1. 애플리케이션 설계- SEC_01(소프트웨어 아키텍처) 기출 및 예상 문제

기출 및 예상 문제(소프트웨어 아키텍처)

1. 소프트웨어의 상위 설계에 속하지 않는 것은?

① 아키텍처 설계 ② 모듈 설계

③ 인터페이스 정의 ④ 사용자 인터페이스 설계

2. 다음 ( ) 안에 들어갈 내용으로 옳은 것은?

컴포넌트 설계 시 "( )에 의한 설계"를 따를 경우, 해당 명세에서는

(1) 컴포넌트의 오퍼레이션 사용 전에 참이 되어야 할 선행 조건

(2) 사용 후 만족되어야 할 결과 조건

(3) 오퍼레이션이 실행되는 동안 항상 만족되어야 할 불변 조건 등이 포함되어야 한다.

① 협약(Contract) ② 프로토콜(Protocol)

③ 패턴(Pattern) ④ 관계(Relation)

3. 소프트웨어 설계에서 사용되는 대표적인 추상화(Abstraction) 기법이 아닌 것은?

① 자료 추상화 ② 제어 추상화

③ 과정 추상화 ④ 강도 추상화

4. 객체지향 설계에서 정보 은닉(Information Hiding)과 관련한 설명으로 틀린 것은?

① 필요하지 않은 정보는 접근할 수 없도록 하여 한 모듈 또는 하부 시스템이 다른 모듈의 구현에 영향을 받지 않게 설계되는 것을 의미 한다.

② 모듈들 사이의 독립성을 유지시키는 데 도움이 된다.

③ 설계에서 은닉되어야 할 기본 정보로는 IP주소와 같은 물리적 코드, 상세 데이터 구조 등이 있다.

④ 모듈 내부의 자료 구조와 접근 동작들에만 수정을 국한하기 때문에 요구사항 등 변화에 따른 수정이 불가능하다.

1. 애플리케이션 설계- SEC_01(소프트웨어 아키텍처) 기출 및 예상 문제

기출 및 예상 문제(소프트웨어 아키텍처)

5. 소프트웨어 아키텍처 설계에서 시스템 품질 속성이 아닌 것은?

① 가용성(Availability)

② 독립성(Isolation)

③ 변경 용이성(Modifiability)

④ 사용성(Usability)

6. 아키텍처 설계 과정이 올바른 순서로 나열된 것은?

㉮ 설계 목표 설정

㉯ 시스템 타입 결정

㉰ 스타일 적용 및 커스터마이즈

㉱ 서브 시스템의 기능, 인터페이스 동작 작성

㉲ 아키텍처 설계 검토

① 가 -> 나 -> 다 -> 라 -> 마

② 마 -> 가 -> 나 -> 라 -> 다

③ 가 -> 마 -> 나 -> 라 -> 다

④ 가 -> 나 -> 다 -> 마 -> 라

7. 모듈화(Modularity)와 관련한 설명으로 틀린 것은?

① 소프트웨어의 모듈은 프로그래밍 언어에서 Subroutine, Function 등으로 표현될 수 있다.

② 모듈의 수가 증가하면 상대적으로 각 크기가 커지며, 모듈 사이의 상호교류가 감소하여 과부하(Overload) 현상이 나타난다.

③ 모듈화는 시스템을 지능적으로 관리할 수 있도록 해주며, 복잡도 문제를 해결하는 데 도움을 준다.

④ 모듈화는 시스템의 유지 보수와 수정을 용이하게 한다.

8. 소프트웨어 아키텍처 설계에 대한 설명으로 옳지 않은 것은?

① 이해 관계자들의 의사소통 도구로 활용된다.

② 설계된 모듈을 프로그래밍 언어를 통해 구현한다.

③ 애플리케이션을 모듈로 분할하고, 모듈 간 인터페이스를 결정하는 과정이다.

④ 기본 원리에는 모듈화, 추상화, 단계적 분해, 정보은닉이 있다.

1. 애플리케이션 설계- SEC_01(소프트웨어 아키텍처) 기출 및 예상 문제

기출 및 예상 문제(소프트웨어 아키텍처)

9. 소프트웨어 모듈화의 장점이 아닌 것은?

① 오류의 파급 효과를 최소화한다.

② 기능의 분리가 가능하여 인터페이스가 복잡하다.

③ 모듈의 재사용 가능으로 개발과 유지보수가 용이하다.

④ 프로그램의 효율적인 관리가 가능하다.

1. 애플리케이션 설계- SEC_02(아키텍처 패턴) 기출 및 예상 문제

기출 및 예상 문제(아키텍처 패턴)

1. 파이프 필터 형태의 소프트웨어 아키텍처에 대한 설명 으로 옳은 것은?

① 노드와 간선으로 구성된다.

② 서브시스템이 입력 데이터를 받아 처리하고 결과를 다음 서브시스템으로 넘겨주는 과정을 반복한다.

③ 계층 모델이라고도 한다.

④ 3개의 서브시스템(모델, 뷰, 제어)으로 구성되어 있다.

2. 소프트웨어 아키텍처와 관련한 설명으로 틀린 것은?

① 파이프 필터 아키텍처에서 데이터는 파이프를 통해 양방향으로 흐르며, 필터 이동 시 오버헤드가 발생하지 않는다.(단방향, 오버헤드가 데이터 변환 시 발생 가능성)

② 외부에서 인식할 수 있는 특성이 담긴 소프트웨어의 골격이 되는 기본 구조로 볼 수 있다.

③ 데이터 중심 아키텍처는 공유 데이터 저장소를 통해 접근자 간의 통신이 이루어지므로 각 접근자의 수정과 확장이 용이하다.

④ 이해 관계자들의 품질 요구사항을 반영하여 품질 속성을 결정한다.

3. 서브시스템이 입력 데이터를 받아 처리하고 결과를 다른 시스템에 보내는 작업이 반복되는 아키텍처 스타일은?

① 클라이언트 서버 구조 ② 계층 구조

③ MVC 구조 ④ 파이프 필터 구조

4. 분산시스템을 위한 마스터-슬레이브(Master-Slave) 아키텍처에 대한 설명으로 틀린 것은?

① 일반적으로 실시간 시스템에서 사용된다.

② 마스터 프로세스는 일반적으로 연산, 통신, 조정을 책임진다.

③ 슬레이브 프로세스는 데이터 수집 기능을 수행할 수 없다.

④ 마스터 프로세스는 슬레이브 프로세스들을 제어할 수 있다.

1. 애플리케이션 설계- SEC_02(아키텍처 패턴) 기출 및 예상 문제

기출 및 예상 문제(아키텍처 패턴)

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

① Peer-To-Peer Model ② MVC Model

③ Layers Model ④ Client-Server Model

6. 소프트웨어 아키텍처 모델 중 MVC와 관련한 설명으로 틀린 것은?

① MVC 모델은 사용자 인터페이스를 담당하는 계층의 응
집도를 높일 수 있고, 여러 개의 다른 UI를 만들어 그 사이 에 결합도를 낮출 수 있다.

② 모델(Model)은 뷰(View)와 제어(Controller)사이에서
전달자 역할을 하며, 뷰마다 모델 서브시스템이 각각 하나 씩 연결된다.

③ 뷰(View)는 모델(Model)에 있는 데이터를 사용자 인터
페이스에 보이는 역할을 담당한다.

④ 제어(Controller)는 모델(Model)에 명령을 보냄으로써
모델의 상태를 변경할 수 있다.

7. 아키텍처 패턴(Architecture Pattern)에 대한 설명 중 가장 옳지
않은 것은?

① 소프트웨어 초기 설계에서 발생하는 문제들을 해결하기 위한 전형 적인 해결 방식들을 의미한다.

② 검증된 구조로 개발하기 때문에 오류가 적어 개발시간을 단축할 수 있다.

③ 서브시스템들에 대한 역할을 정의하고 있지만, 그들 간의 인터페이스에 대한 지침은 없다.

④ 시스템에 대한 이해가 쉬워지고, 특성을 예측할 수 있게 된다.

8. 다음 중 클라이언트-서버(Client-Server) 모델에 대한 설명으로
가장 거리가 먼 것은?

① 사용자는 클라이언트를 통해서 요청을 전달하며, 서버는 이에 응답하는 방식이다.

② 서버는 클라이언트의 요청에 대비하여 항상 대기 상태를 유지한다.

③ 서버와 클라이언트는 서로 독립적이다.

④ 다수의 서버와 하나의 클라이언트로 구성되는 패턴으로 분산 환경
시스템에 적합하다.

