

소프트웨어공학

문 1. 소프트웨어 요구분석 명세서(SRS)에 포함되는 내용이 아닌 것은?

- ① 개발 시스템의 목적 및 범위
- ② 아키텍처 및 인터페이스 명세
- ③ 자료흐름도, 자료사전, 소단위명세서
- ④ 기능적 요구사항 및 비기능적 요구사항

문 2. 프로토타입 개발 모델에 대한 설명으로 옳지 않은 것은?

- ① 시스템 기능을 사용자에게 확인시킴으로써 개발자와 사용자간의 견해차이가 해결될 수 있다.
- ② 분석가나 개발자는 프로토타입을 이용하여 불완전하거나 일치하지 않는 요구사항을 발견할 수 있다.
- ③ 완전하지는 못하지만 작동하는 시스템을 만들어 기능성과 유용성을 관리자에게 보여줄 수 있다.
- ④ 고객의 요구사항을 초기에 구체적으로 기술하기 어렵고 중요한 문제점이 프로젝트의 후반부에 가서야 발견된다.

문 3. 코드의 정적 분석 도구를 통하여 찾을 수 있는 오류에 해당하지 않는 것은?

- ① 원시 코드의 구조적 결함
- ② 실행 경로가 없는 Dead Code
- ③ 데이터 정의의 충돌이나 잘못된 데이터 사용
- ④ 특정 모듈의 실행 시간 메모리 사용량 초과

문 4. 리팩토링에 대한 설명으로 옳지 않은 것은?

- ① 리팩토링의 대상은 읽기 어려운 코드, 중복된 로직의 코드, 복잡한 조건문이 있는 코드 등이 대표적인 것이다.
- ② 리팩토링은 실행 중인 프로그램의 기능 변경이 수반되어야 한다.
- ③ 리팩토링을 통해서 프로그램의 이해가 쉬워진다.
- ④ 리팩토링은 결함을 찾는 데 도움을 준다.

문 5. 같은 유형의 소프트웨어 테스트 기법으로만 묶은 것은?

ㄱ. 기본 경로 테스트
 ㄴ. 페어와이즈 테스트
 ㄷ. 모델 기반 테스트
 ㄹ. 분기 커버리지
 ㅁ. 직교 배열 테스트

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

문 6. 소프트웨어 개발 작업에 일관적이고 체계적인 구조(framework)를 제공하기 위하여 1995년에 ISO/IEC에서 제정한 소프트웨어 생명주기 공정 국제표준은?

- ① ISO/IEC 25000
- ② ISO/IEC 12207
- ③ ISO/IEC 12119
- ④ ISO/IEC 9126

문 7. 소프트웨어 아키텍처의 4+1 관점(view)에 대한 설명으로 옳지 않은 것은?

- ① 유스케이스 관점에서는 외부 행위자에 의해 인식되는 시스템의 기능 요구사항을 보여주는 데 초점을 둔다.
- ② 논리 관점에서는 계층 구조, 제약 사항, 코드 재사용 등과 같은 시스템 구현을 위한 요건을 보여주는 데 초점을 둔다.
- ③ 프로세스 관점에서는 독자적인 제어 스레드를 가질 수 있는 액티브 클래스에 초점을 둔다.
- ④ 배치 관점에서는 물리적인 시스템을 구성하고 있는 각 부분들의 분산 형태와 설치에 초점을 둔다.

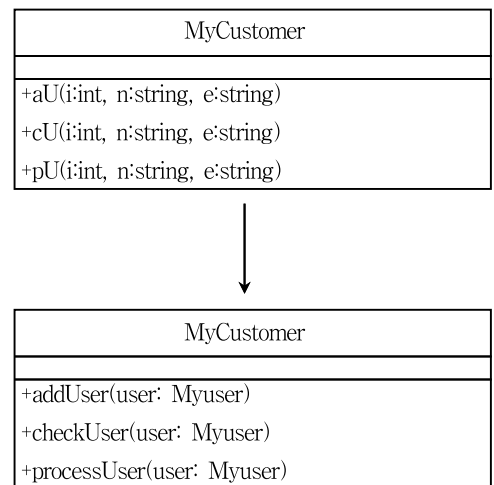
문 8. 테스트 주도 개발(Test-driven development)은 애자일 기법에서 개발되는 증분과 해당 증분을 위한 테스트 코드를 함께 작성해 나가는 방법이다. 이에 대한 설명으로 옳지 않은 것은?

