Тест до практичної роботи 5 ІПЗ

Total points 13/21

Встановити відповідність між паттерном проектування та його описом

	Strategy	Observer	Command	Chain of Responsibility	State	Score
Визначає залежність один-до-багатьох між об'єктами так, що коли один об'єкт змінює стан, всі йому залежні об'єкти автоматично отримують сповіщення і оновлюються.	○	◉	○	○	○	0/0
Інкапсулює запит на виклик метода як об'єкт, що дозволяє параметризувати клієнтів з різними запитами, ставити запити в чергу, а також підтримувати відміну операцій.	○	○	◉	○	○	1/1
Визначає сімейство алгоритмів, інкапсулює кожен з них і робить їх взаємозамінними. Користувач може обирати один з алгоритмів на льоту.	◉	○	○	○	○	0/0
Дозволяє передавати запити послідовно в ланцюжку обробників. Кожен обробник вирішує, чи може він обробити запит, або передає його наступному обробнику в ланцюжку.	○	○	○	◉	○	1/1
Дозволяє об'єкту змінювати свою поведінку при зміні внутрішнього стану. Виглядає так, ніби об'єкт міняє свій клас під час виконання.	○	○	○	○	◉	1/1

Встановити відповідність між паттерном проектування та його описом

	Interpreter	Mediator	Memento	Visitor	Iterator	Score	
Дозволяє зберігати та відновлювати попередні стани об'єкта без розкриття його внутрішнього представлення.	○	○	◉	○	○	1/1	✓
Дозволяє визначити нову операцію, не змінюючи класи об'єктів, над якими вона виконується.	○	○	○	◉	○	1/1	✓
Використовується для виконання операцій або обчислень, визначених в текстовому вигляді.	◉	○	○	○	○	0/0	✓
Визначає інтерфейс для обходу колекцій та реалізацію цього інтерфейсу в кожній конкретній колекції.	○	○	○	○	◉	1/1	✓
Визначає об'єкт, який координує взаємодію між об'єктами, що мають слабку залежність один від одного.	○	◉	○	○	○	0/0	✓

✓ Яка з наступних структур не є складовою частиною паттерна "Медіатор"? 1/1

- ◉ Відвідувач ✓
- ○ Конкретний медіатор
- ○ Колега
- ○ Конкретний колега
- ○ Медіатор

✓ Які методи часто присутні в інтерфейсі медіатора? 1/1

- ○ connect, disconnect
- ○ request, handle
- ○ notify, update
- ◉ send, receive ✓
- ○ publish, subscribe

✓ Які основні методи часто присутні в інтерфейсі ітератора? 1/1

- ○ iterate(), hasMore()
- ◉ next(), hasNext() ✓
- ○ get(), moveNext()
- ○ current(), advance()
- ○ fetch(), hasElement()

✓ Які основні компоненти паттерну Спостерігач? 1/1

- Watcher, Watched, Trigger, Reaction
- Publisher, Subscriber, Event, Handler
- **Subject, Observer, ConcreteSubject, ConcreteObserver** ✓
- Source, Sink, Connector, Receiver
- Parent, Child, Listener, Notifier

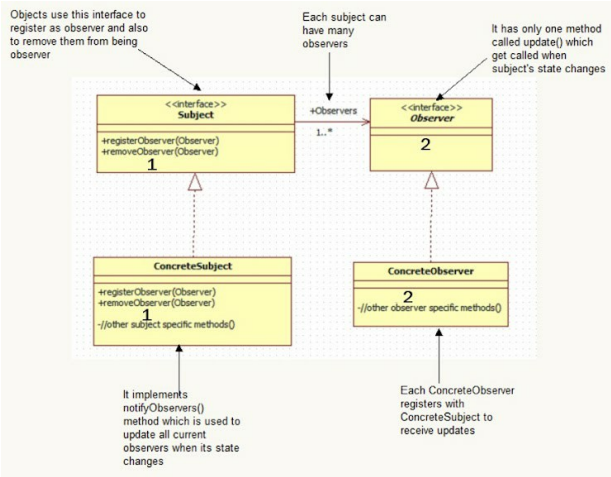
✓ Яка з наведених мов програмування не підтримує інтерфейси (interfaces) та може мати ускладнену реалізацію паттерну Observer? 1/1

- Ruby
- **C** ✓
- Java
- Swift

✓ Як можна використовувати паттерн Observer для реалізації системи сповіщень в мобільному додатку? 1/1

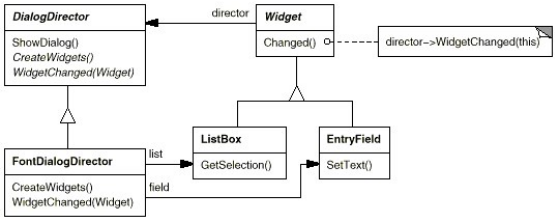
- Використання опитування (polling) для періодичної перевірки стану
- Використання WebSocket для миттєвої передачі змін
- Використання Bluetooth-сполучень
- **Використання push-сповіщень (push notifications)** ✓
- Використання SMS-повідомлень

✓ Які методи слід дописати в 1 та 2? 1/1



- **1. notify() 2. update()** ✓
- 1. receive() 2. send()
- 1. iterator() 2. next()
- 1. send() 2. receive()
- 1. update() 2. notify()
- 1. next() 2. iterator()

✕ Якому шаблону проектування відповідає дана схема? Встановити відповідність між класом або інтерфейсом на діаграмі нижче та компонентом (учасником) шаблону. ·/ ·3 .



Перше це Посередник, це інтерфейс, що координує взаємодії між компонентами.

Там де DialogDirector - ConcreteMediator, це конкретна реалізація посередника.

FontDialogDirector, це Colleague, це компонент, що взаємодіє з іншими компонентами через посередника.

Widget та його реалізації: ListBox і EntryField - ConcreteColleague — конкретна реалізація колег, які передають свої зміни через посередника.

Feedback

DialogDirector - mediator
FontDialogDirector - concreteMediator
Widget - colleague
ListBox, EntryField - concreteColleagues

✕ Дописати методи **hasNext() та next()** в реалізації ітератора зв'язного списку. ·· / · 5

```
class LinkedList<T> implements Iterable<T> {
    6 usages
    private class Node<T> {...}
    4 usages
    private Node<T> head;
    //Додавання в кінець списку
    no usages
    public void add(T data) {...}
    1 usage
    private class LinkedListIterator implements Iterator<T> {
        1 usage
        private Node<T> current;
        1 usage
        LinkedListIterator() {
            this.current = head;
        }
        @Override
        public boolean hasNext() {
            //...
        }

        @Override
        public T next() {
            //...
        }
    }
    @Override
    public Iterator<T> iterator() {
        return new LinkedListIterator();
    }
}
```

```
@Override
public boolean hasNext() {
  return current != null;
}

@Override
public T next() {
  if (!hasNext()) {
    throw new NoSuchElementException();
  }
  T data = current.data;
  current = current.next;

  return data;
}
```

Feedback

```
@Override
public boolean hasNext() {
return current != null;
}
@Override
public T next() {
if (hasNext()) {
T data = current.data;
current = current.next;
return data;
}
}
```

Quilgo Test ID *

This question is filled automatically DO NOT EDIT OR REMOVE

 Skip this question DO NOT REMOVE OR EDIT! ### JANGAN HAPUS ATAU EDIT! ### ¡NO QUITAR NI EDITAR! ### hataen ya sampaadit na karen! ### @#@#@JRxga9VaxkUdoS6z