소프트웨어공학

문 1. 요구사항 모델링 기법과 가장 관련이 적은 것은?

① UML(Unified Modeling Language)
② SDL(Specification and Design Language)
③ CPM(Critical Path Method)
④ SCR(Software Cost Reduction)

문 2. 요구공학의 공정 순서를 바르게 나열한 것은?

> ㄱ. 요구사항 분석
> ㄴ. 요구사항 검증
> ㄷ. 요구사항 명세
> ㄹ. 요구사항 추출

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

문 3. 다음 설명에 해당하는 소프트웨어 설계 패턴은?

> 일대 다의 객체 의존 관계를 정의하며, 한 객체의 상태가 변화되었을 때, 의존 관계에 있는 다른 객체들에게 자동적으로 변화를 통지한다.

① 옵서버(Observer) 패턴
② 팩토리 메소드(Factory Method) 패턴
③ 데코레이터(Decorator) 패턴
④ 전략(Strategy) 패턴

문 4. 네트워크 프로토콜의 OSI 참조 모델과 가장 관련이 깊은 아키텍처 모델은?

① 계층 모델
② 클라이언트－서버 모델
③ 자료흐름 모델
④ 저장소(repository) 모델

문 5. 다음 설명에 적합한 용어는?

> 원(Circle)의 면적을 구하는 getArea() 함수를 갖고 있는 객체, 직사각형(Rectangle)의 면적을 구하는 getArea() 함수를 갖고 있는 객체, 삼각형(Triangle)의 면적을 구하는 getArea() 함수를 갖고 있는 객체들은 getArea()라는 함수를 가지고 있으므로 면적을 구하기 위해서는 'getArea()'란 메시지를 받으면 수행된다. 그러나 각각의 함수에서 면적을 구하는 방법은 모두 다를 것이다.

① 캡슐화(Encapsulation)
② 상속성(Inheritance)
③ 다형성(Polymorphism)
④ 추상화(Abstraction)

문 6. RUP(Rational Unified Process)에서 프로세스를 나타내는 관점이 아닌 것은?

① 시간에 따라 변하는 모델을 나타내는 동적 관점
② 정해진 대로 행동하는 프로세스 활동을 나타내는 정적 관점
③ 프로세스 동안에 사용되는 실무 관행을 제시하는 실무 관점
④ 프로세스 진행 중 이해당사자(stakeholder)들의 조율을 위한 중재 관점

문 7. LOC/MD는 하루(D) 동안 프로그래머(M)가 작성한 코드 라인수(LOC)로서 생산성을 나타내는 단위이다. 어떤 프로젝트에 20명의 프로그래머가 투입되어 1년 동안 100,000 LOC를 작성했다고 하자. 1년간 실제 일한 날이 250일이라고 할 때, 이 프로젝트의 생산성은?

① 15 LOC/MD
② 20 LOC/MD
③ 25 LOC/MD
④ 40 LOC/MD

문 8. 다음 표는 어떤 프로젝트를 수행하는데 필요한 작업, 작업 수행 기간, 작업들 간의 종속 관계(선후 관계)를 나타낸 것이다. 프로젝트의 전체 일정은 지연되지 않고 반드시 원래의 일정대로 종료되어야 한다면 T8 작업을 수행하는 데 허락되는 최대 지연 시간은?

작업	작업 수행 기간(일)	종속 관계
T1	15	－
T2	10	T1
T3	15	T1, T2
T4	15	－
T5	15	T4
T6	10	T3
T7	10	T6
T8	15	T3, T5
T9	10	T7

① 10일
② 15일
③ 20일
④ 25일

문 9. 기능 점수(Function Point)에서 EI 중의 하나인 회원 삭제 기능의 복잡도를 산정하기 위해 DET 개수를 산출하려고 한다. ILF인 회원 정보는 회원 번호, 회원 이름, 연락처, 주소, 이메일 필드로 구성된다. 회원 삭제는 회원 번호와 기능 키(Ctrl + R)를 선택하여 이루어진다. 만약 삭제하려는 회원 번호가 회원 정보에 존재하지 않는다면, 오류메시지가 출력된다. 산출되는 DET 개수는?

① 3개
② 4개
③ 5개
④ 6개

문 10. 요구사항 명세기법에 대한 설명으로 옳지 않은 것은?

① 시스템의 요구사항 특성을 명세하기 위해 자연어를 기반으로 하는 서술형태 또는 그림 중심으로 명세하는 것을 비정형 명세기법이라 한다.

② 수학적 원리와 표기법을 적용하여 시스템의 요구특성을 보다 정확하고 간결한 상태로 명세하는 것을 정형 명세라 한다.

③ 비정형 명세는 요구사항에 대한 불충분한 명세, 일관성의 결여, 내용의 모호성 여부를 검증하기 어렵다.

④ 안전성, 신뢰성, 보안성과 같은 특성이 매우 중요한 시스템 개발 분야에서 비정형 명세기법의 사용이 증가하고 있다.

문 11. 다음 설명을 UML 클래스 다이어그램으로 표현할 때 가장 적합한 용어는?

> ○ 컴퓨터는 여러 개의 부품으로 구성된다.
> ○ 컴퓨터를 더 이상 사용할 수 없게 되면 그 부품들도 다른 곳에서 재사용할 수 없다.

① Inheritance

② Generalization

③ Composition

④ Dependency

문 12. 애자일(Agile) 프로세스 모델에 해당하지 않는 것은?