- ① 멀티 스레드를 사용하는 어플리케이션의 테스트를 위해 개발되었다.
- ② JUnit과 같은 자동화된 테스트 프레임워크 환경이 적극적으로 사용된다.
- ③ 개발 프로세스의 초기에 결함이 발견될 수 있다.
- ④ 새로운 코드나 표준 라이브러리의 컴포넌트를 사용하여 기능이 구현되는 신규 소프트웨어 개발에서 가장 가치가 있다.

문 9. CMMI의 5단계 소프트웨어 프로세스 성숙도 모델에서 Level 2에 해당하는 주요 프로세스 영역이 아닌 것은?

- ① 프로젝트 모니터링 및 통제(Project monitoring and control)
- ② 프로세스 및 제품 품질 보증(Process and product quality assurance)
- ③ 요구사항 개발(Requirements development)
- ④ 측정 및 분석(Measurement and analysis)

문 10. 그림과 같이 MyCustomer 클래스를 대상으로 리팩토링을 실시하였다. 이와 관련 있는 리팩토링 기법만을 모두 고른 것은?



ㄱ. 메소드 추출(Extract Method)
 ㄴ. 메소드 상향(Pull Up Method)
 ㄷ. 메소드명 변경(Rename Method)
 ㄹ. 매개변수 세트를 객체로 전환(Introduce Parameter Object)
 ㅁ. 메소드를 매개변수로 전환(Parameterized Method)

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

문 11. 배열, 레코드, 구조체 등을 매개변수로 사용하는 모듈 사이의 결합도는?

- ① 데이터 결합도(Data Coupling)
- ② 스탬프 결합도(Stamp Coupling)
- ③ 제어 결합도(Control Coupling)
- ④ 내용 결합도(Content Coupling)

문 12. 요구분석 단계를 순서대로 바르게 나열한 것은?

- ㄱ. 요구사항 검증
 ㄴ. 요구사항 명세화
 ㄷ. 타당성 조사
 ㄹ. 요구사항 추출 및 분석

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

문 13. 시스템을 구성하는 물리적인 노드와 통신 경로, 그리고 컴포넌트의 수행 환경을 표시하는 UML 다이어그램은?

- ① 컴포지트(composite) 다이어그램
- ② 타이밍(timing) 다이어그램
- ③ 통신(communication) 다이어그램
- ④ 배치(deployment) 다이어그램

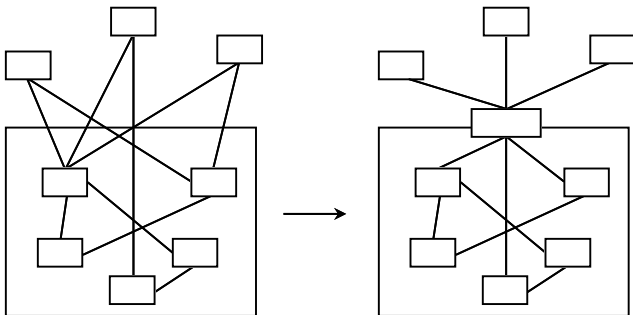
문 14. 프로그램 인스펙션(inspection)을 통해 결함을 검출하고자 할 때, 결함 유형에 따른 검사항목을 옳게 짝지은 것은?

- ① 데이터 결함 - 모든 반복문은 확실히 종료되는가?
- ② 제어 결함 - 버퍼 오버플로우의 가능성이 있는가?
- ③ 인터페이스 결함 - 매개변수들의 순서가 올바른가?
- ④ 메모리 관리 결함 - 예상하지 않은 입력이 문제를 일으킬 수 있는가?

