**OOAD Review Final Exam**

Midterm Semester II (Main Questions) (Role, Use, UML Class Diagram, Code)

1. ចូរនិយាយពីតួនាទី (Role) របស់​​ Decorator Design Pattern ដោយមានភ្ជាប់ឧទាហរណ៍​បញ្ចាក់។

2. ចូរនិយាយពីការប្រើប្រាស់​ (Use) Adapter design pattern ដោយមានភ្ជាប់ឧទាហរណ៍បញ្ចាក់។

3. ចូរគូរ UML Class diagram ទាក់ទងនិងបញ្ហាណាមួយ ដែលមានអនុវត្ត Prototype design pattern។

OOAD Review Final Exam

1. Creational Design Pattern:

- Prototype Pattern

- Factory Method Pattern

- Singleton Pattern

- Builder Pattern

2. Structural Design Pattern:

- Decorator Pattern

- Bridge Pattern

- Composite Pattern

- Adapter Pattern

3. Behavioral Design Pattern

- Strategy Pattern

- State Pattern

- Command Pattern

- Template Method Pattern

- Observer Pattern

Creation Design Pattern

1. Prototype Pattern:

1. ចូរនិយាយពីតួនាទី (Role) របស់​​ **Prototype** ដោយមានភ្ជាប់ឧទាហរណ៍​បញ្ចាក់។

2. ចូរនិយាយពីការប្រើប្រាស់​ (Use) **Prototype** ដោយមានភ្ជាប់ឧទាហរណ៍បញ្ចាក់។

3. ចូរគូរ UML Class diagram ទាក់ទងនិងបញ្ហាណាមួយ ដែលមានអនុវត្ត **Prototype** ។

2. Factory Method Pattern:

1. ចូរនិយាយពីតួនាទី (Role) របស់​​ **Factory Method** ដោយមានភ្ជាប់ឧទាហរណ៍​បញ្ចាក់។

2. ចូរនិយាយពីការប្រើប្រាស់​ (Use) **Factory Method** ដោយមានភ្ជាប់ឧទាហរណ៍បញ្ចាក់។

3. ចូរគូរ UML Class diagram ទាក់ទងនិងបញ្ហាណាមួយ ដែលមានអនុវត្ត **Factory Method** ។

