

소프트웨어공학

문 1. 다음 특징을 가지는 소프트웨어 개발 방법론은?

- 리팩토링(refactoring)
- 사용자 스토리(user story)
- 소규모 릴리즈(small release)
- 짝 프로그래밍(pair programming)
- 테스트 우선 개발(test-first development)

- ① COCOMO
- ② 익스트림 프로그래밍(XP)
- ③ 통합 프로세스(UP)
- ④ 폭포수(waterfall) 모델

문 2. 다음 상황에 가장 적합한 소프트웨어 아키텍처 패턴은?

- 데이터 관리 컴포넌트, 인터랙션 제어 컴포넌트, 화면 관리 컴포넌트를 분리
- 하나의 데이터를 다양한 그래프로 보여 줄 수 있는 데스크탑 응용 프로그램 개발
- 데이터베이스 연동 부분과 사용자 웹 화면 제공 부분으로 나눈 웹 기반 응용 프로그램 개발

- ① MVC(Model-View-Controller)
- ② 리포지토리(repository)
- ③ 트랜잭션 처리(transaction processing)
- ④ 파이프 필터(pipe-and-filter)

문 3. LOC(Lines Of Code) 기법에 의해 예측된 프로젝트의 총라인 수가 80,000 LOC이고, 투입 개발자 수는 8명, 개발자 1인당 월평균 생산성이 500 LOC일 경우, 개발에 소요되는 시간은?

- ① 10개월
- ② 16개월
- ③ 20개월
- ④ 40개월

문 4. 사용자 요구분석 명세서의 바람직한 특성으로 옳지 않은 것은?

- ① 서로 모순되는 부분이 없어야 한다.
- ② 고객과 개발자가 동의한 것이어야 한다.
- ③ 고객과 개발자 모두 쉽게 이해할 수 있어야 한다.
- ④ 시스템의 구조와 서버 시스템의 구조를 포함하여 기술해야 한다.

문 5. 아키텍처의 4 + 1 관점에 대한 설명으로 옳지 않은 것은?

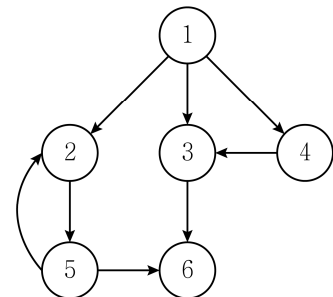
- ① 배치 관점은 시스템을 구성하는 처리 장치 간의 물리적 배치에 초점을 둔다.
- ② 프로세스 관점은 시스템에서 런타임에 상호 작용하는 프로세스들이 어떻게 구성되는지 보여 준다.
- ③ 구현 관점은 물리적 시스템에서 사용하는 소프트웨어 서버 시스템의 모듈이 어떻게 구조화되어 있는가에 관심을 둔다.
- ④ 유스케이스 관점은 시스템의 기능을 제공하기 위해 필요한 클래스나 컴포넌트의 종류, 그리고 이들의 관계에 초점을 둔다.

문 6. 다음 자바 코드에 나타난 것과 같이 동일한 이름의 메시지로 다른 구현을 호출할 수 있는 객체 지향 개념은?

```
Animal a;
a = new Dog( );
a.makeSound( ); // "멍멍" 출력함
a = new Cat( );
a.makeSound( ); // "야옹" 출력함
```

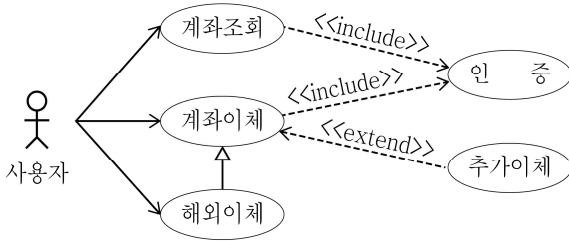
- ① 다형성
- ② 정보 은닉
- ③ 정적 바인딩
- ④ 캡슐화

문 7. 다음 제어흐름 그래프에 대한 순환 복잡도(cyclomatic complexity)는?



- ① 2
- ② 3
- ③ 4
- ④ 5

문 8. 다음 유스케이스 다이어그램에 대한 설명으로 옳지 않은 것은?