문 15. Iterator 패턴에 대한 설명으로 옳지 않은 것은?

- ① 집합 객체의 요소들에 대해 순서대로 접근하는 방법을 제공한다.
- ② 배열(Array), 배열리스트(ArrayList), 해시 테이블과 같은 객체를 처리하는 데 사용하는 패턴이다.
- ③ 서로 다른 인터페이스를 사용하기 때문에 각각의 객체를 참조하기 위한 다형성 코드(polymorphic code)를 개발하는 것이 불가능하다.
- ④ 서로 다른 집합 객체 구조에 대해 동일한 방법으로 순회할 수 있다.

문 16. 그림과 같이 서브시스템 사이의 의사소통 및 종속성을 최소화하기 위하여 단순화된 하나의 인터페이스를 제공하는 디자인 패턴은?

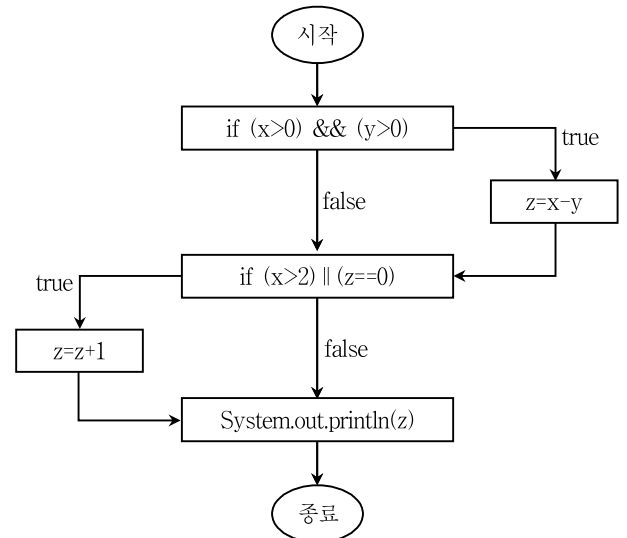


- ① Facade 패턴
- ② Decorator 패턴
- ③ Bridge 패턴
- ④ Adapter 패턴

문 17. 통합 테스트에 대한 설명으로 옳지 않은 것은?

- ① 회귀 테스트는 복잡하고 시간이 중요한 프로젝트에 적용하면 효율적이다.
- ② 점증적인 통합은 빅뱅 방식에 비해 인터페이스를 완전하게 테스트할 가능성이 더 높다.
- ③ 객체지향 시스템의 드라이버는 사용자 인터페이스를 대체하여 사용함으로써 인터페이스의 구현 이전에 기능성 테스트를 수행할 수 있다.
- ④ 객체지향 시스템의 스텝은 하나 이상의 협동 클래스들이 아직 완전히 구현되지 않은 상황에서 사용할 수 있다.

문 18. 다음 제어 흐름 그래프에 나타난 프로그램을 테스트할 때, 옳지 않은 것은?



- ① 분기 커버리지를 만족하는 최소의 테스트 케이스는 2개이다.
- ② 기본 경로의 개수는 3개이다.
- ③ 문장 커버리지를 만족하는 최소의 테스트 케이스는 1개이다.
- ④ {(x:1, y:2, z:0), (x:5, y:0, z:0)}은 분기 커버리지를 만족하지 못한다.

문 19. 오픈 소스 도구와 기능을 연결한 것으로 옳지 않은 것은?

- ① JaCoCo - 자바 소스 보안 취약점 분석 도구
- ② Git - 분산 버전 관리 도구
- ③ Maven - 프로젝트 관리 도구
- ④ Cucumber - 행위 주도 개발(Behavior Driven Development) 지원 도구

문 20. 국제표준과 관련된 내용으로 옳지 않은 것은?

- ① IEEE 830 - 소프트웨어 요구명세서 표준
- ② IEEE 1219 - 소프트웨어 유지보수 표준
- ③ IEC 62304 - 항공기 소프트웨어 안전성 표준
- ④ ISO 26262 - 자동차 기능 안전성 표준