Тест до практичної роботи 8 ІПЗ

Total points 12/23

Email *

collar-supply-yarn@duck.com

ПІБ *

Трамп Дональд Олексійович

Група *

ІО-35

Встановити відповідність міжроджувальним шаблоном та його описом

	визначає інтерфейс для створення об'єктів у суперкласі, але залишає підкласам можливість змінювати тип створюваних об'єктів	надає інтерфейс для створення сімейств пов'язаних або залежних об'єктів без зазначення їхніх конкретних класів	використовується для конструювання складних об'єктів крок за кроком	гарантує, що у класу існує лише один екземпляр, і забезпечує глобальну точку доступу до цього екземпляра	дозволяє створювати нові об'єкти, використовуючи існуючий екземпляр об'єкта як шаблон	Score
Singleton	○	○	○	◉	○	0/0
Abstract factory	○	◉	○	○	○	1/1
Builder	○	○	◉	○	○	1/1
Prototype	○	○	○	○	◉	0/0
Factory Method	◉	○	○	○	○	1/1

✓ Як можна отримати доступ до єдиного екземпляра класу Singleton? 1/1

- ○ Звертаючись до абстрактного методу
- ○ Викликаючи конструктор класу Singleton()
- ◉ Викликаючи статичний метод getInstance() ✓
- ○ Використовуючи фабричний метод
- ○ Створюючи новий екземпляр кожного разу

✓ Який тип об'єкта отримаємо після клонування за допомогою шаблону Прототип? 1/1

- ○ Тип, який завжди є абстрактним
- ○ Тип, який залежить від паттерну фабрики
- ○ Різний тип від оригіналу
- ◉ Тип, який збігається з оригіналом ✓

✓ Як шаблон Прототип взаємодіє зі зміненими об'єктами? 1/1

- Зміни вносяться тільки після перезавантаження програми
- Зміни автоматично впливають на оригінал
- Зміни не впливають на оригінал
- Зміни потрібно вручну передавати об'єктові-оригіналу
- ◉ Зміни вносяться тільки в клонованому об'єкті ✓

✓ Яким чином фабричний метод створює продукт? 1/1

- Викликає метод іншої фабрики для створення продукту
- Викликає конструктор продукту напряму
- Викликає статичний метод продукту
- ◉ Викликає метод фабрики, який повертає новий об'єкт ✓
- Використовує рекурсивний метод для створення продукту

✗ Чи може одна фабрика створювати кілька різних продуктів в шаблоні фабричного методу? 0/1

- (●) Ні, кожен продукт повинен мати свою власну фабрику ✗
- ( ) Так, якщо клієнт вказує, який продукт створювати.
- ( ) Так, якщо продукти реалізують один і той же інтерфейс
- ( ) Ні, кожна фабрика має власний набір продуктів
- ( ) Так, якщо продукти мають спільний батьківський клас

✓ Яким чином клієнт створює об'єкти за допомогою абстрактної фабрики? 1/1

- (●) Викликає методи фабрики, які повертають нові об'єкти ✓
- ( ) Використовує оператор "new" для створення екземплярів
- ( ) Викликає конструктори конкретних продуктів напряму
- ( ) Використовує статичні методи класу
- ( ) Використовує глобальні функції для створення об'єктів

✓ Як можна змінити конфігурацію об'єкта, який будується за допомогою шаблону Builder, без зміни клієнтського коду? 1/1

- ○ Змінюючи конструктор об'єкта
- ○ Викликаючи методи будівництва в іншому порядку
- ○ Впливаючи на логіку клієнтського коду
- ◉ Додаючи нові методи будівництва до інтерфейсу Builder ✓