1. 애플리케이션 설계- SEC_02(아키텍처 패턴) 기출 및 예상 문제

기출 및 예상 문제(아키텍처 패턴)

9. 여러 컴포넌트들 중 각 컴포넌트들이 서비스를 제공하는 서버가 될 수도 있고, 서비스를 요청하는 클라이언트도 될 수 있는 패턴으로 전형적인 멀티스레딩을 사용하는 방식의 패턴을 무엇이라 하는가?

① 클라이언트-서버 ② 블랙보드

③ 이벤트-버스 ④ 피어-투-피어

10. 다음 중, 이벤트-버스 패턴(Event-Bus Pattern)의 주요 컴포넌트가 아닌 것은?

① 소스 ② 리스너

③ 채널 ④ 버퍼

1. 애플리케이션 설계-SEC_03(객체 지향(Object-Oriented)) 기출 문제

기출 및 예상 문제(객체 지향(Object-Oriented))

1. 객체에 대한 설명으로 틀린 것은?

① 객체는 상태, 동작, 고유 식별자를 가진 모든 것이라 할 수 있다.

② 객체는 공통 속성을 공유하는 클래스들의 집합이다.

③ 객체는 필요한 자료 구조와 이에 수행되는 함수들을 가진 하나의 독립된 존재이다.

④ 객체의 상태는 속성값에 의해 정의된다.

2. 객체지향개념 중 하나 이상의 유사한 객체들을 묶어 공통된 특성을 표현한 데이터 추상화를 의미하는 것은?

① Method ② Class

③ Field ④ Message

3. 객체지향의 주요 개념에 대한 설명으로 틀린 것은?

① 캡슐화는 상위 클래스에서 속성이나 연산을 전달받아 새로운 형태의 클래스로 확장하여 사용하는 것을 의미한다.

② 객체는 실세계에 존재하거나 생각할 수 있는 것을 말한다.

③ 클래스는 하나 이상의 유사한 객체들을 묶어 공통된 특성을 표현한 것이다.

④ 다형성은 상속받은 여러 개의 하위 객체들이 다른 형태의 특성을 갖는 객체로 이용될 수 있는 성질이다.

4. 객체지향에서 정보은닉과 가장 밀접한 관계가 있는 것은?

① Encapsulation ② Class

③ Method ④ Instance

1. 애플리케이션 설계-SEC_03(객체 지향(Object-Oriented)) 기출 문제

기출 및 예상 문제(객체 지향(Object-Oriented))

5. 객체지향 기법에서 클래스들 사이의 '부분 전체(Part-Whole)' 관계 또는 부분(is-a-part-of)'의 관계로 설명되는 연관성을 나타내는 용어는?

① 일반화 ② 추상화

③ 캡슐화 ④ 집단화

6. 객체에게 어떤 행위를 하도록 지시하는 명령은?

① Class ② Package

③ Object ④ Message

7. 객체지향 기법에서 같은 클래스에 속한 각각의 객체를 의미하는 것은?

① Instance ② Message

③ Method ④ Module

8. 객체지향 개념에서 연관된 데이터와 함수를 함께 묶어 외부와 경계 를 만들고 필요한 인터페이스만을 밖으로 드러내는 과정은?

① 메시지(Message) ② 캡슐화(Encapsulation)

③ 다형성(Polymorphism) ④ 상속(Inheritance)

1. 애플리케이션 설계-SEC_03(객체 지향(Object-Oriented)) 기출 문제

기출 및 예상 문제(객체 지향(Object-Oriented))

9. 객체지향 기법에서 상위 클래스의 메소드와 속성을 하위 클래스가 물려받는 것을 의미하는 것은?

① Abstraction(추상화)　② Polymorphism(다형성)

③ Encapsulation(캡슐화)　④ Inheritance(상속)

10. 객체지향 개념에서 다형성(Polymorphism)과 관련한 설명으로 틀린 것은?

① 다형성은 현재 코드를 변경하지 않고 새로운 클래스를 쉽게 추가할 수 있게 한다.

② 다형성이란 여러 가지 형태를 가지고 있다는 의미로, 여러 형태를 받아들일 수 있는 특징을 말한다.

③ 메소드 오버라이딩(Overriding)은 상위 클래스에서 정의한 일반 메소드의 구현을 하위 클래스에서 무시하고 재정의할 수 있다.

④ 메소드 오버로딩(Overloading)의 경우 매개 변수 타입은 동일하지만 메소드 명을 다르게 함으로써 구현, 구분할 수 있다.

1. 애플리케이션 설계-SEC_04(객체지향 분석 및 설계) 기출 문제

기출 문제(객체지향 분석 및 설계)

1. 객체지향 분석 방법론 중 Coad-Yourdon 방법에 해당하는 것은?

① E-R 다이어그램을 사용하여 객체의 행위를 데이터 모델링 하는데 초점을 둔 방법이다.

② 객체, 동적, 기능 모델로 나누어 수행하는 방법이다.

③ 미시적 개발 프로세스와 거시적 개발 프로세스를 모두 사용하는 방법이다.

④ Use Case를 강조하여 사용하는 방법이다.

2. 그래픽 표기법을 이용하여 소프트웨어 구성 요소를 모델링 하는 럼바우 분석 기법에 포함되지 않는 것은?

① 객체 모델링 ② 기능 모델링

③ 동적 모델링 ④ 블랙박스 분석 모델링

3. 럼바우(Rumbaugh)의 객체지향분석 절차를 가장 바르게 나열한 것은?

① 객체 모형 → 동적 모형 → 기능 모형

② 객체 모형 → 기능 모형 → 동적 모형

③ 기능 모형 → 동적 모형 → 객체 모형

④ 기능 모형 → 객체 모형 → 동적 모형

4. 다음 내용이 설명하는 객체지향 설계 원칙은?

- 클라이언트는 자신이 사용하지 않는 메소드와 의존관계를 맺으면 안 된다.
- 클라이언트가 사용하지 않는 인터페이스 때문에 영향을 받아서는 안 된다.

① 인터페이스분리 원칙 ② 단일 책임 원칙

③ 개방 폐쇄의 원칙 ④ 리스코프 교체의 원칙

1. 애플리케이션 설계-SEC_04(객체지향 분석 및 설계) 기출 문제

기출 문제(객체지향 분석 및 설계)

5. 클래스 설계 원칙에 대한 바른 설명은?

① 단일 책임 원칙 : 하나의 클래스만 변경 가능해야 한다.

② 개방, 폐쇄의 원칙 : 클래스는 확장에 대해 열려 있어야 하며 변경에 대해 닫혀 있어야 한다.

③ 리스코프 교체의 원칙 : 여러 개의 책임을 가진 클래스는 하나의 책임을 가진 클래스로 대체 되어야 한다.

④ 의존관계 역전의 원칙 : 클라이언트는 자신이 사용하는 메소드와 의존관계를 갖지 않도록 해야 한다.

6. 소프트웨어를 개발하기 위한 비즈니스(업무)를 객체와 속성, 클래스와 멤버, 전체와 부분 등으로 나누어서 분석해 내는 기법은?

① 객체지향 분석 ② 구조적 분석

③ 기능적 분석 ④ 실시간 분석

7. 럼바우(Rumbaugh)의 객체지향 분석 기법 중 자료 흐름도(DFD)를 주로 이용하는 것은?

① 기능 모델링 ② 동적 모델링

③ 객체 모델링 ④ 정적 모델링

8. 럼바우(Rumbaugh) 분석 기법에서 정보 모델링이라고도 하며, 시스템에서 요구되는 객체를 찾아내어 속성과 연산 식별 및 객체들 간의 관계를 규정하여 다이어그램을 표시하는 모델링은?

① Object(객체) ② Dynamic(동적인)

③ Function(함수, 기능, 연산) ④ Static(정적인)

1. 애플리케이션 설계-SEC_05(모듈) 기출 문제

기출 문제(모듈)

1. 결합도(Coupling)에 대한 설명으로 틀린 것은?

① 데이터 결합도(Data Coupling)는 두 모듈이 매개변수로 자료를 전달할 때, 자료 구조 형태로 전달되어 이용될 때 데이터가 결합되어 있다고 한다.

② 내용 결합도(Content Coupling)는 하나의 모듈이 직접적으로 다른 모듈의 내용을 참조할 때 두 모듈은 내용적으로 결합되어 있다고 한다.

③ 공통 결합도(Common Coupling)는 두 모듈이 동일한 전역 데이터를 접근한다면 공통 결합 되어 있다고 한다.

④ 결합도(Coupling)는 두 모듈 간의 상호작용, 또는 의존도 정도를 나타내는 것이다.

2. 다음 중 가장 결합도가 강한 것은?

① Data Coupling ② Stamp Coupling

③ Common Coupling ④ Control Coupling

3. 어떤 모듈이 다른 모듈의 내부 논리 조직을 제어하기 위한 목적으로 제어 신호를 이용하여 통신하는 경우이며, 하위 모듈에서 상위 모듈로 제어 신호가 이동하여 상위 모듈에게 처리 명령을 부여하는 권리 전도 현상이 발생하게 되는 결합도는?