3. Singleton Pattern:

1. ចូរនិយាយពីតួនាទី (Role) របស់​​ **Singleton** ដោយមានភ្ជាប់ឧទាហរណ៍​បញ្ចាក់។

2. ចូរនិយាយពីការប្រើប្រាស់​ (Use) **Singleton** ដោយមានភ្ជាប់ឧទាហរណ៍បញ្ចាក់។

3. ចូរគូរ UML Class diagram ទាក់ទងនិងបញ្ហាណាមួយ ដែលមានអនុវត្ត **Singleton** ។

4. Builder Pattern:

1. ចូរនិយាយពីតួនាទី (Role) របស់​​ **Builder** ដោយមានភ្ជាប់ឧទាហរណ៍​បញ្ចាក់។

2. ចូរនិយាយពីការប្រើប្រាស់​ (Use) **Builder** ដោយមានភ្ជាប់ឧទាហរណ៍បញ្ចាក់។

3. ចូរគូរ UML Class diagram ទាក់ទងនិងបញ្ហាណាមួយ ដែលមានអនុវត្ត **Builder** ។

Structural Design Pattern

5. Decorator Pattern:

1. ចូរនិយាយពីតួនាទី (Role) របស់​​ **Decorator** ដោយមានភ្ជាប់ឧទាហរណ៍​បញ្ចាក់។

2. ចូរនិយាយពីការប្រើប្រាស់​ (Use) **Decorator** ដោយមានភ្ជាប់ឧទាហរណ៍បញ្ចាក់។

3. ចូរគូរ UML Class diagram ទាក់ទងនិងបញ្ហាណាមួយ ដែលមានអនុវត្ត **Decorator** ។

6. Bridge Pattern:

1. ចូរនិយាយពីតួនាទី (Role) របស់​​ **Bridge** ដោយមានភ្ជាប់ឧទាហរណ៍​បញ្ចាក់។

2. ចូរនិយាយពីការប្រើប្រាស់​ (Use) **Bridge** ដោយមានភ្ជាប់ឧទាហរណ៍បញ្ចាក់។

3. ចូរគូរ UML Class diagram ទាក់ទងនិងបញ្ហាណាមួយ ដែលមានអនុវត្ត **Bridge** ។

7. Composite Pattern:

1. ចូរនិយាយពីតួនាទី (Role) របស់​​ **Composite** ដោយមានភ្ជាប់ឧទាហរណ៍​បញ្ចាក់។

2. ចូរនិយាយពីការប្រើប្រាស់​ (Use) **Composite** ដោយមានភ្ជាប់ឧទាហរណ៍បញ្ចាក់។

3. ចូរគូរ UML Class diagram ទាក់ទងនិងបញ្ហាណាមួយ ដែលមានអនុវត្ត **Composite** ។

8. Adapter Pattern:

1. ចូរនិយាយពីតួនាទី (Role) របស់​​ **Adapter** ដោយមានភ្ជាប់ឧទាហរណ៍​បញ្ចាក់។

2. ចូរនិយាយពីការប្រើប្រាស់​ (Use) **Adapter** ដោយមានភ្ជាប់ឧទាហរណ៍បញ្ចាក់។

3. ចូរគូរ UML Class diagram ទាក់ទងនិងបញ្ហាណាមួយ ដែលមានអនុវត្ត **Adapter** ។

Behavioral Design Pattern

9. Strategy Pattern:

1. ចូរនិយាយពីតួនាទី (Role) របស់​​ **Strategy** ដោយមានភ្ជាប់ឧទាហរណ៍​បញ្ចាក់។

2. ចូរនិយាយពីការប្រើប្រាស់​ (Use) **Strategy** ដោយមានភ្ជាប់ឧទាហរណ៍បញ្ចាក់។

3. ចូរគូរ UML Class diagram ទាក់ទងនិងបញ្ហាណាមួយ ដែលមានអនុវត្ត **Strategy** ។

10. State Pattern:

1. ចូរនិយាយពីតួនាទី (Role) របស់​​ **State** ដោយមានភ្ជាប់ឧទាហរណ៍​បញ្ចាក់។

2. ចូរនិយាយពីការប្រើប្រាស់​ (Use) **State** ដោយមានភ្ជាប់ឧទាហរណ៍បញ្ចាក់។

3. ចូរគូរ UML Class diagram ទាក់ទងនិងបញ្ហាណាមួយ ដែលមានអនុវត្ត **State** ។

11. Command Pattern:

1. ចូរនិយាយពីតួនាទី (Role) របស់​​ **Command** ដោយមានភ្ជាប់ឧទាហរណ៍​បញ្ចាក់។

2. ចូរនិយាយពីការប្រើប្រាស់​ (Use) **Command** ដោយមានភ្ជាប់ឧទាហរណ៍បញ្ចាក់។

3. ចូរគូរ UML Class diagram ទាក់ទងនិងបញ្ហាណាមួយ ដែលមានអនុវត្ត **Command** ។

12. Template Method Pattern:

1. ចូរនិយាយពីតួនាទី (Role) របស់​​ **Template Method** ដោយមានភ្ជាប់ឧទាហរណ៍​បញ្ចាក់។

2. ចូរនិយាយពីការប្រើប្រាស់​ (Use) **Template Method** ដោយមានភ្ជាប់ឧទាហរណ៍បញ្ចាក់។

3. ចូរគូរ UML Class diagram ទាក់ទងនិងបញ្ហាណាមួយ ដែលមានអនុវត្ត **Template Method** ។

13. Observer Pattern:

1. ចូរនិយាយពីតួនាទី (Role) របស់​​ **Observer** ដោយមានភ្ជាប់ឧទាហរណ៍​បញ្ចាក់។

2. ចូរនិយាយពីការប្រើប្រាស់​ (Use) **Observer** ដោយមានភ្ជាប់ឧទាហរណ៍បញ្ចាក់។

3. ចូរគូរ UML Class diagram ទាក់ទងនិងបញ្ហាណាមួយ ដែលមានអនុវត្ត **Observer** ។

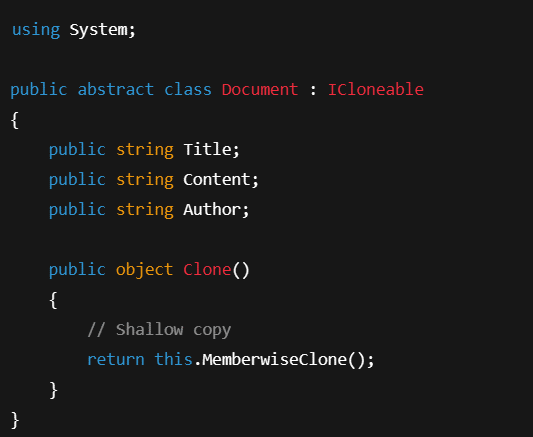
ចម្លើយសម្រាប់សំណួរ

1. ចូរនិយាយពីតួនាទីរបស់​​ Prototype និងភ្ជាប់ឧទាហរណ៍បញ្ជាក់។

- Prototype Pattern គឺជា​ Creational Design Pattern ដែលមានតួនាទីអនុញ្ញាតអោយបង្កើត Object ថ្មីដោយចម្លង (Clone) ពី Object ដែលមានរួចហើយ (Prototype) ។

* បង្កើនល្បឿន​(ការClone objects វាដើរលឿនជាងការបង្កើតថ្មី)

**ឧទាហរណ៍ ៖**  
និយាយថាយើងមាន Document class ដែលមាន properties ដូចជា title, content, author។ ប្រសិនបើអ្នកចង់បង្កើតឯកសារថ្មីដែលដូចនឹងឯកសារមួយ (សម្រាប់ Template ឯកសារ) អ្នកអាចប្រើ clone() ដោយមិនចាំបាច់បង្កើតថ្មីពីដើម។