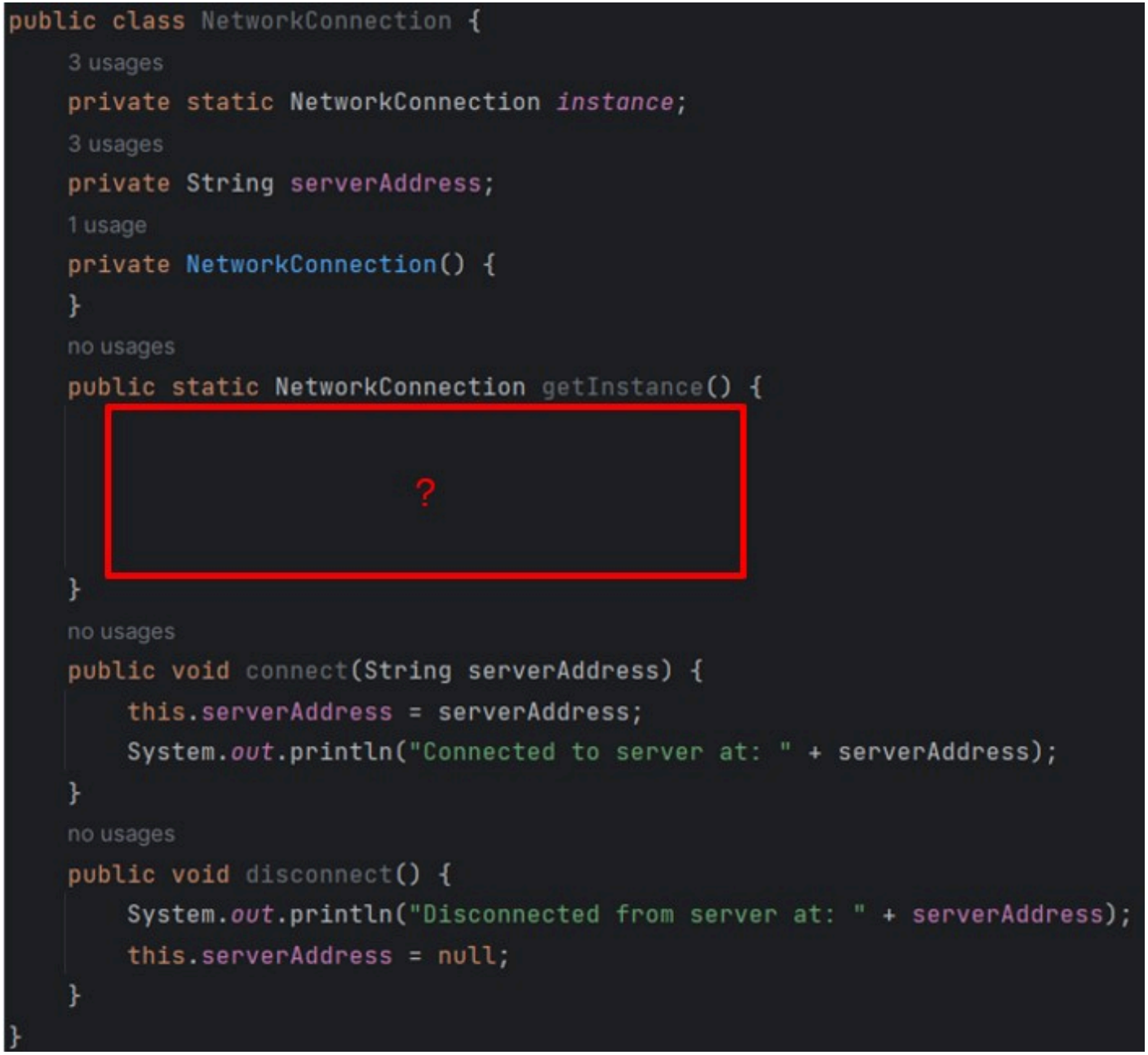
✓ Що входить до складу шаблону Factory method? 2/2

- ☑ Фабрика ✓
- ☑ Продукт ✓
- ☐ Конкретний будівельник
- ☐ Абстрактна фабрика
- ☐ Прототип
- ☑ Конкретна фабрика ✓
- ☐ Реєстр прототипів
- ☑ Конкретний продукт ✓
- ☐ Директор

