

소프트웨어공학

1. 소프트웨어 품질 평가모델 ISO/IEC 9126의 속성과 그 속성에 대한 설명으로 옳지 않은 것은?

- ① 이식성(portability): 별도의 작동이나 수단 없이 다양한 환경에서 적응될 수 있는 능력
- ② 사용성(usability): 쉽게 이해되고 배울 수 있고 사용될 수 있는 능력
- ③ 기능성(functionality): 사용되는 자원의 양에 따라 적절한 성능을 제공할 수 있는 능력
- ④ 신뢰성(reliability): 소프트웨어가 주어진 환경에서 정해진 수준의 성능을 유지할 수 있는 능력

2. 다음은 K 공공기관의 새로운 대국민 헬스케어 플랫폼 구축 프로젝트에 대한 요구사항 명세서의 일부이다. 이 프로젝트를 수행하기 위해 가장 적합한 소프트웨어 개발 프로세스 모델은?

- 도입하려는 AI 진단 알고리즘이 시장에 처음 적용되는 신기술이므로 초기 단계에서 시스템의 정확한 요구사항을 완벽히 정의하기 어렵다.
- 만약 프로젝트가 실패하거나 오류가 발생할 경우, 대규모 예산 낭비뿐만 아니라 국민 건강과 직결되는 심각한 위험(risk)이 도사리고 있다.
- 따라서 단번에 시스템을 완성하기보다는 주기마다 [목표 결정 → 위험 분석 → 개발 및 평가 → 다음 단계 계획]의 4가지 활동을 반복하며 점진적으로 완성도를 높여 가야 한다.

- ① 폭포수(waterfall) 모델
- ② 나선형(spiral) 모델
- ③ 컴포넌트 기반 개발(CBD) 모델
- ④ V(verification) 모델

3. 요구분석에 대한 설명으로 옳지 않은 것은?

- ① 요구분석 단계에서는 시스템의 기능이 무엇(what)인지에 초점을 두고 정리한다.
- ② 요구사항은 기능 요구사항과 비기능 요구사항으로 구분할 수 있으며, 기능 요구사항은 품질을 포함한다.
- ③ 신뢰성(reliability)은 소프트웨어를 믿고 사용할 수 있는 것을 의미하며, 정확성(correctness)은 사용자가 요구하는 기능을 충족하는 정도 또는 요구분석 명세서와 일치하는 정도를 의미한다.
- ④ 가용성은 소프트웨어의 총 운용시간 중 고장 없이 정상적으로 가동된 시간의 비율(%)로 나타낸 것으로 “[MTTF ÷ (MTTF + MTTR)] × 100”으로 계산한다.

4. 다음에서 설명하고 있는 프로젝트 관리 방법론의 주요 특징으로 옳지 않은 것은?

이 방법론은 프로젝트를 작은 단위의 작업과 단계로 나누어 신속한 피드백과 반복 작업을 통해 진행하는 관리 방식으로, 변화하는 요구사항에 유연하게 대응하며, 팀원 간의 협력과 소통을 중시한다. 기간이 고정된 스프린트 단위로 업무를 진행하고, 지속적인 검토를 통해 개선된 가치를 제공하며, 유연성과 팀워크를 우선으로 하여 빠른 변화에 적응할 수 있도록 설계되었다.

- ① 모든 설계 결정을 초기에 고정하고 변경을 최소화한다.
- ② 변화 요구에 대해 수용적이며 고객과의 지속적 협업을 강조한다.
- ③ 반복적 개발을 통해 기능을 점진적으로 추가하며 배포한다.
- ④ 문서화 중심이 아닌 실행 가능한 소프트웨어 중심의 개발 접근 방식이다.

5. 객체지향 분석 및 설계 과정에서 시스템의 모델링을 위해 다양한 UML 다이어그램이 활용된다. 다음 도서 관리 시스템 개발 시나리오에 적용할 가장 적절한 UML 다이어그램을 바르게 연결한 것은?

- (가) 사용자가 도서 대출을 요청할 때, 회원 객체, 대출 관리 객체, 도서 객체 간에 발생하는 메시지 호출과 응답을 시간의 흐름에 따라 도식화하여 객체 간의 상호작용을 분석한다.
- (나) 단일 도서 객체가 시스템에 등록되어 대출 가능, 대출 중, 예약됨, 파손 및 폐기에 이르기까지 발생하는 외부 이벤트에 의해 객체 내부가 어떻게 전이되는지를 모델링한다.
- (다) 사용자가 도서를 반납할 때 연체료 계산 처리와 반납 완료 안내 이메일 발송 처리가 동시에 병렬로 실행되고, 두 작업이 모두 완료된 후 다음 단계로 합쳐지는 전체적인 비즈니스 업무 흐름을 시각적으로 도식화한다.

