

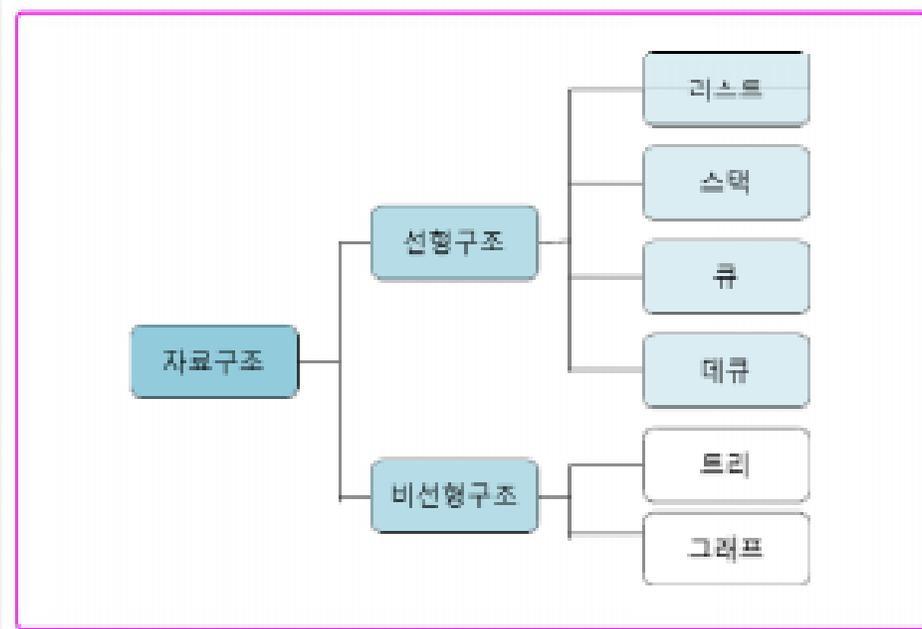
# 제2과목 소프트웨어 개발

## 01 데이터 입출력 구현 A

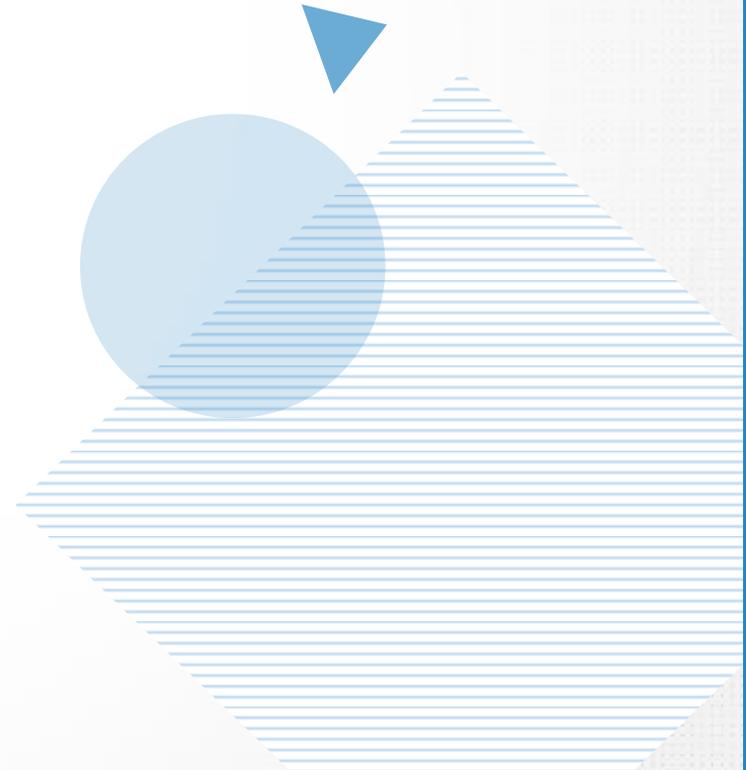


# 자료구조

- ▶ 자료구조의 정의
- ▶ 자료구조의 분류

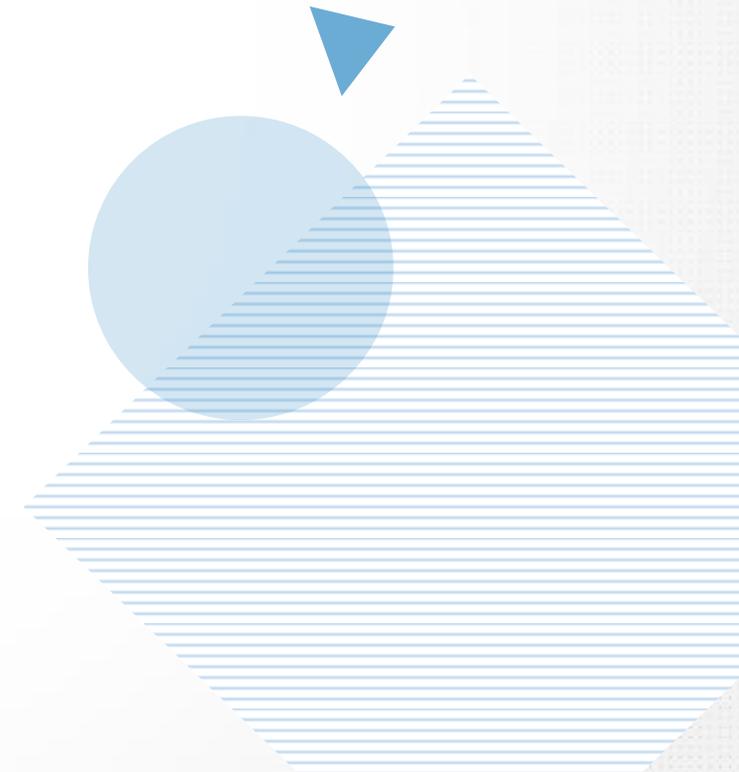


- 리스트(Linear List)
  - 배열(Array)

