

소프트웨어공학

문 1. 다음 설명에 해당하는 소프트웨어 개발 프로세스 방법은?

- 설계가 변경되어도 이를 잘 수용할 수 있도록 짧게 반복하면서 소프트웨어를 개발하는 방법
- 반복적이고 진화적인 프로세스와 유사하지만, 경험의 축적으로 나온 모델
- 소프트웨어 개발이 인력 집약적이라는 데 관심을 두고 사람과 팀 구성에 초점을 맞춤

- ① RAD 모델(Rapid Application Development Model)
- ② 반복/점증 모델(Iterative/Incremental Model)
- ③ 나선형 모델(Spiral Model)
- ④ 애자일 프로세스 모델(Agile Process Model)

문 2. 클래스(Class) 다이어그램에 대한 설명으로 옳지 않은 것은?

- ① 개별 사례들을 보고 공통점을 찾아 인식하는 확장형(extensional)이 있다.
- ② 일반적인 개념의 속성, 동작들을 안에서 파악하여 적용하는 내재형(intentional)이 있다.
- ③ 오퍼레이션이나 처리 과정이 수행되는 동안 일어나는 일들을 단계적으로 표현하고자 할 때 사용하는 다이어그램이다.
- ④ 클래스 다이어그램은 도메인 개념과 속성 및 관계를 나타내는 중요한 모델이다.

문 3. 사용자 인터페이스를 설계할 때 고려 사항이 아닌 것은?

- ① 소프트웨어 내부 모듈 사이의 인터페이스를 정의하거나 외부 시스템 사이의 인터페이스를 정의한다.
- ② 사용자가 어떤 액션을 취하였는지 아니면 오류가 발생했는지 알지 못할 때는 시스템이 적절히 지원하는 것은 중요하다.
- ③ 사용자 인터페이스는 일관성이 있어야 한다.
- ④ 사용자 인터페이스는 사용자로부터의 입력이 최소화되도록 해야 한다.

문 4. 소프트웨어 개발 단계에 따른 테스트의 순서는?

- ㄱ. 통합 테스트
- ㄴ. 인수 테스트
- ㄷ. 단위 테스트
- ㄹ. 시스템 테스트

- ① ㄷ - ㄱ - ㄴ - ㄹ
- ② ㄷ - ㄱ - ㄹ - ㄴ
- ③ ㄷ - ㄴ - ㄱ - ㄹ
- ④ ㄷ - ㄹ - ㄱ - ㄴ

문 5. ISO의 소프트웨어 프로세스 평가를 위한 국제 표준인 SPICE에 대한 설명이다. 이에 해당하는 프로세스 범주는?

시스템과 소프트웨어 제품을 개발하는 모든 프로세스, 즉 요구사항 분석(명세화), 설계, 구현, 테스트 등이 이 범주에 속한다.

- ① 조직 프로세스(Organization Process)
- ② 공학 프로세스(Engineering Process)
- ③ 고객-공급 프로세스(Customer-Supplier Process)
- ④ 지원 프로세스(Support Process)

문 6. 다음은 UML2.0 다이어그램에 대한 내용이다. <보기 1>과 <보기 2>를 바르게 연결한 것은?

<보기 1>

- ㄱ. 시스템의 컴파일 시 계층적 구조를 기술하는 다이어그램이다.
- ㄴ. 시퀀스 다이어그램과 액티비티 다이어그램을 혼합한 다이어그램이다.
- ㄷ. 하나의 클래스 실행 시 내부 구조를 상세하게 표현하고자 할 때 사용하는 복합 구조를 표현하는 다이어그램이다.

<보기 2>

- A. 인터랙션 오버뷰 다이어그램(Interaction Overview Diagram)
- B. 콤포지트 스트럭처 다이어그램(Composite Structure Diagram)
- C. 패키지 다이어그램(Package Diagram)

- | | ㄱ | ㄴ | ㄷ |
|---|---|---|---|
| ① | A | B | C |
| ② | C | A | B |
| ③ | A | C | B |
| ④ | C | B | A |

문 7. 다음 파이썬(Python) 코드에 적용된 디자인 패턴은?

```
from abc import ABCMeta, abstractmethod
class Order(metaclass=ABCMeta):
    @abstractmethod
    def execute(self):
        pass
class BuyStockOrder(Order):
    def __init__(self, stock):
        self.stock = stock
    def execute(self):
        self.stock.buy()
class SellStockOrder(Order):
    def __init__(self, stock):
        self.stock = stock
    def execute(self):
        self.stock.sell()
class StockTrade:
    def buy(self):
        print("You will buy stocks")
    def sell(self):
        print("You will sell stocks")
class Agent:
    def __init__(self):
        self.__orderQueue = []
    def placeOrder(self, order):
        self.__orderQueue.append(order)
        order.execute()
if __name__ == '__main__':
    #Client
    stock = StockTrade()
    buyStock = BuyStockOrder(stock)
    sellStock = SellStockOrder(stock)
    #Invoker
    agent = Agent()
    agent.placeOrder(buyStock)
    agent.placeOrder(sellStock)
```

- ① Interpreter Pattern
- ② Strategy Pattern
- ③ Memento Pattern
- ④ Command Pattern

문 8. 비기능적 요구사항에 해당하는 것은?