- A. 순차 다이어그램(sequence diagram)
- B. 유스케이스 다이어그램(use case diagram)
- C. 활동 다이어그램(activity diagram)
- D. 상태 다이어그램(state diagram)

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

6. 다음 요구사항 명세서에 대한 설명 중 옳지 않은 것은?

- 요구사항 ID: REQ-0001
- 기능: 사용자 로그인
- 설명: 사용자는 로그인 단계에서 아이디와 비밀번호를 입력하여 로그인한다.
- 제약조건: 비밀번호는 반드시 8자 이상이어야 하며, 대문자, 소문자와 숫자를 포함해야 한다.

- ① 요구사항 ID: REQ-0001는 각 요구사항을 구분하는 유일한 식별자이다.
- ② 기능은 요구사항의 핵심 기능을 서술한다.
- ③ 설명은 사용자 로그인 기능의 구현 방법을 상세히 기술한다.
- ④ 제약조건은 요구사항의 조건이나 제한사항을 명확히한다.

7. 다음 Java로 작성된 코드의 기본경로 테스트를 위한 테스트 케이스 수는?

```
void test(int x) {  
    int y;  
    String str;  
    if(x >= 90){  
        y = 100;  
        str = "Pass";  
    }  
    else{  
        str = "Not Yet";  
        if(x >= 80){  
            y = 80;  
            str = str + ": Try Again";  
        }  
        else  
            y = 50;  
    }  
}
```

- ① 1
- ② 2
- ③ 3
- ④ 4

8. 아래와 같은 C 프로그램이 있을 때, 아래 프로그램의 함수 write_i와 read_i 간 결합(coupling) 수준으로 옳은 것은?

```
int global_i;  
void write_i (int x) {  
    global_i = x;  
}  
int read_i () {  
    return global_i;  
}  
int main() {  
    int ret;  
    write_i(4);  
    ret = read_i();  
    return 0;  
}
```

- ① 데이터 결합(data coupling)
- ② 스탬프 결합(stamp coupling)
- ③ 내용 결합(content coupling)
- ④ 공통 결합(common coupling)

9. 다음에 제시된 소프트웨어 유지보수 종류 (가) ~ (라)와 설명 A ~ D를 바르게 연결한 것은?

- (가) 수정 유지보수(corrective maintenance)
- (나) 적응 유지보수(adaptive maintenance)
- (다) 완전 유지보수(perfective maintenance)
- (라) 예방 유지보수(preventive maintenance)

- A. 시스템이 정상 작동하더라도 더 나은 품질이나 성능을 위해 이루어지는 유지보수
- B. 기존 시스템을 새로운 자료나 운영체제, 하드웨어 환경으로 이식하기 위한 유지보수
- C. 발견된 오류나 결함의 원인을 찾아 문제를 해결하는 유지보수
- D. 장치 발생할 수 있는 오류나 성능 저하에 대비하여 사전에 방지하기 위한 유지보수

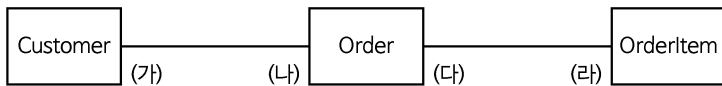
- | | (가) | (나) | (다) | (라) |
|---|-----|-----|-----|-----|
| ① | A | D | B | C |
| ② | A | D | C | B |
| ③ | C | A | B | D |
| ④ | C | B | A | D |

10. 다음은 비기능 요구사항을 측정 가능한 형태로 명세한 것이다. 각 요구사항이 나타내는 시스템의 비기능적 속성을 순서대로 바르게 나열한 것은?

(가) 사용자 요청에 대한 응답시간은 2초 이내여야 한다.
 (나) 신규 사용자는 2시간 이내의 교육으로 주요 기능을 사용할 수 있어야 한다.
 (다) 시스템의 평균 고장 간격은 1,000시간 이상이어야 한다.
 (라) 장애 발생 후 시스템은 30초 이내에 재시작되어야 한다.
 (마) 시스템은 최소 3개의 대상 운영환경에서 실행될 수 있어야 한다.