- ① ‘계좌이체’는 ‘해외이체’보다 일반화된 유스케이스에 해당한다.
- ② ‘계좌이체’ 유스케이스가 수행되면 ‘추가이체’ 유스케이스도 수행된다.
- ③ ‘계좌조회’, ‘계좌이체’, ‘해외이체’ 유스케이스는 ‘사용자’ 액터에 의해 수행된다.
- ④ ‘인증’ 유스케이스는 ‘계좌조회’와 ‘계좌이체’ 유스케이스에 포함되어 수행된다.

문 9. 다음 설명에 해당하는 아키텍처는?

네트워크의 어떠한 노드라도 주어진 연산을 수행할 수 있는 비중앙집중적인(decentralized) 아키텍처로, 서버와 클라이언트의 구분이 없다. 별도의 중앙 서버 없이 사용자의 PC 사이에서 파일을 주고받기 위한 파일 공유 네트워크 등이 이에 해당한다.

- ① 마이크로서비스(microservice) 아키텍처
- ② 분산 컴포넌트(distributed component) 아키텍처
- ③ 서비스 지향(service-oriented) 아키텍처
- ④ 피어 투 피어(peer-to-peer) 아키텍처

문 10. 다음 자바 코드에서 오류가 발생하는 문장은?

```

abstract class Shape {
    public void paint() { draw(); }
    abstract public void draw();
}

abstract class Circle extends Shape {
    public int radius;
    public Circle(int radius) {
        this.radius = radius; }
    double getArea() {
        return (3.14 * radius * radius); }
}

public class NamedCircle extends Circle {
    String name;
    public NamedCircle(int radius, String name) {
        super(radius);
        this.name = name;
    }
    public void draw() {
        System.out.println("반지름이 " +
            radius + "인 원을 그린다.");
    }
    public static void main(String[] args) {
        Shape s;
        s = new Shape();
        NamedCircle w = new NamedCircle(5, "Ring");
        System.out.println(w.getArea());
        w.draw();
    }
}
  
```

- ① super(radius);
- ② Shape s;
- ③ s = new Shape();
- ④ System.out.println(w.getArea());

문 11. 코드 인스펙션(code inspection)에 대한 설명으로 옳지 않은 것은?

- ① 품질 보증 활동으로 인식한다.
- ② 동적(dynamic) 테스트의 하나이다.
- ③ 회의 전에 프로그램 코드, 분석, 설계 등의 문서를 배포한다.
- ④ 검사 팀은 관련 전문가로 이루어지며 검사목록(checklist)을 제공한다.

문 12. 다음 설명과 용어를 바르게 연결한 것은?

- (가) 코드만 남아있고 관련 문서가 없을 때, 프로그램을 분석하여 프로그램 구조, 자료 구조, 모듈 사이의 관계 등 상세한 설계 정보를 추출함
- (나) 겉으로 보이는 동작이나 외부 행위를 바꾸지 않고 소프트웨어 내부 구조를 바꾸며 점진적으로 설계를 향상시키는 기법
- (다) 기존 시스템을 이해하여 새로운 기능을 추가하거나 성능을 향상시키는 등의 새로운 형태로 변경하는 것으로, 단순히 설계정보를 끌어내는 것뿐만 아니라 이 정보를 사용하여 보다 나은 시스템을 만드는 것

- | (가) | (나) | (다) |
|-------|--------|-----|
| ① 역공학 | 리팩토링 | 재공학 |
| ② 역공학 | 프로토타이핑 | 재공학 |
| ③ 재공학 | 리팩토링 | 역공학 |
| ④ 재공학 | 프로토타이핑 | 역공학 |

문 13. UML 다이어그램에 대한 설명으로 옳지 않은 것은?

- 상태 다이어그램(state diagram)은 객체의 상태 변화를 나타낸다.
- 클래스 다이어그램(class diagram)은 클래스와 클래스 사이의 관계를 나타내는 정적인 모델이다.
- 컴포넌트 다이어그램(component diagram)은 물리적인 노드와 커뮤니케이션 경로를 각각 육면체와 선으로 표시한다.
- 활동 다이어그램(activity diagram)은 업무의 흐름을 표현하거나 유스케이스의 구체적인 흐름을 나타내기 위해 사용한다.