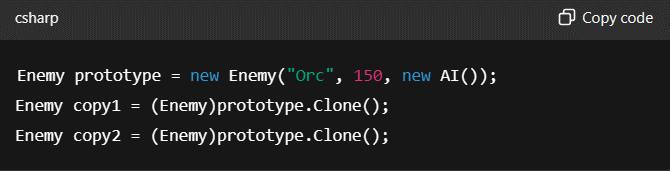


2. ចូរនិយាយពីការប្រើប្រាស់​ (Use) Prototypeដោយមានភ្ជាប់ឧទាហរណ៍បញ្ចាក់។

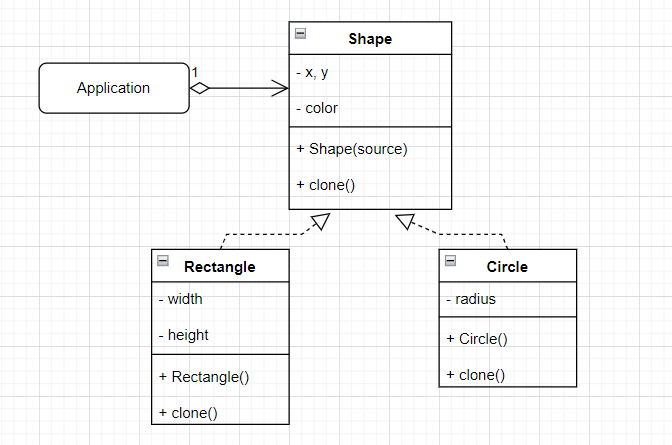
ពេលណា Prototype គួរត្រូវបានប្រើ៖

* នៅពេល object ត្រូវបានបង្កើតដោយស្មុគស្មាញ។
* នៅពេលដែលសមាសភាពលើ object គឺ dynamic (ផ្លាស់ប្តូរតាម runtime)។
* នៅពេលអ្នកចង់ duplicate object ដោយមិនចាំបាច់ដឹងថាវាជា subclass ណា។

**ឧទាហរណ៍ ៖**  
និយាយថា Game មួយមាន Enemy Object ដែលមានប្រព័ន្ធ AI ស្មុគស្មាញ។ បើអ្នកចង់បង្កើត Enemy ច្រើន ដោយប្រើឧបករណ៍ដូចគ្នា អ្នកអាចប្រើ Prototype ដោយ clone ពី original Enemy object។

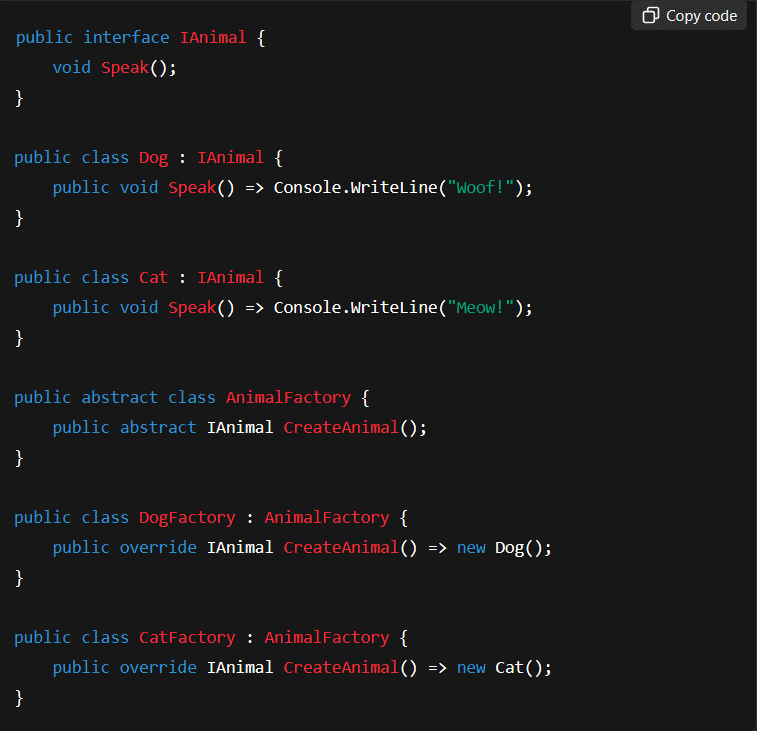


3. ចូរគូរ UML Class diagram ទាក់ទងនិងបញ្ហាណាមួយ ដែលមានអនុវត្ត Prototype ។



**1.** ចូរនិយាយពីតួនាទី (Role) របស់​​ Factory Method ដោយមានភ្ជាប់ឧទាហរណ៍​បញ្ចាក់។

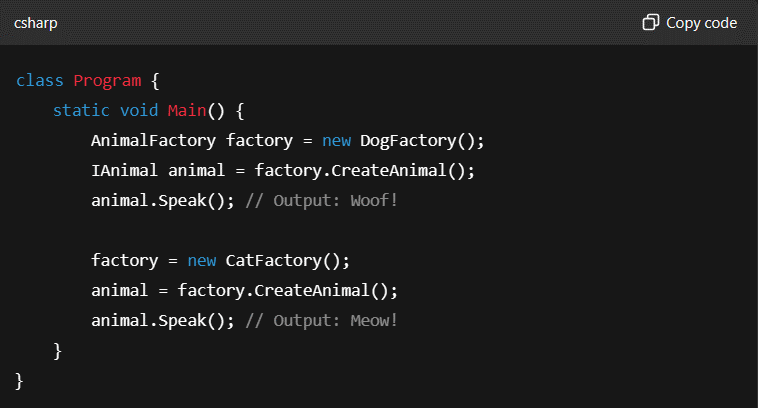
- Factory Method Pattern គឺជា​ Creational Design Pattern ដែលមានតួនាទី​ផ្តល់នូវ Interface ដើម្បីបង្កើត Object ប៉ុន្តែអនុញ្ញាតអោយ​ subclass ជ្រើសរើសថា តើត្រូវបង្កើត Object ពី class ណា។