	(가)	(나)	(다)	(라)	(마)
① 속도	사용성	신뢰성	견고성	이식성	
② 신뢰성	사용성	속도	견고성	이식성	
③ 속도	신뢰성	사용성	이식성	견고성	
④ 속도	사용성	견고성	신뢰성	이식성	

11. 다음은 온라인 쇼핑몰 시스템의 도메인 모델을 나타낸 클래스 다이어그램의 일부에 대한 설명이다. 주어진 비즈니스 규칙을 바탕으로 도출한 다중성(multiplicity) 표기로 옳은 것은?

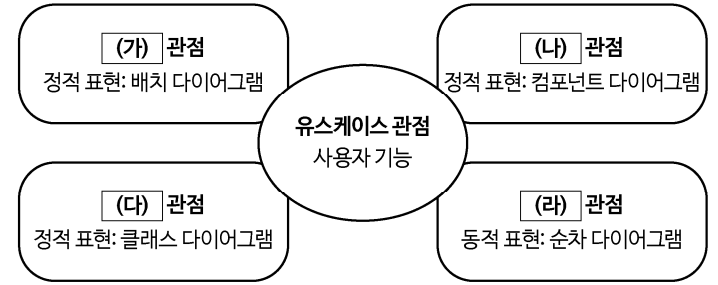


〈비즈니스 규칙〉

- 고객(Customer)과 주문(Order) 클래스는 (가):(나) 관계이다. 하나의 고객은 여러 번 주문할 수 있으며, 아직 단 한 번도 주문하지 않은 신규 가입 고객도 시스템에 존재할 수 있다. 특정 주문은 오직 하나의 고객에게만 종속된다.
- 주문(Order)과 주문품목(OrderItem) 클래스는 (다):(라) 관계이다. 하나의 주문에는 반드시 하나 이상의 주문품목이 포함되어야 하며, 특정 주문품목은 오직 하나의 주문 번호에만 종속된다.

	(가)	(나)	(다)	(라)
① 1	1..*	1	1	
② 1	0..*	1	1..*	
③ 1..*	1	0..*	1..*	
④ 0..*	1	1..*	1	

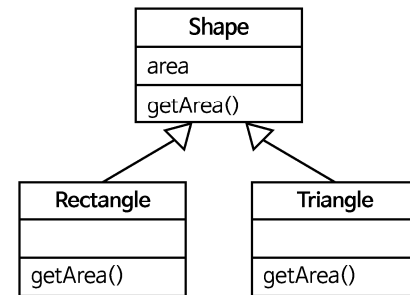
12. 다음은 소프트웨어 아키텍처의 UML 4 + 1 관점에 대한 그림과 설명이다. 빈칸 (가) ~ (라)에 들어갈 관점에 대해 바르게 연결한 것은?



- (가) 시스템을 구성하는 처리 장치 간 물리적 배치에 초점을 둔다.
 (나) 소프트웨어 서브 시스템의 모듈 간 연관 관계와 이들이 어떻게 연결되는지에 초점을 둔다.
 (다) 시스템의 기능을 제공하기 위해 필요한 클래스, 컴포넌트 및 이들의 관계에 초점을 둔다.
 (라) 독자적인 제어 스레드를 가질 수 있는 클래스에 집중하며, 시스템의 동시성과 동기화에 초점을 둔다.

	(가)	(나)	(다)	(라)
① 배치	구현	논리적	프로세스	
② 구현	프로세스	논리적	배치	
③ 배치	논리적	프로세스	구현	
④ 배치	프로세스	논리적	구현	

13. 다음 클래스 다이어그램은 Shape 클래스와 그 하위 클래스 Rectangle, Triangle의 관계를 나타낸 것이다. 아래 코드 실행 결과를 올바르게 예측한 것은?



```

Shape p1 = new Rectangle();
Shape p2 = new Triangle();

p1.getArea();
p2.getArea();
  
```

- ① getArea()는 하위 클래스에만 선언되어 있기 때문에 컴파일 오류가 발생한다.
 ② p1.getArea()와 p2.getArea() 모두 Shape 클래스의 getArea()가 실행된다.
 ③ p1.getArea()는 Rectangle의 메서드가, p2.getArea()는 Triangle의 메서드가 실행된다.
 ④ Rectangle이나 Triangle은 Shape의 메서드를 상속받기 때문에 각각의 getArea()는 실행되지 않는다.