① Data Coupling ② Stamp Coupling

③ Control Coupling ④ Common Coupling

4. 소프트웨어 개발에서 모듈(Module)이 되기 위한 주요 특징에 해당하지 않는 것은?

① 다른 것들과 구별될 수 있는 독립적인 기능을 가진 단위(Unit)이다.

② 독립적인 컴파일이 가능하다.

③ 유일한 이름을 가져야 한다.

④ 다른 모듈에서의 접근이 불가능 해야 한다.

1. 애플리케이션 설계-SEC_05(모듈) 기출 문제

기출 문제(모듈)

5. 응집도의 종류 중 서로 간에 어떠한 의미 있는 연관관계 도 지니지 않은 기능 요소로 구성되는 경우이며, 서로 다른 상위 모듈에 의해 호출되어 처리상의 연관성이 없는 서로 다른 기능을 수행하는 경우의 응집도는?

① Functional Cohesion ② Sequential Cohesion

③ Logical Cohesion ④ Coincidental Cohesion

6. 모듈화(Modularity)와 관련한 설명으로 틀린 것은?

① 시스템을 모듈로 분할하면 각각의 모듈을 별개로 만들고 수정할 수 있기 때문에 좋은 구조가 된다.

② 응집도는 모듈과 모듈 사이의 상호의존 또는 연관 정도를 의미한다.

③ 모듈 간의 결합도가 약해야 독립적인 모듈이 될 수 있다.

④ 모듈 내 구성 요소들 간의 응집도가 강해야 좋은 모듈 설계이다.

7. 응집도가 가장 낮은 것은?

① 기능적 응집도 ② 시간적 응집도

③ 절차적 응집도 ④ 우연적 응집도

8. N-S(Nassi-Schneiderman) Chart에 대한 설명으로 거리가 먼 것은?

① 논리의 기술에 중점을 둔 도형식 표현 방법이다.

② 연속, 선택 및 다중 선택, 반복 등의 제어 논리 구조로 표현한다.

③ 주로 화살표를 사용하여 논리적인 제어 구조로 흐름을 표현한다.

④ 조건이 복합되어 있는 곳의 처리를 시각적으로 명확히 식별하는데 적합하다.

1. 애플리케이션 설계-SEC_05(모듈) 기출 문제

기출 문제(모듈)

9. 결합도가 낮은 것부터 높은 순으로 옳게 나열한 것은?

(ㄱ) 내용 결합도 (ㄴ) 자료 결합도

(ㄷ) 공통 결합도 (ㄹ) 스탬프 결합도

(ㅁ) 외부 결합도 (ㅂ) 제어 결합도

① (ㄱ) → (ㄴ) → (ㄹ) → (ㅂ) → (ㅁ) → (ㄷ)

② (ㄴ) → (ㄹ) → (ㅁ) → (ㅂ) → (ㄷ) → (ㄱ)

③ (ㄴ) → (ㄹ) → (ㅂ) → (ㅁ) → (ㄷ) → (ㄱ)

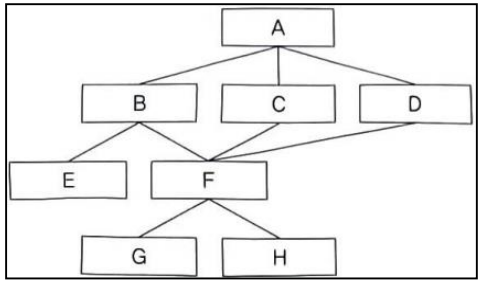
④ (ㄱ) → (ㄴ) → (ㄹ) → (ㅁ) → (ㅂ) → (ㄷ)

10. 다음 중 가장 강한 응집도(Cohesion)는?

① Sequential Cohesion ② Procedural Cohesion

③ Logical Cohesion ④ Coincidental Cohesion

11. 다음은 어떤 프로그램 구조를 나타낸다. 모듈 F에서의 fan-in, fan-out의 수는 얼마인가?



① fan-in: 2, fan-out: 3 ② fan-in: 3, fan-out: 2

③ fan-in: 1, fan-out: 2 ④ fan-in: 2, fan-out: 1

12. 프로그램 설계도의 하나인 NS-Chart에 대한 설명으로 가장 거리가 먼 것은?

① 논리의 기술에 중점을 두고 도형을 이용한 표현 방법이다.

② 이해하기 쉽고 코드 변환이 용이하다.

③ 화살표나 GOTO를 사용하여 이해하기 쉽다.

④ 연속 선택, 반복 등의 제어 논리 구조를 표현한다.

1. 애플리케이션 설계-SEC_06(공통 모듈)기출 및 출제 예상 문제

기출 문제 및 출제 예상문제(공통 모듈)

1. 공통 모듈에 대한 명세 기법 중 해당 기능에 대해 일관 되게 이해되고 한 가지로 해석될 수 있도록 작성하는 원칙은?

① 상호 작용성 ② 명확성

③ 독립성 ④ 내용성

2. 공통모듈의 재사용 범위에 따른 분류가 아닌 것은?

① 컴포넌트 재사용 ② 더미코드 재사용

③ 함수와 객체 재사용 ④ 애플리케이션 재사용

3. 명백한 역할을 가지고 독립적으로 존재할 수 있는 시스템의 부분으로 넓은 의미에서는 재사용 되는 모든 단위라고 볼 수 있으며, 인터페이스를 통해서만 접근할 수 있는 것은?

① Model ② Sheet

③ Component ④ Cell

4. 바람직한 소프트웨어 설계 지침이 아닌 것은?

① 적당한 모듈의 크기를 유지한다.

② 모듈 간의 접속 관계를 분석하여 복잡도와 중복을 줄인다.

③ 모듈 간의 결합도는 강할수록 바람직하다.

④ 모듈 간의 효과적인 제어를 위해 설계에서 계층적 자료 조직이 제시 되어야 한다.

1. 애플리케이션 설계-SEC_06(공통 모듈)기출 및 출제 예상 문제

기출 문제 및 출제 예상문제(공통 모듈)

5. 소프트웨어의 일부분을 다른 시스템에서 사용할 수 있는 정도를 의미하는 것은?

① 신뢰성(Reliability)

② 유지보수성(Maintainability)

③ 가시성(Visibility)

④ 재사용성(Reusability)

6. 좋은 소프트웨어 설계를 위한 소프트웨어의 모듈간의 결합도(Coupling)와 모듈 내 요소 간 응집도(Cohesion)에 대한 설명으로 옳은 것은?

① 응집도는 낮게 결합도는 높게 설계한다.

② 응집도는 높게 결합도는 낮게 설계한다.

③ 양쪽 모두 낮게 설계한다.

④ 양쪽 모두 높게 설계한다.

7. 효과적인 모듈 설계를 위한 유의사항으로 거리가 먼 것은?

① 모듈간의 결합도를 약하게 하면 모듈 독립성이 향상된다.

② 복잡도와 중복성을 줄이고 일관성을 유지시킨다.

③ 모듈의 기능은 예측이 가능해야 하며 지나치게 제한적이어야 한다.

④ 유지보수가 용이해야 한다.

8. 소프트웨어 재사용에 대한 내용 중 옳지 않은 것은?

① 비용을 절감하고 개발 시간을 단축하여 생산성이 증가한다.

② 재사용되는 대상은 외부 모듈과의 결합도가 낮아야 한다

③ 규모에 따라 변수, 함수, 객체, 컴포넌트 단위로 재사용 된다.

④ 재사용을 위해서는 사용법이 공개되어야 한다.

1. 애플리케이션 설계-SEC_06(공통 모듈)기출 및 출제 예상 문제

기출 문제 및 출제 예상 문제(공통 모듈)

9. 공통 모듈을 설계할 때 공통 부분을 명세하기 위한 기법 에 해당하지 않는 것은?

① 독립성　　② 명확성

③ 일관성　　④ 정확성

1. 애플리케이션 설계-SEC_07(코드)기출 및 출제 예상 문제

기출 문제 및 출제 예상 문제(코드)

1. 코드의 기본 기능으로 거리가 먼 것은?

① 복잡성 ② 표준화

③ 분류 ④ 식별

2. 코드 설계에서 일정한 일련번호를 부여하는 방식의 코드는?

① 연상 코드 ② 블록 코드

③ 순차 코드 ④ 표의 숫자 코드

3. 코드화 대상 항목의 중량, 면적, 용량 등의 물리적 수치를 이용하여 만든 코드는?

① 순차 코드 ② 10진 코드

③ 표의 숫자 코드 ④ 블록 코드

4. 사원 번호의 발급 과정에서 둘 이상의 서로 다른 사람에게 동일한 번호가 부여된 경우에 코드의 어떤 기능을 만족시키지 못한 것인가?