① 스크럼(Scrum)

② 크리스탈(Crystal)

③ 특징 주도 개발(FDD)

④ 스파이스(SPICE)

문 13. 웹 서비스(web service)와 관련된 표준이 아닌 것은?

① SOAP

② IDL

③ UDDI

④ WSDL

문 14. 다음은 소프트웨어 개발 과정에서 나타나는 위험이다. 이 중에서 나머지와 다른 유형에 해당하는 위험은?

① 시스템 개발이 완료되기 전에 경쟁 제품이 출시된다.

② 경험 있는 스태프가 프로젝트가 완수되기 전에 떠나간다.

③ 프로젝트에 반드시 필요한 하드웨어가 제 때에 납품되지 않는다.

④ 관리 조직의 우선순위가 바뀐다.

문 15. 리팩토링의 대상이 되는 코드 스멜(code smell)에 해당하지 않는 것은?

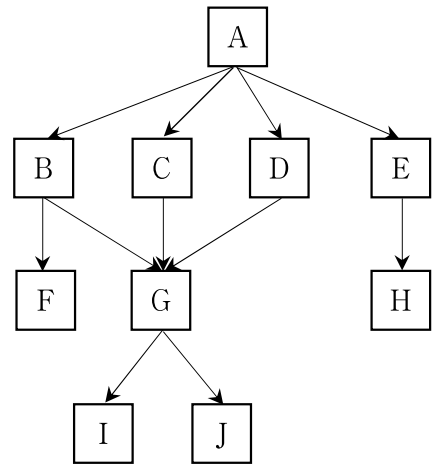
① 읽기 어려운 프로그램

② 중복된 로직을 가진 프로그램

③ 외부에서 보이는 기능을 변경해야 하는 프로그램

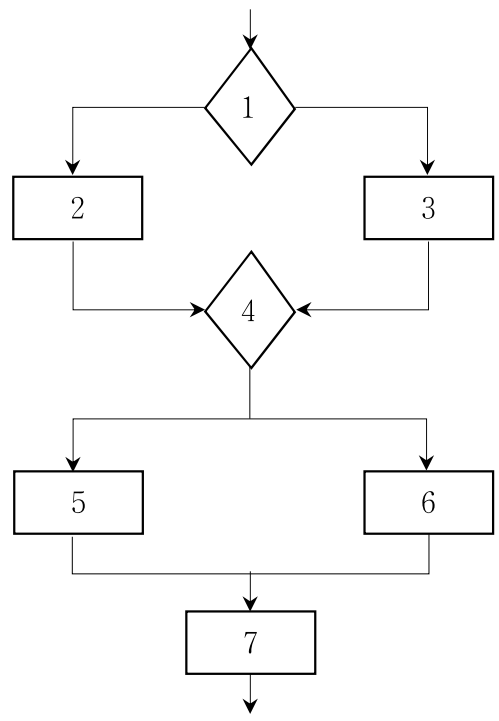
④ 복잡한 조건문이 포함된 프로그램

문 16. 다음은 소프트웨어의 구성요소인 모듈의 계층적 구성을 나타내는 프로그램 구조도이다. 모듈 G에서의 팬인(Fan-In)과 팬아웃(Fan-Out)은?



	팬인	팬아웃
①	1	2
②	2	1
③	2	3
④	3	2

문 17. 다음과 같은 제어 흐름을 갖는 프로그램을 테스팅할 때 사용한 테스트 기준과 그 테스트 기준을 100 % 만족시키기 위한 최소의 테스트 케이스 개수가 바르게 짝지어지지 않은 것은? (단, 제어 흐름도의 모든 경로는 실행가능하다)



① 문장 커버리지(statement coverage) – 1개

② 분기 커버리지(branch coverage) – 2개

③ 경로 테스팅(path testing) – 4개

④ 기본 경로 테스팅(basis path testing) – 3개

문 18. 아래와 같은 send 메소드를 테스팅한다고 가정하자. 이 메소드의 4개의 인자가 가지는 모든 가능한 값을 이용한 테스트의 수는 $3 \times 3 \times 3 \times 3 = 81$이다. 81개의 가능한 테스트 케이스 중에서, 테스트의 개수를 줄이면서도 테스트 도메인 전체에 걸쳐 균일하게 분포하는 테스트 케이스들을 찾아서 테스팅하는 기법은?

```
void send(x1, x2, x3, x4)
- 인자 x1의 가능한 값 : 1, 2, 3
- 인자 x2의 가능한 값 : 1, 2, 3
- 인자 x3의 가능한 값 : 1, 2, 3
- 인자 x4의 가능한 값 : 1, 2, 3
```

① 그래프 기반 테스팅
② 직교 배열(orthogonal array) 테스팅
③ 동등 분할(equivalence partitioning) 테스팅
④ 경곗값 분석(boundary value analysis) 테스팅

문 19. 다음 코드에서 McCabe의 순환적 복잡도(cyclomatic complexity)는?

```
int TestMe(int x, int y, int z) {
 if (x >= y) {
   if (x >= z)
     result = x;
   else
     result = z;
 }
 else
     result = y;
}
```

① 1
② 2
③ 3
④ 4

문 20. 객체지향 시스템의 유지보수에 대한 설명으로 옳지 않은 것은?

① 단일 클래스에 대한 변경은 프로그램의 다른 부분에 영향을 주지 않을 수 있다.
② 상속으로 인한 의존 관계는 유지보수를 더욱 용이하게 한다.
③ 객체지향 패러다임을 사용하는 많은 이유 중에 하나는 유지보수성을 증진시키는 데 있다.
④ 정보 은닉은 회귀 결함의 수를 크게 감소시켜준다.