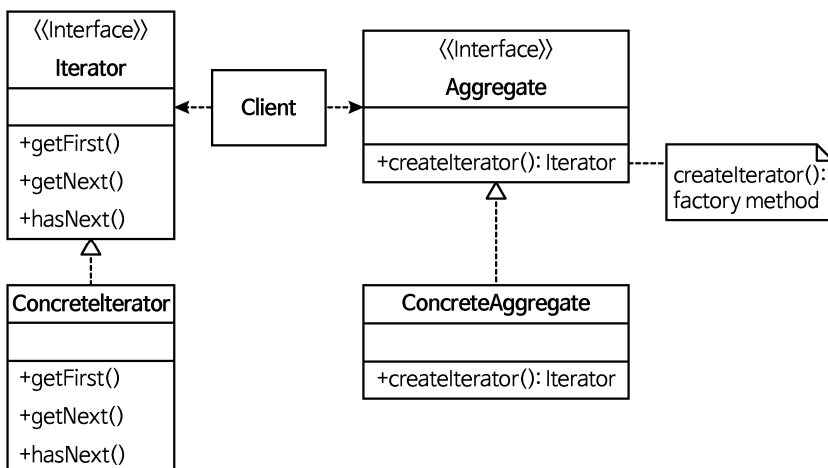
14. 소프트웨어 개발에서 모델링에 대한 설명으로 옳지 않은 것은?

- ① 자연어를 사용하여 모델링하는 경우 일반 사용자가 이해하기 쉬운 장점이 있으나, 해석이 달라질 수 있는 문제가 있다.
- ② 자연어로 작성하기에 복잡한 요구사항들을 모델링할 때, 형식 언어(formal language)를 사용하면 보다 간결하게 표현할 수 있다.
- ③ 데이터 흐름도(DFD, Data Flow Diagram) 표기법은 소프트웨어를 모델링하기 위한 구조적 방법이다.
- ④ E-R 다이어그램 표기법은 소프트웨어를 모델링하기 위한 객체지향 방법이다.

15. 화이트박스 테스트 검증 기준에 대한 설명 중 옳지 않은 것은?

- ① 기본경로 테스트(basic path test)는 프로그램의 실행 흐름을 기준으로 원시 코드의 독립적인 경로가 최소 한 번은 실행되도록 테스트 케이스를 설계한다.
- ② 다중조건 커버리지(multiple condition coverage)는 프로그램 내의 결정문을 결정하는 조건의 참/거짓의 모든 조합이 실행될 수 있도록 테스트 케이스를 설계한다.
- ③ 조건 커버리지(condition coverage)는 프로그램 내의 결정문을 결정하는 조건의 참/거짓이 최소한 한 번은 실행될 수 있도록 테스트 케이스를 설계한다.
- ④ 조건/결정 커버리지(condition/decision coverage)는 분기를 결정하는 각 개별 조건식이 다른 개별 조건식에 영향을 받지 않고 전체 조건식의 결과에 독립적으로 영향을 주도록 테스트 케이스를 설계한다.

16. 다음은 GoF Iterator 설계 패턴의 클래스 다이어그램이다. 해당 다이어그램에 대한 설명 중에서 옳지 않은 것은?



- ① 복합 객체 요소의 내부 구조를 외부에 노출하지 않고 순차적으로 접근하기 위한 설계 패턴이다.
- ② Java 언어의 Vector와 같이 불규칙한 크기의 객체 집합인 자료구조에는 적용할 수 없다.
- ③ Aggregate 인터페이스는 Client가 Iterator를 생성할 수 있도록 한다.
- ④ ConcreteAggregate는 적합한 ConcreteIterator의 인스턴스를 반환하기 위한 인스턴스 생성 인터페이스를 구현한다.