- ① 항공편, 탑승객, 예약을 입력하는 방법을 결정해야 한다.
- ② 여행사와 고객이 데이터베이스에 접근할 때 어떤 정보를 얻을 수 있는지 결정해야 한다.
- ③ 자주 탑승하는 고객을 서비스하기 위해 시스템을 확장할 수 있도록 설계해야 한다.
- ④ 요금 계산 방법을 결정해야 한다.

문 9. 형상 관리 요소 및 절차에 대한 설명으로 옳지 않은 것은?

- ① 형상 요소는 새 프로젝트가 시작될 때 파악한다.
- ② 소프트웨어 형상에 대한 정보를 추적하고 유지하기 위해 형상의 상태를 보관 및 관리한다.
- ③ 형상은 대상 시스템을 분석하여 높은 수준의 추상화된 정보를 추출하는 작업이다.
- ④ 시스템 형상의 일관성과 개발자 사이의 협력을 확인하기 위해 형상 항목에 대한 변경을 관리한다.

문 10. 다음 설명에 해당하는 스크럼(Scrum) 관련 활동은?

스프린트가 끝나는 시점이나 일정 주기로 수행한다. 이 활동을 통하여 프로젝트를 진행하는 과정에서 드러난 좋았던 점, 여러 가지 문제나 미진한 점 등을 도출한다. 이 활동을 통해 이미 설정된 프로세스로만 프로젝트를 진행하지 않고 프로세스를 지속적으로 개선하여 변화하는 비즈니스 환경에 보다 능동적으로 적응할 수 있도록 한다.

- ① 스프린트 회고
- ② 스프린트 리뷰
- ③ 일일 스크럼 미팅
- ④ 릴리스 계획

문 11. 객체지향 설계가 갖는 특징으로 옳지 않은 것은?

- ① 객체지향 설계에서 중요한 것은 시스템을 구성하는 객체와 속성, 연산을 정의하는 것이다.
- ② 객체지향 설계는 하나의 커다란 작업을 여러 개의 작은 작업으로 분할하고, 분할된 각각의 소작업을 함수(모듈)로 구현하는 것이다.
- ③ 객체지향 설계에서는 주어진 객체의 특성을 분석하여 공통된 특징을 갖는 슈퍼클래스를 생성하는 추상화 기법을 통해 객체의 설계 비용과 시간을 줄일 수 있다.
- ④ 객체지향 설계에서는 캡슐화를 통해 객체의 세부내용 변경에 의해 발생될 수 있는 오류의 파급을 줄일 수 있다.

문 12. 소프트웨어 설계 시 고려해야 하는 결합도가 낮은 것부터 순서대로 바르게 나열한 것은?

- ① 스템프 결합도 - 자료 결합도 - 제어 결합도 - 공통 결합도 - 내용 결합도
- ② 자료 결합도 - 스템프 결합도 - 제어 결합도 - 내용 결합도 - 공통 결합도
- ③ 스템프 결합도 - 자료 결합도 - 제어 결합도 - 내용 결합도 - 공통 결합도
- ④ 자료 결합도 - 스템프 결합도 - 제어 결합도 - 공통 결합도 - 내용 결합도

문 13. 소프트웨어 개발 단계를 시간의 흐름에 따라 네 개의 범주(도입, 상세, 구축, 이행)로 나누고, 각 범주에는 요구사항 도출부터 설계, 구현, 평가까지의 개발 생명주기가 포함되어 있는 방법론은?

- ① XP(eXtreme Programming)
- ② UP(Unified Process)
- ③ CMM(Capability Maturity Model)
- ④ SPICE(Software Process Improvement and Capability dEtermination)

문 14. 빈칸에 들어갈 내용을 순서대로 바르게 나열한 것은?

하나의 모듈을 테스트할 때 상위나 하위 모듈이 개발이 안 된 경우가 있다. 상위나 하위 모듈이 개발될 때까지 기다릴 수 없어 가상의 상위나 하위 모듈을 만들어 사용한다. 상위 모듈 역할을 하는 가상의 모듈을 () (이)라 하고 그 역할은 테스트할 모듈을 ()하는 것이다. 반대로 하위 역할을 하는 모듈을 () (이)라 하고 인자를 통해 받은 값을 가지고 수행한 후 그 결과를 테스트할 모듈에 넘겨주는 역할을 한다.