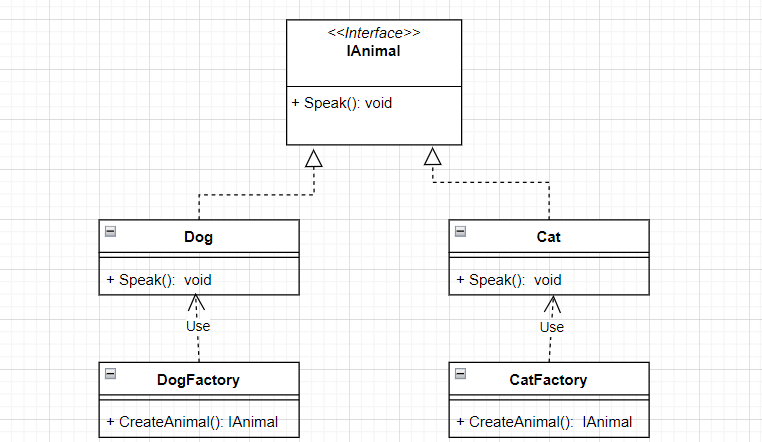
**2.** ចូរនិយាយពីការប្រើប្រាស់​ (Use) Factory Method ដោយមានភ្ជាប់ឧទាហរណ៍បញ្ចាក់។

គួរប្រើ Factory Method Pattern នៅពេលដែល:

* ចង់បង្កើត objects បែប dynamic ពី class ផ្សេងៗ
* ចង់ជៀសវាងការប្រើ new ដោយផ្ទាល់ក្នុង client code
* ចង់អោយការបន្ថែម type ថ្មីៗមិនប៉ះពាល់ទៅលើ code ចាស់ (Open/Closed Principle)

**ឧទាហរណ៍ ៖**  


**3.** ចូរគូរ UML Class diagram ទាក់ទងនិងបញ្ហាណាមួយ ដែលមានអនុវត្ត Factory Method។



1. ចូរនិយាយពីតួនាទី (Role) របស់​​ Singleton ដោយមានភ្ជាប់ឧទាហរណ៍​បញ្ចាក់។

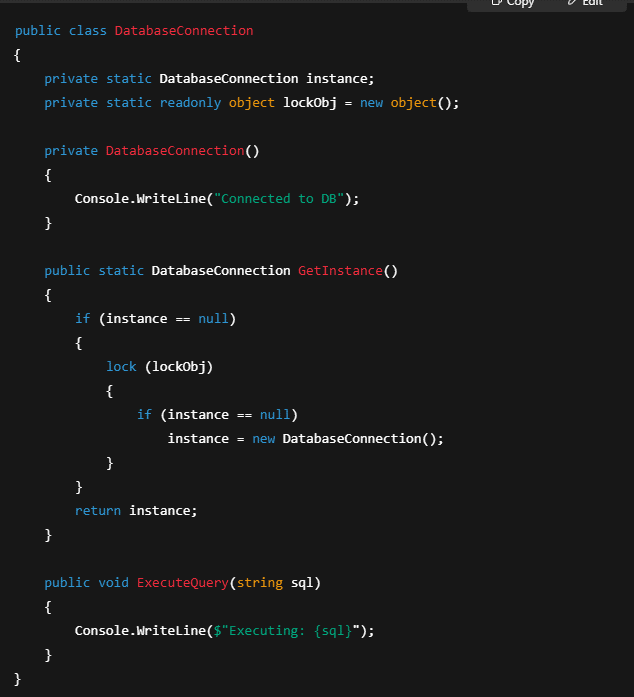
- Singleton Pattern គឺជា​ Creational Design Pattern ដែលមានតួនាទីដូចជា៖

* បង្កើត object តែមួយ (one and only one instance) សម្រាប់ class មួយ។
* ផ្ដល់ access point មួយតែមួយ (global access point) ដើម្បីឲ្យ client ទាំងអស់ប្រើ object នោះ។
* ការគ្រប់គ្រង instance មិនមែនអោយ client ទទួលខុសត្រូវទេ ប៉ុន្តែ class ខ្លួនវាដែលគ្រប់គ្រងការបង្កើតនិងចូលដំណើរការ។

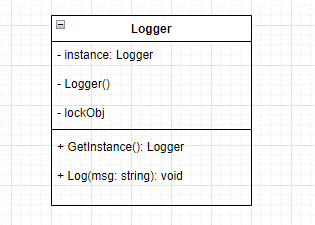


2. ចូរនិយាយពីការប្រើប្រាស់​ (Use) Singleton ដោយមានភ្ជាប់ឧទាហរណ៍បញ្ចាក់។

* ត្រូវការតែ instance មួយ (e.g. Logger, Configuration, Database connection)។
* ត្រូវការគ្រប់គ្រង access ច្បាស់លាស់ទៅកាន់ instance នោះ។
* ការបង្កើត multiple instances អាចបង្ករបញ្ហា (e.g. database conflict, inconsistent settings)។
* ការគ្រប់គ្រងគួរអោយ class ដៃខ្លួនវាធ្វើទៅវិញ មិនមែន client ។