17. 다음 Java 코드에 적용된 디자인 패턴과 Student 클래스와 Member Generator 클래스 간 관계를 옳게 짝 지은 것은?

```

interface Member{ void printType(); }

class Student implements Member {
    public void printType()
    { System.out.println("Student"); }
}

class Staff implements Member {
    public void printType()
    { System.out.println("Staff"); }
}

class Professor implements Member {
    public void printType()
    { System.out.println("Professor"); }
}

class MemberGenerator {
    public Member newMember(String memberType){
        if(memberType.equals("Student"))
            return new Student();
        else if(memberType.equals("Staff"))
            return new Staff();
        else if(memberType.equals("Professor"))
            return new Professor();
        return null;
    }
}
  
```

- ① singleton 패턴 - 연관 관계
- ② singleton 패턴 - 의존 관계
- ③ factory method 패턴 - 연관 관계
- ④ factory method 패턴 - 의존 관계

18. 중환자실 환자 모니터링 시스템은 생체신호 이상에 대한 신속한 경보, 환자 정보 보호, 경보 기능의 안전성, 24시간 서비스 가용성, 의료기기 변경에 대한 유지보수성이 중요하다. 이러한 요구사항을 고려한 아키텍처 설계 방안으로 옳은 것은?

- ① 신속한 처리가 필요한 경보 연산을 다수의 컴포넌트에 분산하여 상호 통신하도록 하고, 안전 관련 기능은 소수의 컴포넌트에 모아 배치한다.
- ② 신속한 경보에 필요한 핵심 연산은 소수의 컴포넌트에 집중하고, 변경이 잦은 의료기기 연동 기능은 작고 독립적인 컴포넌트로 구성한다.
- ③ 안전 관련 기능은 가능한 여러 컴포넌트에 분산하고, 중요 환자 정보는 보호 수준이 높은 내부 계층에 배치한다.
- ④ 높은 가용성을 위해 컴포넌트의 중복을 최소화하고, 유지보수성을 높이기 위해 여러 컴포넌트가 하나의 공유 데이터 구조를 사용하도록 한다.

19. 최근 소프트웨어 유지보수 및 운영 환경에서 필수적으로 적용되는 지속적 통합(CI), 지속적 배포(CD) 파이프라인과 데브옵스(DevOps) 환경에 대한 설명으로 옳지 않은 것은?

- ① 지속적 통합은 다수의 개발자가 작성한 소스 코드를 통합하여 소프트웨어 제품 형상을 개발하고 검증하는 프로세스를 말한다.
- ② 지속적 배포 단계에서 릴리스는 서버에 자원을 할당하고 인프라를 구축하여 최종 사용자에게 서비스를 오픈하는 단계를 의미하며, 설치는 배포 가능한 소프트웨어 패키지를 작성하는 단계를 의미한다.
- ③ 데브옵스 환경에서는 코드의 품질을 유지하고 최상의 품질을 배포할 수 있도록 자동화 테스트 도구를 활용한다.
- ④ 데브옵스에서는 소프트웨어가 정확히 동작하고 있는지 지속적으로 모니터링해야 하며, 모니터링 과정에서 발견된 문제에 대해 신속하게 대응한다.

20. 다음 Java 코드의 조건 커버리지(condition coverage)를 가장 크게 만족하는 테스트 케이스는? (단, 단축 연산(short-circuit evaluation)은 수행하지 않는다)

```
void method(int x, int y, int z) {  
    if (x > 20 && y == 5) {  
        x = 10;  
        z = 5;  
    }  
    if (x == 10 || z % 2 == 0) {  
        z += 10;  
    }  
}
```

- ① (x=10, y=10, z=4), (x=5, y=5, z=3)
- ② (x=25, y=5, z=3), (x=8, y=5, z=3)
- ③ (x=10, y=10, z=4)
- ④ (x=25, y=5, z=3), (x=10, y=5, z=3), (x=5, y=5, z=1)

21. 다음 제시된 Java 코드에 포함된 코드 스멜과 이를 해결하기 위한 리팩토링 기법에 대한 설명으로 옳지 않은 것은?

```
public class Employee {  
    public static final int EMP1 = 0;  
    public static final int EMP2 = 1;  
    public static final int EMP3 = 2;  
    private int empType;  
  
    public Employee(int empType) {  
        this.empType = empType;  
    }  
  
    public int paySalary() {  
        switch (empType) {  
            case EMP1:  
                return 5000;  
            case EMP2:  
                return 6000;  
            case EMP3:  
                return 9000;  
            default:  
                throw new IllegalArgumentException(  
                    "Invalid employee type");  
        }  
    }  
}
```

- ① 새로운 조건이 생길 때마다 switch 문을 수정해야 하므로 버그 발생 가능성이 증가한다.
- ② 코드 스멜 해결을 위해 empType을 서브클래스로 분리하고, paySalary() 메서드를 다형성으로 구현하는 리팩토링 기법을 사용한다.
- ③ 주어진 코드는 객체지향 설계원리 SOLID의 OCP(개방-폐쇄 원칙)를 위반한다.
- ④ 주어진 코드에는 높은 결합도로 인해 변경에 취약한 유사 데이터들의 그룹 중복(data clumps) 코드 스멜이 포함된다.