문 14. 바람직한 설계에 대한 설명으로 옳지 않은 것은?

- 모듈 간의 결합도를 강하게 유지한다.
- 유지관리를 잘할 수 있도록 설계한다.
- 분할과 정복의 원리를 적용하여 복잡도를 감소시킨다.
- 변경하기 쉽고 변경으로 인한 영향도 최소화되도록 한다.

문 15. 요구사항 명세 기법 중 자연어 기반 비정형 명세에 대한 설명으로 옳은 것만을 모두 고르면?

- ㄱ. 표현이 모호할 수 있고, 해석이 다를 수 있다.
- ㄴ. 수학적 증명 기술을 이용하여 완전성을 검증할 수 있다.
- ㄷ. 자연어를 기반으로 작성하기 때문에 쉽게 작성할 수 있다.

- ㄷ
- ㄱ, ㄴ
- ㄱ, ㄷ
- ㄴ, ㄷ

문 16. 다음 설명에 해당하는 코드 악취(code smell)는?

수정할 때마다 관련된 여러 클래스에서 수많은 자잘한 변경을 해야 하는 상황이 발생하여 여러 클래스를 동시에 수정해야 하는 상황

- 과잉 중개 메서드(middle man)
- 기능의 산재(shotgun surgery)
- 메시지 체인(message chains)
- 직무 유기 클래스(lazy class)

문 17. 다음 소스 코드와 테스트 입력을 사용한 테스트 커버리지 설명으로 옳지 않은 것은? (단, 단축연산(short-circuit evaluation)은 수행하지 않는다)

문장 번호	소스 코드	테스트 입력
1	void temp(int x, int y) {	T = {t1: (x=-2, y=3), t2: (x=2, y=0) }
2	if(x < 0 && y >= 0)	
3	printf("%d", y);	
4	if(x >= 0)	
	printf("%d", x);	
	}	

- t2만 사용하면 문장(statement) 커버리지가 75 %이다.
- T를 사용하면 문장 커버리지가 100 %이다.
- T를 사용하면 분기(branch) 커버리지가 100 %이다.
- T를 사용하면 조건(condition) 커버리지가 100 %이다.

문 18. UML 시퀀스 다이어그램에서 사용되지 않는 것은?

- 구획면(swimlane)
- 메시지(message)
- 생명선(lifeline)
- 활성화(activation)

문 19. 다음 테스트 주도 개발(TDD)의 단계를 순서대로 바르게 나열한 것은?

- (가) 무엇을 테스트할 것인지 생각한다.
 (나) 테스트를 통과하도록 소스 코드를 작성한다.
 (다) 테스트 코드를 작성한다.
 (라) 소스 코드와 테스트 코드를 리팩토링한다.
 (마) 구현해야 할 소스 코드가 남아 있으면 위의 단계를 반복한다.

- ① (가) - (나) - (다) - (라) - (마)
 ② (가) - (다) - (나) - (라) - (마)
 ③ (나) - (가) - (다) - (라) - (마)
 ④ (나) - (가) - (라) - (다) - (마)

문 20. 다음 상황에 적합한 디자인 패턴은?

슈팅 모바일 게임을 개발하는데, 이 게임은 사용자가 화면을 터치하면 게임 캐릭터가 사용자가 터치한 곳으로 무기를 발사해 적을 공격하는 기능이 있다. 이는 세부적으로 '사용자 터치 인식 → 터치 한 방향으로 캐릭터를 회전 → 무기를 겨누는 동작 실행 → 무기 발사' 순서로 실행되는 단계를 가지며, 각 단계별 동작은 별도 클래스의 메서드로 구현된다. 따라서 캐릭터 무기 발사를 구현하기 위해서는 클라이언트 코드가 각각의 클래스와 내부 메서드 사용 방법을 알아야 한다. 이 문제를 해결하기 위해 클라이언트 코드에게는 위의 복잡한 메서드 호출 과정을 숨기고, 위 과정을 대신 수행하는 높은 레벨의 인터페이스를 제공하고자 한다.