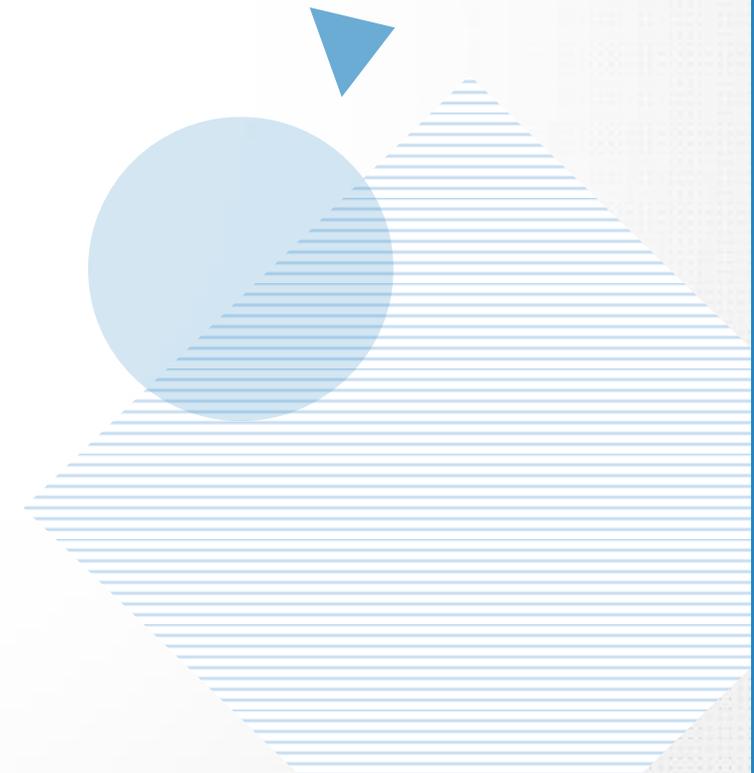
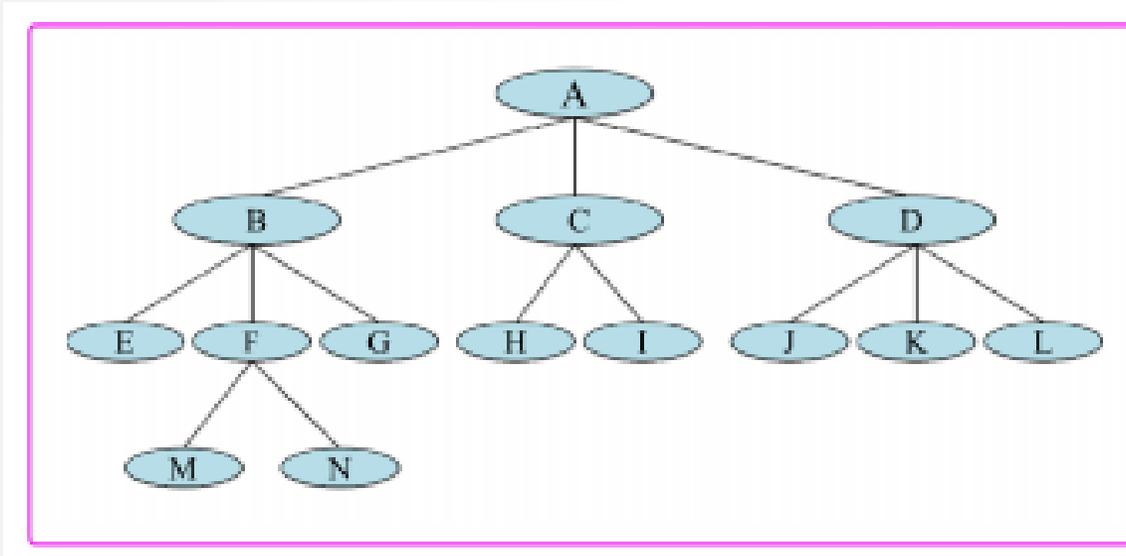


➤ 스택(Stack)

➤ 큐(Queue)