① 표준화 기능 ② 식별 기능

③ 배열 기능 ④ 연상 기능

1. 애플리케이션 설계-SEC_07(코드)기출 및 출제 예상 문제

기출 문제 및 출제 예상 문제(코드)

5. 회사에서 각 부서의 명칭을 코드화하기 위하여 대분류, 중분류, 소분류 등으로 나누어 나타내고자 한다. 이 때 가장 적합한 코드의 종류는?

① 구분 코드(Block Code)

② 그룹 분류 코드(Group Classification Code)

③ 연상 기호 코드(Mnemonic Code)

④ 순차 코드(Sequence Code)

6. 코드화 대상 자료 전체를 계산하여 이를 필요로 하는 분류 단위로 블록을 구분하고, 각 블록 내에서 순서대로 번호를 부여하는 방식으로, 적은 자릿수로 많은 항목 표시 가 가능하고 예비 코드를 사용할 수 있어 추가가 용이하다. 구분 순차 코드라고도 하는 이것을 무엇이라 하는가?

① 순차(Sequence) 코드

② 표의 숫자(Significant Digit) 코드

③ 블록(Block) 코드

④ 연상(Mnemonic) 코드

7. 코드 부여 체계에 대한 설명으로 옳지 않은 것은?

① 모듈이나 컴포넌트에 식별할 수 있는 코드를 부여하는 것을 말한다.

② 프로그래머가 모듈을 개발할 때 마다 임의로 코드를 부여한다.

③ 하나 이상의 코드를 조합하여 사용한다.

④ 코드 부여 체계의 담당자는 코드 규칙을 상세히 정의해야 한다.

8. 코드의 주요 기능에서 다양한 데이터를 기준에 맞추어 표현할 수 있는 기능은?

① 식별 기능 ② 분류 기능

③ 표준화 기능 ④ 배열 기능

1. 애플리케이션 설계-SEC_07(코드)기출 및 출제 예상 문제

기출 문제 및 출제 예상 문제(코드)

9. 필요한 기능을 하나의 코드로 수행하기 어려운 경우 2개 이상의 코드를 조합하여 만드는 방법은 무엇인가?

① 구분 코드(Block Code)

② 합성 코드(Combined Code)

③ 연상 기호 코드(Mnemonic Code)

④ 순차 코드(Sequence Code)

10. 소프트웨어 개발에서 코드를 부여할 대상이 되는 개체 가 아닌 것은?

① 모듈 ② 컴포넌트

③ 클래스 ④ 인터페이스

1. 애플리케이션 설계-SEC_08(디자인 패턴)기출 및 출제 예상 문제

기출 문제 및 출제 예상 문제(디자인 패턴)

1. 소프트웨어 설계에서 자주 발생하는 문제에 대한 일반적 이고 반복적인 해결 방법을 무엇이라고 하는가?

① 모듈 분해 ② 디자인 패턴

③ 연관 관계 ④ 클래스 도출

2. GoF(Gangs of Four) 디자인 패턴에서 생성(Creational) 패턴에 해당하는 것은?

① 컴포지트(구조 패턴) ② 어댑터(구조 패턴)

③ 추상 팩토리 ④ 옵서버(행위 패턴)

3. 디자인 패턴 사용의 장단점에 대한 설명으로 거리가 먼 것은?

① 소프트웨어 구조 파악이 용이하다.

② 객체지향 설계 및 구현의 생산성을 높이는데 적합하다.

③ 재사용을 위한 개발 시간이 단축된다.

④ 절차형 언어와 함께 이용될 때 효율이 극대화된다.

4. 다음 내용이 설명하는 디자인 패턴은?

◆ 객체를 생성하기 위한 인터페이스를 정의하여 어떤 클래스가 인스턴스화 될 것인지는 서브 클래스가 결정하도록 하는 것

◆ Virtual-Constructor 패턴이라고도 함

① Visitor 패턴(행위 패턴) ② Observer 패턴(행위 패턴)

③ Factory Method 패턴 ④ Bridge 패턴(구조 패턴)

1. 애플리케이션 설계-SEC_08(디자인 패턴)기출 및 출제 예상 문제

기출 문제 및 출제 예상 문제(디자인 패턴)

5. GoF(Gang of Four)디자인 패턴을 생성, 구조, 행동 패턴 의 세 그룹으로 분류할 때, 구조 패턴이 아닌 것은?

① Adapter 패턴 (구조 패턴) ② Bridge 패턴(구조 패턴)

③ Builder 패턴(생성 패턴) ④ Proxy 패턴(구조 패턴)

6. 디자인 패턴 중에서 행위적 패턴에 속하지 않는 것은?

① 커맨드(Command) 패턴(행위 패턴)

② 옵서버(Observer) 패턴(행위 패턴)

③ 프로토타입(Prototype) 패턴(생성 패턴)

④ 상태(State) 패턴(행위 패턴)

설명 : 행위 패턴은 하나의 객체로 수행할 수 없는 작업을 여러 객체로 분배하면서 결합도를 최소화 할 수 있도록 도움을 주는 패턴이다.

행위 패턴의 종류에는 책임 연쇄, 커맨드, 인터프리터, 반복자, 중재자, 메멘토, 옵서버, 상태, 전략, 템플릿 메소드, 방문자 패턴이 있다.

7. 디자인 패턴을 이용한 소프트웨어 재사용으로 얻어지는 장점이 아닌 것은?

① 소프트웨어 코드의 품질을 향상시킬 수 있다.

② 개발 프로세스를 무시할 수 있다.

③ 개발자들 사이의 의사소통을 원활하게 할 수 있다.

④ 소프트웨어의 품질과 생산성을 향상시킬 수 있다.

8. GoF(Gangs of Four) 디자인 패턴에 대한 설명으로 틀린 것은?

① Factory Method Pattern은 상위 클래스에서 객체를 생성하는 인터페이스를 정의하고, 하위클래스에서 인스턴스를 생성하도록 하는 방식이다.

② Prototype Pattern은 Prototype을 먼저 생성하고 인스턴스를 복제 하여 사용하는 구조이다.

③ Bridge Pattern은 기존에 구현되어 있는 클래스에 기능 발생 시 기존 클래스를 재사용할 수 있도록 중간에서 맞춰주는 역할을 한다.

④ Mediator Pattern은 객체간의 통제와 지시의 역할을 하는 중재자를 두어 객체지향의 목표를 달성하게 해준다.

설명 : 브리지 패턴은 구현부에서 추상층을 분리하여 서로가 독립적으 로 확장할 수 있도록 구성한 패턴이다. 기존 클래스를 이용하고 싶을 때, 중간

1. 애플리케이션 설계-SEC_08(디자인 패턴)기출 및 출제 예상 문제

기출 문제 및 출제 예상 문제(디자인 패턴)

9. GoF(Gangs of Four) 디자인 패턴 중 생성 패턴으로 옳은 것은?

① Singleton Pattern ② Adapter Pattern(구조 패턴)

③ Decorator Pattern(구조 패턴) ④ State Pattern(행위 패턴)

설명 : 생성 패턴의 종류에는 추상 팩토리, 빌더, 팩토리 메서드, 프로토타입, 싱글톤 패턴이 있다.

10. GoF(Gangs of Four) 디자인 패턴의 생성 패턴에 속하지 않는 것은?

① 추상 팩토리(Abstract Factory)

② 빌더(Builder)

③ 어댑터(Adapter) – 구조 패턴

④ 싱글톤(Singleton)

11. GoF(Gang of Four) 디자인 패턴과 관련한 설명으로 틀린 것은?

① 디자인 패턴을 목적(Purpose)으로 분류할 때 생성, 구조, 행위로 분류할 수 있다.

② Strategy 패턴은 대표적인 구조 패턴으로 인스턴스를 복제하여 사용하는 구조를 말한다.

③ 행위 패턴은 클래스나 객체들이 상호작용하는 방법과 책임을 분산 하는 방법을 정의한다.

④ Singleton 패턴은 특정 클래스의 인스턴스가 오직 하나임을 보장 하고, 이 인스턴스에 대한 접근 방법을 제공한다.

설명 : Strategy 패턴은 동일한 계열의 알고리즘들을 개별적으로 캡슐화하여 상호 교환할 수 있게 정의하는 패턴이며, 클라이언트는 독립적으로 원하는 알고리즘을 선택하여 사용할 수가 있고 클라이언트에 영향 없이 알고리즘 변경이 가능하다.

프로토타입 패턴은 생성 패턴으로 인스턴스를 복제하여 사용하는 구조를 말한다.

