Abstraction
* Encapsulation

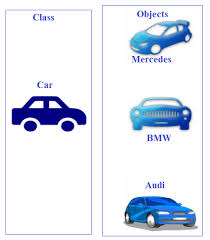
1. **What is Class? What is Object?**

• Object: អ្វីៗទាំងអស់គឺជាObject ដែលវាមាន លក្ខណៈ (states) និង សកម្មភាព រឺ ចលនា (behavior) ដែលមានតម្លៃជាក់លាក់និងច្បាស់លាស់។ ហើយនៅក្នុង Programming លក្ខណៈ States សំដៅទៅលើ fields របស់ object និង behavior សំដៅទៅលើ methods ដែលវាស្ថិតនៅក្នុងពុម្ភគំរូ Class.

**ឧទាហរណ៍ៈ** ឡានមួយមាន Statesដូចជាឈ្មោះ ម៉ាក ពណ៍ កញ្ចក់ និង​មាន behavior ដូចជា ទៅមុខ ថយក្រោយ​ ជាដើម។

• Class: គឺជាពុម្ពគំរូមួយ template (Blueprint) ដែលជួយបង្កើត និងជួយកំនត់លក្ខណៈអោយ objectរបស់ Class នោះ។

**ឧទាហរណ៍ៈ** Class and Object

[](http://javacodelab.com/category/java-interview-questions/)

1. **What is Inheritance?**

គឺជាការអនុញ្ញាតឪ្យ objectមួយអាចទទួលយកទិន្នន័យពីobjectមួយផ្សេងទៀតមកប្រើប្រាស់បាន។ ដូច្នេះយើងអាចកំណត់វាបានថាជា mechanism ដែលជួយឪ្យ objectមួយស្គាល់ពីលក្ខណៈនិង behaviors របស់objectមួយផ្សេងទៀតបានបន្ទាប់ពីធ្វើការ inherit/extend។នៅក្នុងការប្រើប្រាស់inheritance យើងអាចបង្កើត class ថ្មីដោយពឹងផ្អែកទៅលើclass ដែលមានរួច ហើយបង្កើតថ្មីឬប្រើប្រាស់ឡើងវិញនៅ methods និង fields របស់ class មេនោះបាន។

**class** Faculty

{

**float** salary=30000;

}

**class** Science **extends** Faculty

{

**float** bonous=2000;

**public** **static** **void** main(String args[])

{

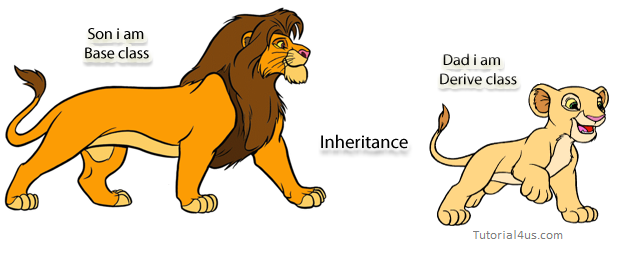
Science obj=**new** Science();

System.**out**.println("Salary is:"+obj.salary);

System.**out**.println("Bonous is:"+obj.bonous);

}

}



**Real World Example**

**Code Example**

1. **What is Encapsulation?**

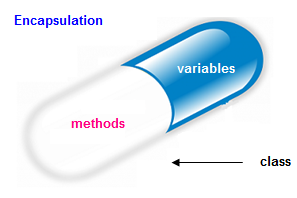
គឺជាមធ្យោបាយក្នុងការដាក់រួមបញ្ចូលគ្នារវាងទិន្នន័យ ឫ វេចខ្ចប់data (wrapping data) ទៅក្នុងធាតុតែមួយ(ការបង្កើត Class)។ នៅក្នុង **Encapsulation** ទិន្នន័យ(variable)និងមុខងារ(method) ត្រូវបានវេចខ្ចប់បញ្ចូលទៅក្នុងclass ហើយក្នុងការប្រើមធ្យោបាយ Encapsulation នេះ គ្រប់ទិន្នន័យទាំងអស់ត្រូវបានការពារពីការ Access ដោយផ្ទាល់ពី Class ដទៃខាងក្រៅ។ ទិន្នន័យទាំងនោះអាច Access បានតាមរយៈ Method តែប៉ុណ្នោះ។ វាត្រូវបានគែស្គាល់ម៉្យាងទៀតថា Data Hiding។

របៀបបង្កើត ​Encapsulation

ដើម្បីបង្កើតឲ្យមាន Encapsulation យើងត្រូវ៖

១.បង្កើត Data Member ជា private

២.បង្កើត Access Method ជា public



**class** Employee

{

**private** String name;

**public** String getName()

{

**return** name;

}

**public** **void** setName(String name){

**this**.name=name;

}

}

**class** Demo

{

**public** **static** **void** main(String[] args)

{

Employee e=**new** Employee();

e.setName("Harry");

System.**out**.println(e.getName());

}

}

**Code Example**

**Real World Example**

1. **What is Polymorphism?**

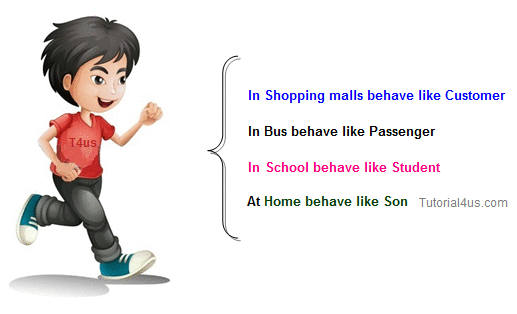
គឺជាសមត្ថភាពមួយដែលអាចឪ្យ method ធ្វើការផ្សេងៗគ្នាដោយទៅតាម object ដែលវាសម្តែងលើ ​ហើយទម្រង់បែបនេះបានកើតឡើងនៅក្នុងOOPនៅពេល objectនោះមានមុខងារ ឫ ការសំដែងខុសៗគ្នាទៅតាមmethods ដែលវាបានកំណត់។

Polymorphism មានពីរប្រភេទគឺ៖

* Overloaded Methods (Compile time Polymorphism)
* Overridden Methods(Run time Polymorphism)

**Code Example**

**Real World Example**



**class** Person

{

**void** walk()

{

System.**out**.println("Can Run....");

}

}

**class** Employee **extends** Person

{

**void** walk()

{

System.**out**.println("Running Fast...");

}

**public** **static** **void** main(String arg[])

{

Person p=**new** Employee(); //upcasting

p.walk();

}

}