## ▶ 트리(Tree)



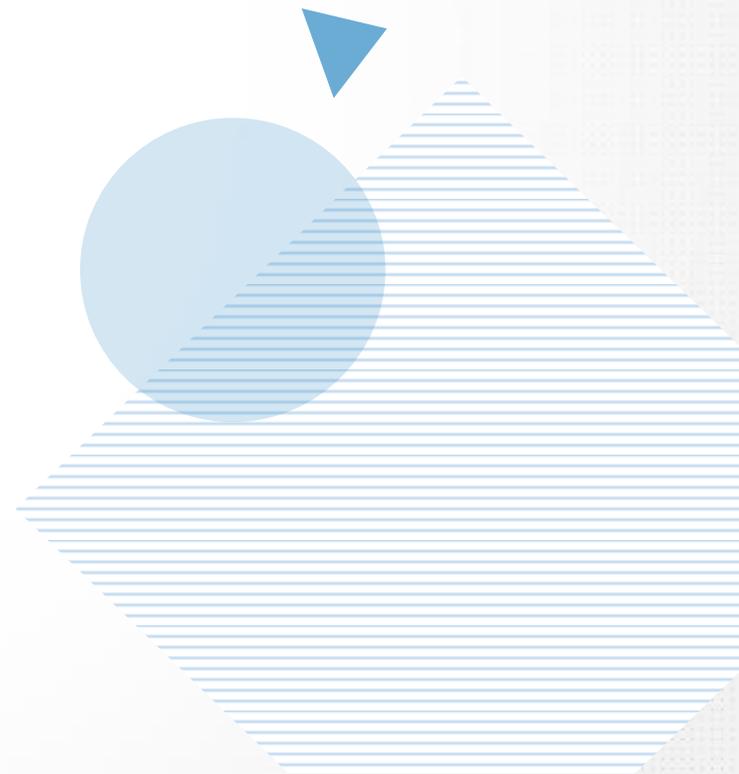


## 문제

➤ 리스트내에 데이터 삽입,삭제가 한쪽 끝에서 이루어 지는 데이터 구조를 무엇이라 하는가?