12. GoF(Gang of Four)의 디자인 패턴에서 행위 패턴에 속하는 것은?

① Builder (생성 패턴) ② Visitor(행위 패턴)

1. 애플리케이션 설계- SEC_01(소프트웨어 아키텍처) 기출 및 예상 문제

기출 및 예상 문제(소프트웨어 아키텍처)

1. 소프트웨어의 상위 설계에 속하지 않는 것은?

① 아키텍처 설계 ② 모듈 설계

③ 인터페이스 정의 ④ 사용자 인터페이스 설계

2. 다음 ( ) 안에 들어갈 내용으로 옳은 것은?

컴포넌트 설계 시 "( )에 의한 설계"를 따를 경우, 해당 명세에서는

(1) 컴포넌트의 오퍼레이션 사용 전에 참이 되어야 할 선행 조건

(2) 사용 후 만족되어야 할 결과 조건

(3) 오퍼레이션이 실행되는 동안 항상 만족되어야 할 불변 조건 등이 포함되어야 한다.

① 협약(Contract) ② 프로토콜(Protocol)

③ 패턴(Pattern) ④ 관계(Relation)

3. 소프트웨어 설계에서 사용되는 대표적인 추상화(Abstraction) 기법이 아닌 것은?

① 자료 추상화 ② 제어 추상화

③ 과정 추상화 ④ 강도 추상화

4. 객체지향 설계에서 정보 은닉(Information Hiding)과 관련한 설명으로 틀린 것은?

① 필요하지 않은 정보는 접근할 수 없도록 하여 한 모듈 또는 하부 시스템이 다른 모듈의 구현에 영향을 받지 않게 설계되는 것을 의미 한다.

② 모듈들 사이의 독립성을 유지시키는 데 도움이 된다.

③ 설계에서 은닉되어야 할 기본 정보로는 IP주소와 같은 물리적 코드, 상세 데이터 구조 등이 있다.

④ 모듈 내부의 자료 구조와 접근 동작들에만 수정을 국한하기 때문에 요구사항 등 변화에 따른 수정이 불가능하다.

1. 애플리케이션 설계- SEC_01(소프트웨어 아키텍처) 기출 및 예상 문제

기출 및 예상 문제(소프트웨어 아키텍처)

5. 소프트웨어 아키텍처 설계에서 시스템 품질 속성이 아닌 것은?

① 가용성(Availability)

② 독립성(Isolation)

③ 변경 용이성(Modifiability)

④ 사용성(Usability)

6. 아키텍처 설계 과정이 올바른 순서로 나열된 것은?

㉮ 설계 목표 설정

㉯ 시스템 타입 결정

㉰ 스타일 적용 및 커스터마이즈

㉱ 서브 시스템의 기능, 인터페이스 동작 작성

㉲ 아키텍처 설계 검토

① 가 -> 나 -> 다 -> 라 -> 마

② 마 -> 가 -> 나 -> 라 -> 다

③ 가 -> 마 -> 나 -> 라 -> 다

④ 가 -> 나 -> 다 -> 마 -> 라

7. 모듈화(Modularity)와 관련한 설명으로 틀린 것은?

① 소프트웨어의 모듈은 프로그래밍 언어에서 Subroutine, Function 등으로 표현될 수 있다.

② 모듈의 수가 증가하면 상대적으로 각 크기가 커지며, 모듈 사이의 상호교류가 감소하여 과부하(Overload) 현상이 나타난다.

③ 모듈화는 시스템을 지능적으로 관리할 수 있도록 해주며, 복잡도 문제를 해결하는 데 도움을 준다.

④ 모듈화는 시스템의 유지 보수와 수정을 용이하게 한다.

8. 소프트웨어 아키텍처 설계에 대한 설명으로 옳지 않은 것은?

① 이해 관계자들의 의사소통 도구로 활용된다.

② 설계된 모듈을 프로그래밍 언어를 통해 구현한다.

③ 애플리케이션을 모듈로 분할하고, 모듈 간 인터페이스를 결정하는 과정이다.

④ 기본 원리에는 모듈화, 추상화, 단계적 분해, 정보은닉이 있다.

1. 애플리케이션 설계- SEC_01(소프트웨어 아키텍처) 기출 및 예상 문제

기출 및 예상 문제(소프트웨어 아키텍처)

9. 소프트웨어 모듈화의 장점이 아닌 것은?

① 오류의 파급 효과를 최소화한다.

② 기능의 분리가 가능하여 인터페이스가 복잡하다.

③ 모듈의 재사용 가능으로 개발과 유지보수가 용이하다.

④ 프로그램의 효율적인 관리가 가능하다.

1. 애플리케이션 설계- SEC_02(아키텍처 패턴) 기출 및 예상 문제

기출 및 예상 문제(아키텍처 패턴)

1. 파이프 필터 형태의 소프트웨어 아키텍처에 대한 설명 으로 옳은 것은?

① 노드와 간선으로 구성된다.

② 서브시스템이 입력 데이터를 받아 처리하고 결과를 다음 서브시스템으로 넘겨주는 과정을 반복한다.

③ 계층 모델이라고도 한다.

④ 3개의 서브시스템(모델, 뷰, 제어)으로 구성되어 있다.

2. 소프트웨어 아키텍처와 관련한 설명으로 틀린 것은?

① 파이프 필터 아키텍처에서 데이터는 파이프를 통해 양방향으로 흐르며, 필터 이동 시 오버헤드가 발생하지 않는다.(단방향, 오버헤드가 데이터 변환 시 발생 가능성)

② 외부에서 인식할 수 있는 특성이 담긴 소프트웨어의 골격이 되는 기본 구조로 볼 수 있다.

③ 데이터 중심 아키텍처는 공유 데이터 저장소를 통해 접근자 간의 통신이 이루어지므로 각 접근자의 수정과 확장이 용이하다.

④ 이해 관계자들의 품질 요구사항을 반영하여 품질 속성을 결정한다.

3. 서브시스템이 입력 데이터를 받아 처리하고 결과를 다른 시스템에 보내는 작업이 반복되는 아키텍처 스타일은?

① 클라이언트 서버 구조 ② 계층 구조

③ MVC 구조 ④ 파이프 필터 구조

4. 분산시스템을 위한 마스터-슬레이브(Master-Slave) 아키텍처에 대한 설명으로 틀린 것은?

① 일반적으로 실시간 시스템에서 사용된다.

② 마스터 프로세스는 일반적으로 연산, 통신, 조정을 책임진다.

③ 슬레이브 프로세스는 데이터 수집 기능을 수행할 수 없다.

④ 마스터 프로세스는 슬레이브 프로세스들을 제어할 수 있다.

1. 애플리케이션 설계- SEC_02(아키텍처 패턴) 기출 및 예상 문제

기출 및 예상 문제(아키텍처 패턴)

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

① Peer-To-Peer Model ② MVC Model

③ Layers Model ④ Client-Server Model

6. 소프트웨어 아키텍처 모델 중 MVC와 관련한 설명으로 틀린 것은?

① MVC 모델은 사용자 인터페이스를 담당하는 계층의 응
집도를 높일 수 있고, 여러 개의 다른 UI를 만들어 그 사이 에 결합도를 낮출 수 있다.

② 모델(Model)은 뷰(View)와 제어(Controller)사이에서
전달자 역할을 하며, 뷰마다 모델 서브시스템이 각각 하나 씩 연결된다.

③ 뷰(View)는 모델(Model)에 있는 데이터를 사용자 인터
페이스에 보이는 역할을 담당한다.

④ 제어(Controller)는 모델(Model)에 명령을 보냄으로써
모델의 상태를 변경할 수 있다.

7. 아키텍처 패턴(Architecture Pattern)에 대한 설명 중 가장 옳지
않은 것은?

① 소프트웨어 초기 설계에서 발생하는 문제들을 해결하기 위한 전형 적인 해결 방식들을 의미한다.

② 검증된 구조로 개발하기 때문에 오류가 적어 개발시간을 단축할 수 있다.

③ 서브시스템들에 대한 역할을 정의하고 있지만, 그들 간의 인터페이스에 대한 지침은 없다.

④ 시스템에 대한 이해가 쉬워지고, 특성을 예측할 수 있게 된다.

8. 다음 중 클라이언트-서버(Client-Server) 모델에 대한 설명으로
가장 거리가 먼 것은?

① 사용자는 클라이언트를 통해서 요청을 전달하며, 서버는 이에 응답하는 방식이다.

② 서버는 클라이언트의 요청에 대비하여 항상 대기 상태를 유지한다.

③ 서버와 클라이언트는 서로 독립적이다.

④ 다수의 서버와 하나의 클라이언트로 구성되는 패턴으로 분산 환경
시스템에 적합하다.