22. 「SW사업 대가산정 가이드(2025년)」 및 「소프트웨어사업 계약 및 관리감독에 관한 지침(2023년)」에 따라서 지능형 냉난방 조절 시스템 개발에 대한 평균 노력을 추정하고자 한다. 해당 시스템의 기능별 개수 및 평균 복잡도와 사업 규모별 1인 생산성이 <표 1> 및 <표 2>와 같이 주어졌을 때 평균 노력(MM)을 계산한 결과로 옳은 것은? (단, 필요한 보정 계수는 1.5로 계산한다)

<표 1> 기능별 개수 및 평균 복잡도

기능 유형	개수	평균 복잡도
외부입력	15	4.0
외부출력	5	5.2
외부조회	12	3.9
내부 논리 파일	4	7.5
외부 인터페이스 파일	2	5.4

<표 2> 사업 규모별 1인 생산성

사업규모(FP)	1인 생산성(FP/MM)
1,000 미만	19
1,000 이상 ~ 2,000 미만	22
2,000 이상 ~ 3,000 미만	24
3,000 이상	22

- ① 약 0.25 MM
- ② 약 9.14 MM
- ③ 약 13.71 MM
- ④ 약 17.27 MM

23. CMMI에 대한 설명 중 옳지 않은 것은?

- ① 조직의 프로세스를 개선하기 위한 프레임워크로 소프트웨어 개발, 시스템 엔지니어링, 서비스 관리 등 다양한 도메인에서 활용될 수 있다.
- ② CMMI 수준 평가는 성숙도 수준(maturity level)과 역량 수준(capability level)의 두 가지 수준을 평가하는 방법으로 나뉜다.
- ③ 정의(defined) 수준에는 조직을 위한 표준 프로세스는 없고, 프로젝트별 프로세스만 있다.
- ④ 협력 업체 관리는 성숙도 수준에서 관리(managed) 수준에 속하는 프로세스 영역이다.

24. 소프트웨어 품질 평가모델에 대한 설명으로 옳지 않은 것은?

- ① ISO/IEC 9126은 내부품질/외부품질 및 사용품질로 구성되어 있으며 내부품질/외부품질 특성으로는 기능성, 사용성, 효율성, 신뢰성, 이식성, 유지보수성이 있다.
- ② ISO/IEC 25000 시리즈는 ISO/IEC 9126과 ISO/IEC 14598 시리즈를 통합한 프레임워크로 품질 모델 영역, 품질 측정 영역 등 4개 영역으로 구성된다.
- ③ ISO/IEC 14598은 소프트웨어의 개발 과정 또는 완성된 제품에 대해 공급자와 구매자가 객관적으로 품질을 평가할 수 있도록 절차와 방법을 정의한 국제표준으로 평가지원 부분과 평가 프로세스로 구성한다.
- ④ ISO/IEC 12119는 테스트의 절차와 결과의 일관성을 확보하기 위한 지침을 제공하며 패키지 소프트웨어의 품질 요구사항을 정의한다.

25. 어느 대규모 소프트웨어 개발 프로젝트의 작업 항목, 소요 기간, 선행 작업 관계가 다음 표와 같다. 이 프로젝트의 일정 계획 및 임계 경로(critical path) 분석에 대한 설명으로 옳지 않은 것은? (단, 모든 작업은 종료 후 시작(FS, Finish to Start) 관계로 연결되어 있다)

작업	소요 기간	선행 작업
A	4일	없음
B	3일	A
C	6일	A
D	4일	A
E	6일	B
F	4일	C, D
G	5일	E, F
H	3일	D
I	4일	G, H

- ① 프로젝트의 임계 경로는 A→C→F→G→I이며, 프로젝트의 최소 완료 기간은 23일이다.
- ② 만약 작업 B가 2일 지연된다면 전체 프로젝트의 최소 완료 기간은 24일로 늘어나고 임계 경로가 변경된다.
- ③ 작업 D가 3일 지연되더라도 전체 프로젝트의 최소 완료 기간은 지연되지 않는다.
- ④ 작업 H는 이 프로젝트의 모든 작업 중 가장 많은 8일의 여유 시간을 확보하고 있다.