- ① 스택
- ② 큐
- ③ 데크
- ④ 원형 큐

정답 1



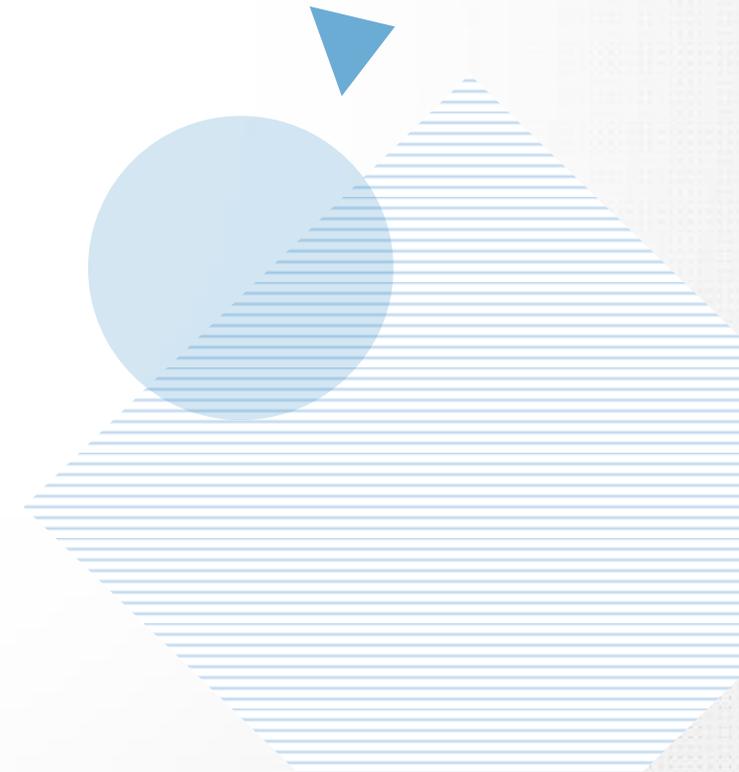
# 제2과목 소프트웨어 개발

## 02 데이터 입출력 구현 B



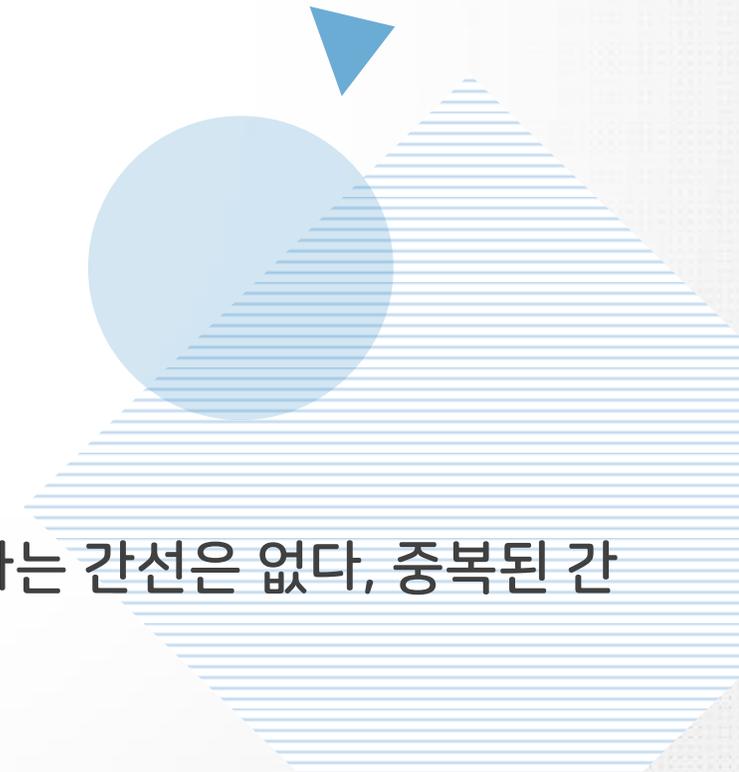
## ▶ 트리의 순회 방법

- 중위 순회
  - 왼쪽 서브트리 → 중간 노드 → 오른쪽 서브트리
- 전위 순회
  - 중간 노드 → 왼쪽 서브트리 → 오른쪽 서브트리
- 후위 순회
  - 왼쪽 서브트리 → 오른쪽 서브트리 → 중간 노드



## ▶ 그래프

- 그래프 개념
  - 객체 간의 관계를 표현할 수 있는 자료구조
- 그래프 표현 방법
- 그래프 종류
  - 무방향 그래프 : 선에 방향이 없다.
  - 방향 그래프 : 선에 방향이 있다.