1. 애플리케이션 설계- SEC_02(아키텍처 패턴) 기출 및 예상 문제

기출 및 예상 문제(아키텍처 패턴)

9. 여러 컴포넌트들 중 각 컴포넌트들이 서비스를 제공하는 서버가 될 수도 있고, 서비스를 요청하는 클라이언트도 될 수 있는 패턴으로 전형적인 멀티스레딩을 사용하는 방식의 패턴을 무엇이라 하는가?

① 클라이언트-서버 ② 블랙보드

③ 이벤트-버스 ④ 피어-투-피어

10. 다음 중, 이벤트-버스 패턴(Event-Bus Pattern)의 주요 컴포넌트가 아닌 것은?

① 소스 ② 리스너

③ 채널 ④ 버퍼

1. 애플리케이션 설계-SEC_03(객체 지향(Object-Oriented)) 기출 문제

기출 및 예상 문제(객체 지향(Object-Oriented))

1. 객체에 대한 설명으로 틀린 것은?

① 객체는 상태, 동작, 고유 식별자를 가진 모든 것이라 할 수 있다.

② 객체는 공통 속성을 공유하는 클래스들의 집합이다.

③ 객체는 필요한 자료 구조와 이에 수행되는 함수들을 가진 하나의 독립된 존재이다.

④ 객체의 상태는 속성값에 의해 정의된다.

2. 객체지향개념 중 하나 이상의 유사한 객체들을 묶어 공통된 특성을 표현한 데이터 추상화를 의미하는 것은?

① Method ② Class

③ Field ④ Message

3. 객체지향의 주요 개념에 대한 설명으로 틀린 것은?

① 캡슐화는 상위 클래스에서 속성이나 연산을 전달받아 새로운 형태의 클래스로 확장하여 사용하는 것을 의미한다.

② 객체는 실세계에 존재하거나 생각할 수 있는 것을 말한다.

③ 클래스는 하나 이상의 유사한 객체들을 묶어 공통된 특성을 표현한 것이다.

④ 다형성은 상속받은 여러 개의 하위 객체들이 다른 형태의 특성을 갖는 객체로 이용될 수 있는 성질이다.

4. 객체지향에서 정보은닉과 가장 밀접한 관계가 있는 것은?

① Encapsulation ② Class

③ Method ④ Instance

1. 애플리케이션 설계-SEC_03(객체 지향(Object-Oriented)) 기출 문제

기출 및 예상 문제(객체 지향(Object-Oriented))

5. 객체지향 기법에서 클래스들 사이의 '부분 전체(Part-Whole)' 관계 또는 부분(is-a-part-of)'의 관계로 설명되는 연관성을 나타내는 용어는?

① 일반화 ② 추상화

③ 캡슐화 ④ 집단화

6. 객체에게 어떤 행위를 하도록 지시하는 명령은?

① Class ② Package

③ Object ④ Message

7. 객체지향 기법에서 같은 클래스에 속한 각각의 객체를 의미하는 것은?

① Instance ② Message

③ Method ④ Module

8. 객체지향 개념에서 연관된 데이터와 함수를 함께 묶어 외부와 경계 를 만들고 필요한 인터페이스만을 밖으로 드러내는 과정은?

① 메시지(Message) ② 캡슐화(Encapsulation)

③ 다형성(Polymorphism) ④ 상속(Inheritance)

1. 애플리케이션 설계-SEC_03(객체 지향(Object-Oriented)) 기출 문제

기출 및 예상 문제(객체 지향(Object-Oriented))

9. 객체지향 기법에서 상위 클래스의 메소드와 속성을 하위 클래스가 물려받는 것을 의미하는 것은?

① Abstraction(추상화)　② Polymorphism(다형성)

③ Encapsulation(캡슐화)　④ Inheritance(상속)

10. 객체지향 개념에서 다형성(Polymorphism)과 관련한 설명으로 틀린 것은?

① 다형성은 현재 코드를 변경하지 않고 새로운 클래스를 쉽게 추가할 수 있게 한다.

② 다형성이란 여러 가지 형태를 가지고 있다는 의미로, 여러 형태를 받아들일 수 있는 특징을 말한다.

③ 메소드 오버라이딩(Overriding)은 상위 클래스에서 정의한 일반 메소드의 구현을 하위 클래스에서 무시하고 재정의할 수 있다.

④ 메소드 오버로딩(Overloading)의 경우 매개 변수 타입은 동일하지만 메소드 명을 다르게 함으로써 구현, 구분할 수 있다.

1. 애플리케이션 설계-SEC_04(객체지향 분석 및 설계) 기출 문제

기출 문제(객체지향 분석 및 설계)

1. 객체지향 분석 방법론 중 Coad-Yourdon 방법에 해당하는 것은?

① E-R 다이어그램을 사용하여 객체의 행위를 데이터 모델링 하는데 초점을 둔 방법이다.

② 객체, 동적, 기능 모델로 나누어 수행하는 방법이다.

③ 미시적 개발 프로세스와 거시적 개발 프로세스를 모두 사용하는 방법이다.

④ Use Case를 강조하여 사용하는 방법이다.

2. 그래픽 표기법을 이용하여 소프트웨어 구성 요소를 모델링 하는 럼바우 분석 기법에 포함되지 않는 것은?

① 객체 모델링 ② 기능 모델링

③ 동적 모델링 ④ 블랙박스 분석 모델링

3. 럼바우(Rumbaugh)의 객체지향분석 절차를 가장 바르게 나열한 것은?

① 객체 모형 → 동적 모형 → 기능 모형

② 객체 모형 → 기능 모형 → 동적 모형

③ 기능 모형 → 동적 모형 → 객체 모형

④ 기능 모형 → 객체 모형 → 동적 모형

4. 다음 내용이 설명하는 객체지향 설계 원칙은?

- 클라이언트는 자신이 사용하지 않는 메소드와 의존관계를 맺으면 안 된다.
- 클라이언트가 사용하지 않는 인터페이스 때문에 영향을 받아서는 안 된다.

① 인터페이스분리 원칙 ② 단일 책임 원칙

③ 개방 폐쇄의 원칙 ④ 리스코프 교체의 원칙

1. 애플리케이션 설계-SEC_04(객체지향 분석 및 설계) 기출 문제

기출 문제(객체지향 분석 및 설계)

5. 클래스 설계 원칙에 대한 바른 설명은?

① 단일 책임 원칙 : 하나의 클래스만 변경 가능해야 한다.

② 개방, 폐쇄의 원칙 : 클래스는 확장에 대해 열려 있어야 하며 변경에 대해 닫혀 있어야 한다.

③ 리스코프 교체의 원칙 : 여러 개의 책임을 가진 클래스는 하나의 책임을 가진 클래스로 대체 되어야 한다.

④ 의존관계 역전의 원칙 : 클라이언트는 자신이 사용하는 메소드와 의존관계를 갖지 않도록 해야 한다.

6. 소프트웨어를 개발하기 위한 비즈니스(업무)를 객체와 속성, 클래스와 멤버, 전체와 부분 등으로 나누어서 분석해 내는 기법은?

① 객체지향 분석 ② 구조적 분석

③ 기능적 분석 ④ 실시간 분석

7. 럼바우(Rumbaugh)의 객체지향 분석 기법 중 자료 흐름도(DFD)를 주로 이용하는 것은?

① 기능 모델링 ② 동적 모델링

③ 객체 모델링 ④ 정적 모델링

8. 럼바우(Rumbaugh) 분석 기법에서 정보 모델링이라고도 하며, 시스템에서 요구되는 객체를 찾아내어 속성과 연산 식별 및 객체들 간의 관계를 규정하여 다이어그램을 표시하는 모델링은?

① Object(객체) ② Dynamic(동적인)

③ Function(함수, 기능, 연산) ④ Static(정적인)

1. 애플리케이션 설계-SEC_05(모듈) 기출 문제

기출 문제(모듈)

1. 결합도(Coupling)에 대한 설명으로 틀린 것은?

① 데이터 결합도(Data Coupling)는 두 모듈이 매개변수로 자료를 전달할 때, 자료 구조 형태로 전달되어 이용될 때 데이터가 결합되어 있다고 한다.

② 내용 결합도(Content Coupling)는 하나의 모듈이 직접적으로 다른 모듈의 내용을 참조할 때 두 모듈은 내용적으로 결합되어 있다고 한다.

③ 공통 결합도(Common Coupling)는 두 모듈이 동일한 전역 데이터를 접근한다면 공통 결합 되어 있다고 한다.

④ 결합도(Coupling)는 두 모듈 간의 상호작용, 또는 의존도 정도를 나타내는 것이다.

2. 다음 중 가장 결합도가 강한 것은?

① Data Coupling ② Stamp Coupling

③ Common Coupling ④ Control Coupling

3. 어떤 모듈이 다른 모듈의 내부 논리 조직을 제어하기 위한 목적으로 제어 신호를 이용하여 통신하는 경우이며, 하위 모듈에서 상위 모듈로 제어 신호가 이동하여 상위 모듈에게 처리 명령을 부여하는 권리 전도 현상이 발생하게 되는 결합도는?