**ឧទាហរណ៍ ៖**  


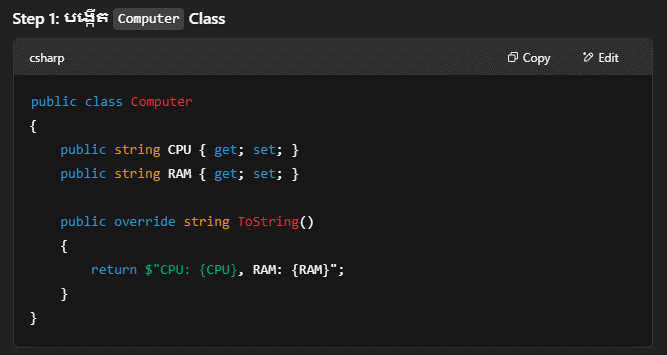
3. ចូរគូរ UML Class diagram ទាក់ទងនិងបញ្ហាណាមួយ ដែលមានអនុវត្ត Singleton ។

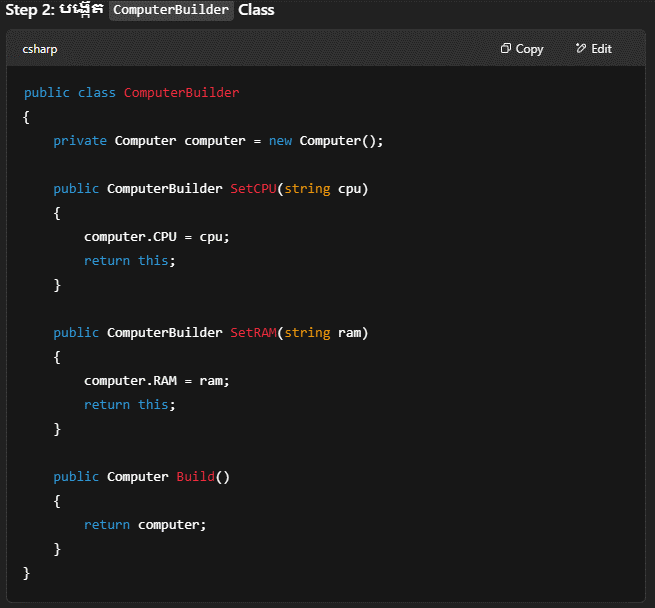


1. ចូរនិយាយពីតួនាទី (Role) របស់​​ Builder ដោយមានភ្ជាប់ឧទាហរណ៍​បញ្ចាក់។

- Builder Pattern គឺជា​ Creational Design Pattern ដែលមានតួនាទី​ដូចខាងក្រោម៖

* គ្រប់គ្រងដំណើរការបង្កើត object ស្មុគស្មាញ (complex objects)
* មានសារៈសំខាន់នៅពេលដែល object មាន **property ច្រើន**, ហើយ client មិនចង់ប្រើ constructor ដែលមាន parameter ជាច្រើន

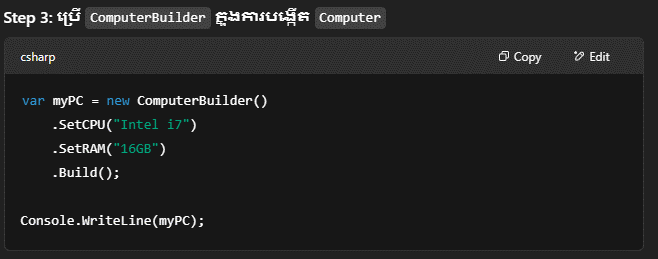




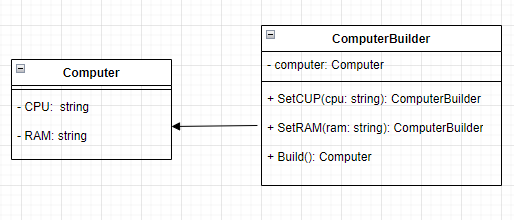
2. ចូរនិយាយពីការប្រើប្រាស់​ (Use) Builder ដោយមានភ្ជាប់ឧទាហរណ៍បញ្ចាក់។

គួរប្រើ Builder Pattern នៅពេលដែល:

* Builder គឺល្អសម្រាប់ការបង្កើត complex objects ដែលមាន parts ឬ components ច្រើន ហើយអាចត្រូវការការបង្កើតបែបផ្សេងៗ។
* វាផ្តល់នូវ flexibility និង fine control លើ steps នៃការបង្កើត។
* Builder pattern អាចនាំឲ្យមាន separation of concerns ពីការបង្កើតនិងការប្រើប្រាស់ object ដោយប្រើ step-by-step process។

**ឧទាហរណ៍ ៖**  


3. ចូរគូរ UML Class diagram ទាក់ទងនិងបញ្ហាណាមួយ ដែលមានអនុវត្ត Builder ។



1. ចូរនិយាយពីតួនាទី (Role) របស់​​ Decorator ដោយមានភ្ជាប់ឧទាហរណ៍​បញ្ចាក់។

- Decorator Pattern គឺជា​ Structural Design Pattern ដែលមានតួនាទី​បន្ថែមសមត្ថភាព​(behavior) or state​ ថ្មីៗទៅកាន់ Object មួយ​ដោយគ្មានការផ្លាស់ប្តូរ Code នៅក្នុង class ដើម។

* Object មិនដឹងថាកំពុងត្រូវបាន decorate (Transparent Extension)​​។
* ប្រើបានសម្រាប់បន្ថែម feature ដោយមិនចាំបាច់សរសេរ subclass ច្រើន។
* អនុវត្តដោយ **wrap object មួយជាមួយ class ថ្មី** ដែលមានសមត្ថភាពបន្ថែម។