Встановити відповідність між елементом шаблону Abstract factory та його описом

	Це абстрактний клас або інтерфейс, який визначає стандартну поведінку для всіх продуктів одного сімейства.	Це клас (або класи), який реалізує абстрактний клас чи інтерфейс та представляє конкретну реалізацію продукту.	Це абстрактний клас або інтерфейс, який визначає методи для створення всіх продуктів одного сімейства.	Це клас, який успадковує від абстрактного класу або реалізує інтерфейс та надає конкретну реалізацію методів для створення продуктів.	Це клас, який використовує продукти, створені абстрактною фабрикою.	Score
AbstractProduct	✓	○	◉	○	○	0/1
AbstractFactory	◉	○	✓	○	○	0/0
Client	○	○	○	○	◉	0/0
ConcreteProduct	○	◉	○	○	○	1/1
ConcreteFactory	○	○	○	◉	○	0/0

✕ Доповнити код методу **getIstance()** відповідно до шаблону **Singleton** ···/3



```
public class NetworkConnection {
    3 usages
    private static NetworkConnection instance;
    3 usages
    private String serverAddress;
    1 usage
    private NetworkConnection() {
    }
    no usages
    public static NetworkConnection getInstance() {
        ?
    }
    no usages
    public void connect(String serverAddress) {
        this.serverAddress = serverAddress;
        System.out.println("Connected to server at: " + serverAddress);
    }
    no usages
    public void disconnect() {
        System.out.println("Disconnected from server at: " + serverAddress);
        this.serverAddress = null;
    }
}
```

✕ Створити екземпляр класу Product, використовуючи builder ···/3

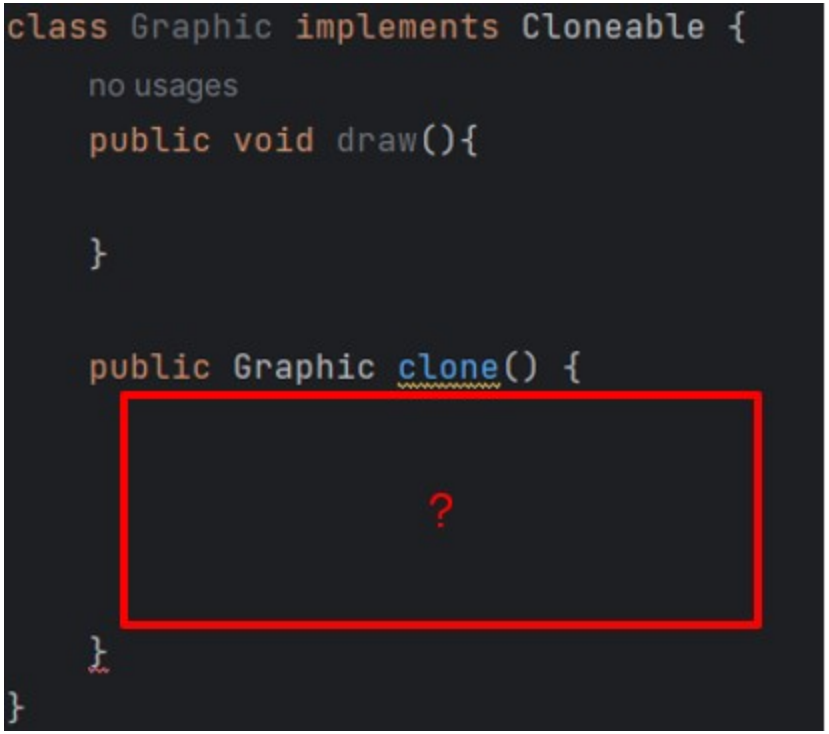
```java
public class Product {
    1 usage
    private String name;
    1 usage
    private int price;
    1 usage
    private String color;

    1 usage
    public Product(String name, int price, String color) {
        this.name = name;
        this.price = price;
        this.color = color;
    }
}
```

```java
public class ProductBuilder {
    2 usages
    private String name;
    2 usages
    private int price;
    2 usages
    private String color;
    1 usage
    public ProductBuilder setName(String name) {
        this.name = name;
        return this;
    }
    1 usage
    public ProductBuilder setPrice(int price) {
        this.price = price;
        return this;
    }
    1 usage
    public ProductBuilder setColor(String color) {
        this.color = color;
        return this;
    }

    1 usage
    public Product build() {
        return new Product(name,price,color);
    }
}
```

✕ Дописати метод clone() для клонування об'єкта ···/3



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! ### @#@#@nedRd8K7dOFidYrv