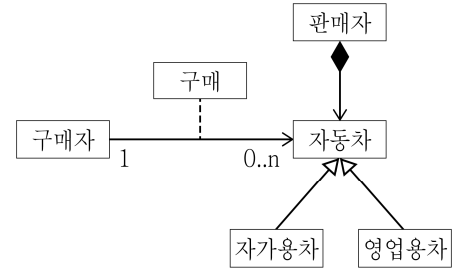
- ① 어댑터 패턴
 ② 추상 팩토리 패턴
 ③ 컴포지트 패턴
 ④ 퍼사드 패턴

문 21. 다음 코드에 대해 기본 경로(basis path) 테스트를 수행할 때, 모든 독립적인 경로가 최소한 한 번은 실행되는 테스트 입력 집합은?

```
int max(int a, int b, int c) {
    int result = a;
    if (b > result)
        result = b;
    if (c > result)
        result = c;
    return result;
}
```

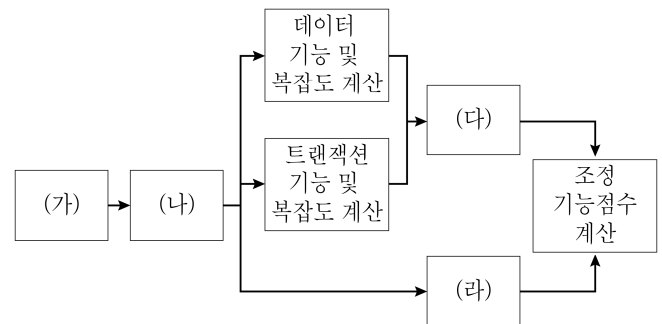
- ① (a=1, b=2, c=3)
 ② (a=1, b=1, c=1), (a=1, b=2, c=1)
 ③ (a=1, b=1, c=1), (a=1, b=1, c=2), (a=1, b=2, c=2)
 ④ (a=1, b=2, c=3), (a=1, b=3, c=2), (a=2, b=3, c=3)

문 22. 다음 클래스 다이어그램에 대한 설명으로 옳은 것은?



- ① '자동차' 클래스는 '구매자' 클래스를 직접 참조한다.
 ② '자가용차', '영업용차' 클래스는 '자동차' 클래스와 실체화 관계를 갖는다.
 ③ '판매자' 객체가 소멸되어도 집약(aggregation)관계인 '자동차' 객체는 소멸되지 않는다.
 ④ '구매' 클래스는 '구매자' 클래스와 '자동차' 클래스의 연관 관계에 속성이나 행위를 추가하기 위해 사용된다.

문 23. 다음은 IFPUG에서 정의한 기능점수 산정 절차이다. (가) ~ (라)에 해당하는 것을 A ~ D에서 바르게 연결한 것은?



- A. 계산 범위 및 애플리케이션 경계 식별
 B. 기능점수 측정 유형 결정
 C. 미조정 기능점수 계산
 D. 조정 인자 계산

- | | (가) | (나) | (다) | (라) |
|---|-----|-----|-----|-----|
| ① | A | B | D | C |
| ② | B | A | C | D |
| ③ | B | A | D | C |
| ④ | D | A | C | B |

문 24. 다음은 ‘갑’ 프로젝트의 계획 단계에서 수립한 CPM(critical path method) 네트워크의 작업 목록표이다. ‘갑’ 프로젝트 완료에 필요한 최소 시간은?

작업	선행작업	소요기간(일)
A	D	15
B	A, D, E	10
C	E	15
D	-	5
E	-	10
F	A, B	20
G	B, C	15

- ① 40일
- ② 45일
- ③ 50일
- ④ 90일

문 25. UML 시퀀스 다이어그램의 결합 프래그먼트(combined fragment)에서 사용하는 인터랙션 오퍼레이터에 대한 설명으로 옳은 것은?

- ① ref - 동시에 실행되는 상호작용을 표현한다.
- ② opt - 다른 다이어그램을 참조할 때 사용한다.
- ③ par - 특정 조건을 만족하는 경우에만 상호작용을 수행한다.
- ④ alt - 여러 상호작용 중 조건을 만족하는 상호작용만 수행하는데, 각 상호작용은 점선으로 분리한다.