ឧទាហរណ៍៖

2. ចូរនិយាយពីការប្រើប្រាស់​ (Use) Decorator ដោយមានភ្ជាប់ឧទាហរណ៍បញ្ចាក់។

Decorator ត្រូវបានប្រើនៅពេល៖

* ចង់បន្ថែម behavior or feature ទៅកាន់ object មួយៗ ជា dynamic
* យើងចង់ធ្វើឲ្យ object ខ្លះមា behavior ផ្សេងៗដោយ **មិនប៉ះពាល់ដល់ others។**
* យើងមិនចង់បង្កើត subclass ច្រើនពេក

ឧទាហរណ៍៖

3. ចូរគូរ UML Class diagram ទាក់ទងនិងបញ្ហាណាមួយ ដែលមានអនុវត្ត Decorator ។

**ឧទាហរណ៍ ៖**

1. ចូរនិយាយពីតួនាទី (Role) របស់​​ Bridge ដោយមានភ្ជាប់ឧទាហរណ៍​បញ្ចាក់។

- Bridge Pattern គឺជា​ Structural Design Pattern ដែលមានតួនាទី​

* Bridge pattern មានតួនាទីក្នុងការបំបែក **Abstraction** ចេញពី **Implementation** ដើម្បីអាចអភិវឌ្ឍដោយឯករាជ្យ។
* ប្រើពេលដែលអ្នកចង់បន្ថែម version ថ្មី មិនប៉ះពាល់ដល់ version ចាស់។
* ជៀសវាងការបង្កើត subclass ច្រើនពេកព្រោះការលាយបញ្ចូល Abstraction និង Implementation ទាំងពីរ

ឧទាហរណ៍៖

2. ចូរនិយាយពីការប្រើប្រាស់​ (Use) Bridge ដោយមានភ្ជាប់ឧទាហរណ៍បញ្ចាក់។

Decorator ត្រូវបានប្រើនៅពេល៖

* អ្នកចង់លាក់ Implementation ពី Client។
* ចង់ជៀសវាងការចងបណ្តាញជាប់គ្នារវាង Abstraction និង Implementation។
* ចង់ផ្លាស់ប្តូរ Implementation ដោយមិនប៉ះពាល់ទៅ Abstraction (e.g. ដោយមិនបង្ហាញអ្នកប្រើ)។
* បង្កើតការចម្រុះ runtime ជាមួយ component ផ្សេងៗ។

ឧទាហរណ៍៖

3. ចូរគូរ UML Class diagram ទាក់ទងនិងបញ្ហាណាមួយ ដែលមានអនុវត្ត Bridge ។

ឧទាហរណ៍៖

1. ចូរនិយាយពីតួនាទី (Role) របស់​​ Adapter ដោយមានភ្ជាប់ឧទាហរណ៍​បញ្ជាក់។

- Adapter Pattern គឺជា​ Structural Design Pattern ដែលមានតួនាទីដូចខាងក្រោម៖

Adapter មានតួនាទីក្នុងការចងភ្ជាប់ **interface** មួយទៅជាផ្សេង ដើម្បីឲ**្យ client** អាចប្រើប្រាស់ **class** ដែលវាធ្វើការមិនបានស្រប interface។ វាជារបៀបធ្វើឲ្យ **object ដែលមាន interface មិនត្រូវគ្នា** អាចធ្វើការរួមគ្នាបាន។

ឧទាហរណ៍៖

2. ចូរនិយាយពីការប្រើប្រាស់​ (Use) Adapter ដោយមានភ្ជាប់ឧទាហរណ៍បញ្ជាក់។

Adapter ត្រូវបានប្រើប្រាស់នៅពេលដែលយើងចង់ **ប្រើប្រាស់ class ដើម** (Adaptee) ដែលមាន interface មិនស្របនឹង system ឬ client ដែលមានរួច។ វាជួយសន្សំពេលក្នុងការផ្លាស់ប្តូរ code ដើម។

* ប្រើនៅពេល reuse legacy code
* ធ្វើឲ្យបរិបទ system ថ្មី និងចាស់ធ្វើការផ្សារភ្ជាប់បាន