① Data Coupling ② Stamp Coupling

③ Control Coupling ④ Common Coupling

4. 소프트웨어 개발에서 모듈(Module)이 되기 위한 주요 특징에 해당하지 않는 것은?

① 다른 것들과 구별될 수 있는 독립적인 기능을 가진 단위(Unit)이다.

② 독립적인 컴파일이 가능하다.

③ 유일한 이름을 가져야 한다.

④ 다른 모듈에서의 접근이 불가능 해야 한다.

1. 애플리케이션 설계-SEC_05(모듈) 기출 문제

기출 문제(모듈)

5. 응집도의 종류 중 서로 간에 어떠한 의미 있는 연관관계 도 지니지 않은 기능 요소로 구성되는 경우이며, 서로 다른 상위 모듈에 의해 호출되어 처리상의 연관성이 없는 서로 다른 기능을 수행하는 경우의 응집도는?

① Functional Cohesion ② Sequential Cohesion

③ Logical Cohesion ④ Coincidental Cohesion

6. 모듈화(Modularity)와 관련한 설명으로 틀린 것은?

① 시스템을 모듈로 분할하면 각각의 모듈을 별개로 만들고 수정할 수 있기 때문에 좋은 구조가 된다.

② 응집도는 모듈과 모듈 사이의 상호의존 또는 연관 정도를 의미한다.

③ 모듈 간의 결합도가 약해야 독립적인 모듈이 될 수 있다.

④ 모듈 내 구성 요소들 간의 응집도가 강해야 좋은 모듈 설계이다.

7. 응집도가 가장 낮은 것은?

① 기능적 응집도 ② 시간적 응집도

③ 절차적 응집도 ④ 우연적 응집도

8. N-S(Nassi-Schneiderman) Chart에 대한 설명으로 거리가 먼 것은?

① 논리의 기술에 중점을 둔 도형식 표현 방법이다.

② 연속, 선택 및 다중 선택, 반복 등의 제어 논리 구조로 표현한다.

③ 주로 화살표를 사용하여 논리적인 제어 구조로 흐름을 표현한다.

④ 조건이 복합되어 있는 곳의 처리를 시각적으로 명확히 식별하는데 적합하다.

1. 애플리케이션 설계-SEC_05(모듈) 기출 문제

기출 문제(모듈)

9. 결합도가 낮은 것부터 높은 순으로 옳게 나열한 것은?

(ㄱ) 내용 결합도 (ㄴ) 자료 결합도

(ㄷ) 공통 결합도 (ㄹ) 스탬프 결합도

(ㅁ) 외부 결합도 (ㅂ) 제어 결합도

① (ㄱ) → (ㄴ) → (ㄹ) → (ㅂ) → (ㅁ) → (ㄷ)

② (ㄴ) → (ㄹ) → (ㅁ) → (ㅂ) → (ㄷ) → (ㄱ)

③ (ㄴ) → (ㄹ) → (ㅂ) → (ㅁ) → (ㄷ) → (ㄱ)

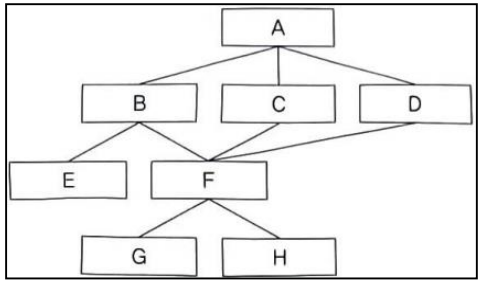
④ (ㄱ) → (ㄴ) → (ㄹ) → (ㅁ) → (ㅂ) → (ㄷ)

10. 다음 중 가장 강한 응집도(Cohesion)는?

① Sequential Cohesion ② Procedural Cohesion

③ Logical Cohesion ④ Coincidental Cohesion

11. 다음은 어떤 프로그램 구조를 나타낸다. 모듈 F에서의 fan-in, fan-out의 수는 얼마인가?



① fan-in: 2, fan-out: 3 ② fan-in: 3, fan-out: 2

③ fan-in: 1, fan-out: 2 ④ fan-in: 2, fan-out: 1

12. 프로그램 설계도의 하나인 NS-Chart에 대한 설명으로 가장 거리가 먼 것은?

① 논리의 기술에 중점을 두고 도형을 이용한 표현 방법이다.

② 이해하기 쉽고 코드 변환이 용이하다.

③ 화살표나 GOTO를 사용하여 이해하기 쉽다.

④ 연속 선택, 반복 등의 제어 논리 구조를 표현한다.

1. 애플리케이션 설계-SEC_06(공통 모듈)기출 및 출제 예상 문제

기출 문제 및 출제 예상문제(공통 모듈)

1. 공통 모듈에 대한 명세 기법 중 해당 기능에 대해 일관 되게 이해되고 한 가지로 해석될 수 있도록 작성하는 원칙은?

① 상호 작용성 ② 명확성

③ 독립성 ④ 내용성

2. 공통모듈의 재사용 범위에 따른 분류가 아닌 것은?

① 컴포넌트 재사용 ② 더미코드 재사용

③ 함수와 객체 재사용 ④ 애플리케이션 재사용

3. 명백한 역할을 가지고 독립적으로 존재할 수 있는 시스템의 부분으로 넓은 의미에서는 재사용 되는 모든 단위라고 볼 수 있으며, 인터페이스를 통해서만 접근할 수 있는 것은?

① Model ② Sheet

③ Component ④ Cell

4. 바람직한 소프트웨어 설계 지침이 아닌 것은?

① 적당한 모듈의 크기를 유지한다.

② 모듈 간의 접속 관계를 분석하여 복잡도와 중복을 줄인다.

③ 모듈 간의 결합도는 강할수록 바람직하다.

④ 모듈 간의 효과적인 제어를 위해 설계에서 계층적 자료 조직이 제시 되어야 한다.

1. 애플리케이션 설계-SEC_06(공통 모듈)기출 및 출제 예상 문제

기출 문제 및 출제 예상문제(공통 모듈)

5. 소프트웨어의 일부분을 다른 시스템에서 사용할 수 있는 정도를 의미하는 것은?

① 신뢰성(Reliability)

② 유지보수성(Maintainability)

③ 가시성(Visibility)

④ 재사용성(Reusability)

6. 좋은 소프트웨어 설계를 위한 소프트웨어의 모듈간의 결합도(Coupling)와 모듈 내 요소 간 응집도(Cohesion)에 대한 설명으로 옳은 것은?

① 응집도는 낮게 결합도는 높게 설계한다.

② 응집도는 높게 결합도는 낮게 설계한다.

③ 양쪽 모두 낮게 설계한다.

④ 양쪽 모두 높게 설계한다.

7. 효과적인 모듈 설계를 위한 유의사항으로 거리가 먼 것은?

① 모듈간의 결합도를 약하게 하면 모듈 독립성이 향상된다.

② 복잡도와 중복성을 줄이고 일관성을 유지시킨다.

③ 모듈의 기능은 예측이 가능해야 하며 지나치게 제한적이어야 한다.

④ 유지보수가 용이해야 한다.

8. 소프트웨어 재사용에 대한 내용 중 옳지 않은 것은?

① 비용을 절감하고 개발 시간을 단축하여 생산성이 증가한다.

② 재사용되는 대상은 외부 모듈과의 결합도가 낮아야 한다

③ 규모에 따라 변수, 함수, 객체, 컴포넌트 단위로 재사용 된다.

④ 재사용을 위해서는 사용법이 공개되어야 한다.

1. 애플리케이션 설계-SEC_06(공통 모듈)기출 및 출제 예상 문제

기출 문제 및 출제 예상 문제(공통 모듈)

9. 공통 모듈을 설계할 때 공통 부분을 명세하기 위한 기법 에 해당하지 않는 것은?

① 독립성　　② 명확성

③ 일관성　　④ 정확성

1. 애플리케이션 설계-SEC_07(코드)기출 및 출제 예상 문제

기출 문제 및 출제 예상 문제(코드)

1. 코드의 기본 기능으로 거리가 먼 것은?

① 복잡성 ② 표준화

③ 분류 ④ 식별

2. 코드 설계에서 일정한 일련번호를 부여하는 방식의 코드는?

① 연상 코드 ② 블록 코드

③ 순차 코드 ④ 표의 숫자 코드

3. 코드화 대상 항목의 중량, 면적, 용량 등의 물리적 수치를 이용하여 만든 코드는?

① 순차 코드 ② 10진 코드

③ 표의 숫자 코드 ④ 블록 코드

4. 사원 번호의 발급 과정에서 둘 이상의 서로 다른 사람에게 동일한 번호가 부여된 경우에 코드의 어떤 기능을 만족시키지 못한 것인가?