- ① 스텝(stub) - 구현 - 드라이버(driver)
- ② 드라이버(driver) - 구현 - 스텝(stub)
- ③ 스텝(stub) - 호출 - 드라이버(driver)
- ④ 드라이버(driver) - 호출 - 스텝(stub)

문 15. 다음 코드 변환에 대한 리팩토링 기법에 맞는 최적의 작업은?

```
Void printOwing(double amount) {
    printBanner();

    System.out.println("name:"+_name);
    System.out.println("amount:"+_amount);
}
```



```
Void printOwing(double amount) {
    printBanner();
    printDetails(amount);
}

Void printDetails(double amount) {
    System.out.println("name:"+_name);
    System.out.println("amount:"+_amount);
}
```

- ① 메소드 이동 작업이다.
- ② 메소드 추출 작업이다.
- ③ 인터페이스 추출 작업이다.
- ④ 템플릿 메소드 형성 작업이다.

문 16. McCall의 소프트웨어 품질 요소에 대한 설명으로 옳지 않은 것은?

- ① 사용성(Usability) - 사용자가 쉽게 이해하고 사용이 용이하며 흥미를 느끼는 정도
- ② 정확성(Correctness) - 기능적인 요구사항 만족과 고객 목적 이행 정도
- ③ 이식성(Portability) - 다른 시스템과 정보 교류의 연동 및 통합 정도
- ④ 신뢰성(Reliability) - 필요한 정확성으로 기능을 수행하는 정도

문 17. 객체지향의 개념에서 다형성(Polymorphism)의 주요 설명으로 옳지 않은 것은?

- ① 속성과 관련된 오퍼레이션을 클래스 안에 묶어서 하나로 취급한다.
- ② 하나의 인터페이스에 메소드를 데이터 타입(Data Type) 및 파라미터 수를 변경하여 재정의가 가능하다.
- ③ 하나의 인터페이스를 일관성 있게 사용 중심에서 제공할 수 있다.
- ④ 오버로딩(Overloading), 오버라이딩(Overriding)을 이용한 재사용성을 높일 수 있다.

문 18. T. McCabe의 순환 복잡도(Cyclomatic Complexity)에 대한 설명으로 옳지 않은 것은?

- ① 사이클로매틱 수는 각 모듈에 대한 제어도(fan-out)를 이용하여 측정한다.
- ② 사이클로매틱 수는 코드 전체에서 독립적인 경로의 수를 선형적으로 측정한다.
- ③ 사이클로매틱 수는 그래프 이론을 기반으로 하여, 코드를 동등한 제어 흐름 그래프로 변환한 다음, 메트릭을 결정하기 위한 그래프의 속성을 이용하여 계산한다.
- ④ 원시코드의 구조적인 복잡성을 알아내는 척도이다.

문 19. <표 1>과 같이 기능항목에 대한 복잡도가 주어지고, 입력은 10개, 출력은 5개, 질의는 8개, 파일은 28개, 응용인터페이스는 4개이며, 복잡도는 단순이다. 개발팀의 생산성은 주당 60 기능점수로 구현된다. <표 2>를 참조하여 프로젝트를 수행하기 위한 노력을 추정할 값은?

<표 1>

	항목	복잡도		
		단순	보통	복잡
1	입력(트랜잭션)	3	4	6
2	출력(화면/양식)	4	5	7
3	질의	2	4	6
4	파일	5	10	15
5	응용 인터페이스	4	7	10

<표 2>

특성	처리복잡도	비고
데이터 통신	5	0: 영향 없음
분산데이터 처리	2	1: 약간의 영향
처리속도	0	2: 어느 정도 영향
트랜잭션 처리	2	3: 보통 정도의 영향
하드웨어 이용	3	4: 중요한 영향
온라인 데이터 입력	0	5: 심각한 영향
온라인 갱신 정도	0	
사용자 편의성	2	
계산의 복잡성	2	
재사용성	2	
설치의 용이함	2	
운영의 용이함	0	
이식성	3	
유지 보수성	2	

- ① 3.30 persons-week ② 3.31 persons-week
- ③ 3.32 persons-week ④ 3.33 persons-week

문 20. 원인-결과 그래프 기법으로 테스트 케이스를 구하고자 한다. 문제에 대한 원인과 결과가 다음과 같을 때, 제한조건을 포함한 그래프는?

- 첫 번째 열이 A 또는 B로 시작하고, 두 번째 열이 #이면 '출입가능'을 출력한다.
- 첫 번째 열이 A 또는 B로 시작하지 않으면 '출입금지'로 출력한다.
- 첫 번째 열이 A 또는 B로 시작하고, 두 번째 열이 #이 아니면 '비밀번호 오류'를 출력한다.
- A는 부장, B는 사원을 의미한다.

원인	결과
1: 첫 번째 열이 A	101: '출입금지' 출력
2: 첫 번째 열이 B	102: '출입가능' 출력
3: 두 번째 열이 #	103: '비밀번호 오류' 출력