ឧទាហរណ៍៖

3. ចូរគូរ UML Class diagram ទាក់ទងនិងបញ្ហាណាមួយ ដែលមានអនុវត្ត Adapter ។

ឧទាហរណ៍៖

Behavioral Design Pattern

9. Strategy Pattern:

1. ចូរនិយាយពីតួនាទី (Role) របស់​​ **Strategy** ដោយមានភ្ជាប់ឧទាហរណ៍​បញ្ចាក់។

2. ចូរនិយាយពីការប្រើប្រាស់​ (Use) **Strategy** ដោយមានភ្ជាប់ឧទាហរណ៍បញ្ចាក់។

3. ចូរគូរ UML Class diagram ទាក់ទងនិងបញ្ហាណាមួយ ដែលមានអនុវត្ត **Strategy** ។

10. State Pattern:

1. ចូរនិយាយពីតួនាទី (Role) របស់​​ **State** ដោយមានភ្ជាប់ឧទាហរណ៍​បញ្ចាក់។

2. ចូរនិយាយពីការប្រើប្រាស់​ (Use) **State** ដោយមានភ្ជាប់ឧទាហរណ៍បញ្ចាក់។

3. ចូរគូរ UML Class diagram ទាក់ទងនិងបញ្ហាណាមួយ ដែលមានអនុវត្ត **State** ។

11. Command Pattern:

1. ចូរនិយាយពីតួនាទី (Role) របស់​​ **Command** ដោយមានភ្ជាប់ឧទាហរណ៍​បញ្ចាក់។

2. ចូរនិយាយពីការប្រើប្រាស់​ (Use) **Command** ដោយមានភ្ជាប់ឧទាហរណ៍បញ្ចាក់។

3. ចូរគូរ UML Class diagram ទាក់ទងនិងបញ្ហាណាមួយ ដែលមានអនុវត្ត **Command** ។

12. Template Method Pattern:

1. ចូរនិយាយពីតួនាទី (Role) របស់​​ **Template Method** ដោយមានភ្ជាប់ឧទាហរណ៍​បញ្ចាក់។

2. ចូរនិយាយពីការប្រើប្រាស់​ (Use) **Template Method** ដោយមានភ្ជាប់ឧទាហរណ៍បញ្ចាក់។

3. ចូរគូរ UML Class diagram ទាក់ទងនិងបញ្ហាណាមួយ ដែលមានអនុវត្ត **Template Method** ។

13. Observer Pattern:

1. ចូរនិយាយពីតួនាទី (Role) របស់​​ **Observer** ដោយមានភ្ជាប់ឧទាហរណ៍​បញ្ចាក់។

2. ចូរនិយាយពីការប្រើប្រាស់​ (Use) **Observer** ដោយមានភ្ជាប់ឧទាហរណ៍បញ្ចាក់។

3. ចូរគូរ UML Class diagram ទាក់ទងនិងបញ្ហាណាមួយ ដែលមានអនុវត្ត **Observer** ។

1. ចូរនិយាយពីតួនាទី (Role) របស់​​ Strategy ដោយមានភ្ជាប់ឧទាហរណ៍​បញ្ចាក់។

**Strategy Pattern គឺ**ជា Behavioral Design Pattern ដែលមានតួនាទី  
➡️ បំបែក algorithm ឬ logic ខុសៗគ្នា ទៅជាក្លាសដាច់ដោយឡែក ហើយអនុញ្ញាតឲ្យប្តូរប្រតិបត្តិការទាំងនោះបាន **dynamically នៅពេល runtime**។

* Define family of algorithms, encapsulate each one
* ធ្វើការប្តូរយុទ្ធសាស្ត្របានសេរី ដោយមិនប៉ះពាល់ទៅ client
* ប្រើ polymorphism ដើម្បី បន្លាយ behavior ដោយមិនប្រើ if-else or switch

ឧទាហរណ៍៖

2. ចូរនិយាយពីការប្រើប្រាស់​ (Use) Strategy ដោយមានភ្ជាប់ឧទាហរណ៍បញ្ចាក់។

**Strategy Pattern ត្រូវបានប្រើនៅពេល៖**  
✅ អ្នកមាន algorithm ច្រើនដែលអាចផ្លាស់ប្តូរបានដោយអាស្រ័យលើស្ថានភាព  
✅ ចង់ជៀសវាង if-else/switch statements  
✅ ចង់រចនា system ឲ្យ flexible និង extensible ជាងមុន

ឧទាហរណ៍៖

3. ចូរគូរ UML Class diagram ទាក់ទងនិងបញ្ហាណាមួយ ដែលមានអនុវត្ត Strategy ។

ឧទាហរណ៍៖

1. ចូរនិយាយពីតួនាទី (Role) របស់​​ State ដោយមានភ្ជាប់ឧទាហរណ៍​បញ្ចាក់។

State Pattern គឺជា *Behavioral Design Pattern* ដែលមានតួនាទី  
➡️ *អនុញ្ញាតឲ្យ object មួយ ប្តូរប្រព្រឹត្តិការណ៍របស់វា (behavior) ដោយផ្អែកលើ state ផ្ទាល់ខ្លួន។*

**ចំណុចសំខាន់៖**

* បង្កើត class ផ្សេងៗសម្រាប់ state ដូចជា “Start”, “Stop”, “Pause”...
* ប្តូរប្រព្រឹត្តិការណ៍បាន ដោយគ្រាន់តែប្តូរ state object ទៅ class ថ្មី
* ជៀសវាង if-else/switch statement ដល់ state logic

ឧទាហរណ៍៖

2. ចូរនិយាយពីការប្រើប្រាស់​ (Use) State ដោយមានភ្ជាប់ឧទាហរណ៍បញ្ចាក់។

**State Pattern ត្រូវបានប្រើនៅពេល៖**  
✅ Behavior របស់ object ផ្លាស់ប្តូរផ្អែកលើ internal state  
✅ ចង់ជៀសវាង logic with many if-else/switch  
✅ ចង់ឲ្យ code មាន structure មុតមាំ និងត្រូវរីករាយពេលបន្ថែម state ថ្មី

ឧទាហរណ៍៖

3. ចូរគូរ UML Class diagram ទាក់ទងនិងបញ្ហាណាមួយ ដែលមានអនុវត្ត State ។

10. State Pattern:

ឧទាហរណ៍៖

1. ចូរនិយាយពីតួនាទី (Role) របស់​​ Command ដោយមានភ្ជាប់ឧទាហរណ៍​បញ្ចាក់។

Command Pattern មានតួនាទី  
➡️ *បម្លែងពាក្យបញ្ជា (request) ឬការប្រព្រឹត្តិការណ៍មួយទៅជា object ដាច់ដោយឡែក ដែលអាចរក្សា, ផ្ញើ, និងអនុវត្តបាននៅពេលក្រោយ។*