① 표준화 기능 ② 식별 기능

③ 배열 기능 ④ 연상 기능

1. 애플리케이션 설계-SEC_07(코드)기출 및 출제 예상 문제

기출 문제 및 출제 예상 문제(코드)

5. 회사에서 각 부서의 명칭을 코드화하기 위하여 대분류, 중분류, 소분류 등으로 나누어 나타내고자 한다. 이 때 가장 적합한 코드의 종류는?

① 구분 코드(Block Code)

② 그룹 분류 코드(Group Classification Code)

③ 연상 기호 코드(Mnemonic Code)

④ 순차 코드(Sequence Code)

6. 코드화 대상 자료 전체를 계산하여 이를 필요로 하는 분류 단위로 블록을 구분하고, 각 블록 내에서 순서대로 번호를 부여하는 방식으로, 적은 자릿수로 많은 항목 표시 가 가능하고 예비 코드를 사용할 수 있어 추가가 용이하다. 구분 순차 코드라고도 하는 이것을 무엇이라 하는가?

① 순차(Sequence) 코드

② 표의 숫자(Significant Digit) 코드

③ 블록(Block) 코드

④ 연상(Mnemonic) 코드

7. 코드 부여 체계에 대한 설명으로 옳지 않은 것은?

① 모듈이나 컴포넌트에 식별할 수 있는 코드를 부여하는 것을 말한다.

② 프로그래머가 모듈을 개발할 때 마다 임의로 코드를 부여한다.

③ 하나 이상의 코드를 조합하여 사용한다.

④ 코드 부여 체계의 담당자는 코드 규칙을 상세히 정의해야 한다.

8. 코드의 주요 기능에서 다양한 데이터를 기준에 맞추어 표현할 수 있는 기능은?

① 식별 기능 ② 분류 기능

③ 표준화 기능 ④ 배열 기능

1. 애플리케이션 설계-SEC_07(코드)기출 및 출제 예상 문제

기출 문제 및 출제 예상 문제(코드)

9. 필요한 기능을 하나의 코드로 수행하기 어려운 경우 2개 이상의 코드를 조합하여 만드는 방법은 무엇인가?

① 구분 코드(Block Code)

② 합성 코드(Combined Code)

③ 연상 기호 코드(Mnemonic Code)

④ 순차 코드(Sequence Code)

10. 소프트웨어 개발에서 코드를 부여할 대상이 되는 개체 가 아닌 것은?

① 모듈 ② 컴포넌트

③ 클래스 ④ 인터페이스

1. 애플리케이션 설계-SEC_08(디자인 패턴)기출 및 출제 예상 문제

기출 문제 및 출제 예상 문제(디자인 패턴)

1. 소프트웨어 설계에서 자주 발생하는 문제에 대한 일반적 이고 반복적인 해결 방법을 무엇이라고 하는가?

① 모듈 분해 ② 디자인 패턴

③ 연관 관계 ④ 클래스 도출

2. GoF(Gangs of Four) 디자인 패턴에서 생성(Creational) 패턴에 해당하는 것은?

① 컴포지트(구조 패턴) ② 어댑터(구조 패턴)

③ 추상 팩토리 ④ 옵서버(행위 패턴)

3. 디자인 패턴 사용의 장단점에 대한 설명으로 거리가 먼 것은?

① 소프트웨어 구조 파악이 용이하다.

② 객체지향 설계 및 구현의 생산성을 높이는데 적합하다.

③ 재사용을 위한 개발 시간이 단축된다.

④ 절차형 언어와 함께 이용될 때 효율이 극대화된다.

4. 다음 내용이 설명하는 디자인 패턴은?

◆ 객체를 생성하기 위한 인터페이스를 정의하여 어떤 클래스가 인스턴스화 될 것인지는 서브 클래스가 결정하도록 하는 것

◆ Virtual-Constructor 패턴이라고도 함

① Visitor 패턴(행위 패턴) ② Observer 패턴(행위 패턴)

③ Factory Method 패턴 ④ Bridge 패턴(구조 패턴)

1. 애플리케이션 설계-SEC_08(디자인 패턴)기출 및 출제 예상 문제

기출 문제 및 출제 예상 문제(디자인 패턴)

5. GoF(Gang of Four)디자인 패턴을 생성, 구조, 행동 패턴 의 세 그룹으로 분류할 때, 구조 패턴이 아닌 것은?

① Adapter 패턴 (구조 패턴) ② Bridge 패턴(구조 패턴)

③ Builder 패턴(생성 패턴) ④ Proxy 패턴(구조 패턴)

6. 디자인 패턴 중에서 행위적 패턴에 속하지 않는 것은?

① 커맨드(Command) 패턴(행위 패턴)

② 옵서버(Observer) 패턴(행위 패턴)

③ 프로토타입(Prototype) 패턴(생성 패턴)

④ 상태(State) 패턴(행위 패턴)

설명 : 행위 패턴은 하나의 객체로 수행할 수 없는 작업을 여러 객체로 분배하면서 결합도를 최소화 할 수 있도록 도움을 주는 패턴이다.

행위 패턴의 종류에는 책임 연쇄, 커맨드, 인터프리터, 반복자, 중재자, 메멘토, 옵서버, 상태, 전략, 템플릿 메소드, 방문자 패턴이 있다.

7. 디자인 패턴을 이용한 소프트웨어 재사용으로 얻어지는 장점이 아닌 것은?

① 소프트웨어 코드의 품질을 향상시킬 수 있다.

② 개발 프로세스를 무시할 수 있다.

③ 개발자들 사이의 의사소통을 원활하게 할 수 있다.

④ 소프트웨어의 품질과 생산성을 향상시킬 수 있다.

8. GoF(Gangs of Four) 디자인 패턴에 대한 설명으로 틀린 것은?

① Factory Method Pattern은 상위 클래스에서 객체를 생성하는 인터페이스를 정의하고, 하위클래스에서 인스턴스를 생성하도록 하는 방식이다.

② Prototype Pattern은 Prototype을 먼저 생성하고 인스턴스를 복제 하여 사용하는 구조이다.

③ Bridge Pattern은 기존에 구현되어 있는 클래스에 기능 발생 시 기존 클래스를 재사용할 수 있도록 중간에서 맞춰주는 역할을 한다.

④ Mediator Pattern은 객체간의 통제와 지시의 역할을 하는 중재자를 두어 객체지향의 목표를 달성하게 해준다.

설명 : 브리지 패턴은 구현부에서 추상층을 분리하여 서로가 독립적으 로 확장할 수 있도록 구성한 패턴이다. 기존 클래스를 이용하고 싶을 때, 중간

1. 애플리케이션 설계-SEC_08(디자인 패턴)기출 및 출제 예상 문제

기출 문제 및 출제 예상 문제(디자인 패턴)

9. GoF(Gangs of Four) 디자인 패턴 중 생성 패턴으로 옳은 것은?

① Singleton Pattern ② Adapter Pattern(구조 패턴)

③ Decorator Pattern(구조 패턴) ④ State Pattern(행위 패턴)

설명 : 생성 패턴의 종류에는 추상 팩토리, 빌더, 팩토리 메서드, 프로토타입, 싱글톤 패턴이 있다.

10. GoF(Gangs of Four) 디자인 패턴의 생성 패턴에 속하지 않는 것은?

① 추상 팩토리(Abstract Factory)

② 빌더(Builder)

③ 어댑터(Adapter) – 구조 패턴

④ 싱글톤(Singleton)

11. GoF(Gang of Four) 디자인 패턴과 관련한 설명으로 틀린 것은?

① 디자인 패턴을 목적(Purpose)으로 분류할 때 생성, 구조, 행위로 분류할 수 있다.

② Strategy 패턴은 대표적인 구조 패턴으로 인스턴스를 복제하여 사용하는 구조를 말한다.

③ 행위 패턴은 클래스나 객체들이 상호작용하는 방법과 책임을 분산 하는 방법을 정의한다.

④ Singleton 패턴은 특정 클래스의 인스턴스가 오직 하나임을 보장 하고, 이 인스턴스에 대한 접근 방법을 제공한다.

설명 : Strategy 패턴은 동일한 계열의 알고리즘들을 개별적으로 캡슐화하여 상호 교환할 수 있게 정의하는 패턴이며, 클라이언트는 독립적으로 원하는 알고리즘을 선택하여 사용할 수가 있고 클라이언트에 영향 없이 알고리즘 변경이 가능하다.

프로토타입 패턴은 생성 패턴으로 인스턴스를 복제하여 사용하는 구조를 말한다.

12. GoF(Gang of Four)의 디자인 패턴에서 행위 패턴에 속하는 것은?

① Builder (생성 패턴) ② Visitor(행위 패턴)