ចំណុចសំខាន់៖

* បំបែក sender និង receiver ដើម្បីអាច manage request នៅពេលក្រោយ
* អាចធ្វើ Undo/Redo, queue commands, log history
* Encapsulate action ជា object

ឧទាហរណ៍៖

2. ចូរនិយាយពីការប្រើប្រាស់​ (Use) Command ដោយមានភ្ជាប់ឧទាហរណ៍បញ្ចាក់។

**Command Pattern ត្រូវបានប្រើនៅពេល៖**  
✅ ចង់បំបែក logic រវាង sender និង receiver  
✅ ចង់ធ្វើ Undo, Redo, logging, scheduling  
✅ ចង់ encapsulate ការប្រព្រឹត្តិ action ក្នុង object ដូចជា queue / transaction

ឧទាហរណ៍៖

3. ចូរគូរ UML Class diagram ទាក់ទងនិងបញ្ហាណាមួយ ដែលមានអនុវត្ត Command ។

ឧទាហរណ៍៖

1. ចូរនិយាយពីតួនាទី (Role) របស់​ Template Method ដោយមានភ្ជាប់ឧទាហរណ៍​បញ្ចាក់។

Template Method Pattern គឺជា *Behavioral Design Pattern* ដែលកំណត់ algorithm រួមដែលចែកចាយការងារផ្សេងៗមកឲ្យខាងក្នុង subclass ដើម្បីបង្កើត behavior របស់វា។

Role:

* កំណត់ algorithm មួយសម្រាប់ការអនុវត្តន៍វ៉ាងសំខាន់ៗ
* កំណត់ការលំអិតនៅក្នុង method មួយដែលអាចត្រូវបាន overriden ឬត្រូវបានធ្វើអោយពុម្ពដោយ subclass
* ជួយអោយ code រឹងមាំនិងងាយស្រួលសម្រាប់ការប្រើប្រាស់ក្រោម context ដែលដូចគ្នា

ឧទាហរណ៍៖

2. ចូរនិយាយពីការប្រើប្រាស់​ (Use) Template method ដោយមានភ្ជាប់ឧទាហរណ៍បញ្ចាក់។

**Template Method Pattern ត្រូវបានប្រើនៅពេល៖**  
✅ យើងមាន algorithm ដែលមានភាពធម្មតា ដែលអ្នកចង់កំណត់ structure តែចំណុចខ្លះៗអាចផ្លាស់ប្តូរបាន  
✅ ចង់ធ្វើការ overriden ការអនុវត្តខ្លះៗក្នុង subclass ប៉ុន្តែចង់រក្សាទុក structure គេយ៉ាងដូចជា  
✅ មានការងារដែលធ្វើជារបៀបសម្រាប់ប្រភេទផ្សេងៗ (ដូចជា parsing ទិន្នន័យ JSON និង XML)  
✅ យើងចង់ប្រើ **common algorithm** ប៉ុន្តែអនុវត្តលម្អិតនៅក្នុង subclass។

ឧទាហរណ៍៖

3. ចូរគូរ UML Class diagram ទាក់ទងនិងបញ្ហាណាមួយ ដែលមានអនុវត្ត template method ។

ឧទាហរណ៍៖

1. ចូរនិយាយពីតួនាទី (Role) របស់​​ Adapter ដោយមានភ្ជាប់ឧទាហរណ៍​បញ្ជាក់។

Observer Pattern គឺជា *Behavioral Design Pattern* ដែលអនុញ្ញាតឲ្យ system ទំនាក់ទំនងជាមួយ object មួយចំនួន (observers) ដែលត្រូវបានចុះបញ្ជី ដើម្បីទទួលការអាប់ដេតនៅពេលដែល subject មួយមានការប្រែប្រួល (change)។

* Subject មានការ update (notify) observers នៅពេលដែលមានការប្រែប្រួល។
* Observer គឺជា object ដែលត្រូវបានចុះបញ្ជីរួមគ្នាជាមួយ subject ហើយបង្ហាញនូវទិន្នន័យថ្មីៗនៅពេលដែល subject ផ្លាស់ប្តូរ។

ឧទាហរណ៍៖

2. ចូរនិយាយពីការប្រើប្រាស់​ (Use) Observer ដោយមានភ្ជាប់ឧទាហរណ៍បញ្ជាក់។

**Observer Pattern ត្រូវបានប្រើនៅពេល៖**  
✅ យើងចង់មាន notification ជាមួយមនុស្សជាច្រើន នៅពេលដែល **subject** ផ្លាស់ប្តូរ។  
✅ ចង់អោយ object មួយនៅស្ថិតក្នុងភាព loosely coupled និងធ្វើការពង្រីកប្រព័ន្ធដោយមិនបង្ខំឲ្យការផ្លាស់ប្តូរនៅក្នុង observer និង subject។  
✅ ចង់តាមដានភាពផ្លាស់ប្តូរនៃ data នៅពេលដែលមានការផ្លាស់ប្តូរ។

ឧទាហរណ៍៖

3. ចូរគូរ UML Class diagram ទាក់ទងនិងបញ្ហាណាមួយ ដែលមានអនុវត្ត Adapter ។