- 그래프 특징
  - 네트워크 모델이다, 2개 이상의 경로가 가능하다, 자기 자신을 향하는 간선은 없다, 중복된 간선을 허용하지 않는다.



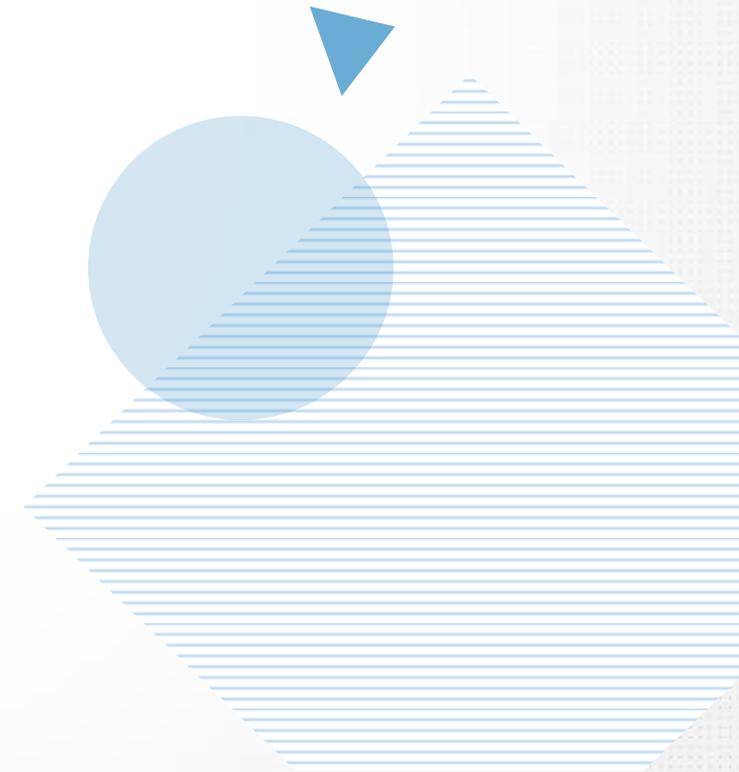
- 순차 파일
- 직접 파일
- 색인 순차 파일
  - 기본 영역
  - 색인영역(트랙, 실린더, 마스터)
  - 오버플로우 영역



# 1. 다음 중 트리에 대한 설명으로 옳은 것은?

- ① 루트노드가 많은 트리일수록 좋은 트리이다.
- ② 트리와 관련된 알고리즘을 재귀적인 방식으로 구현하면 실행시간이 빨라진다.
- ③ 트리의 최대레벨과 트리의 높이와는 무관하다.
- ④ 트리의 노드 중 차수가 0인 노드를 리프노드라고 한다.

정답4



# 제2과목 소프트웨어 개발

## 03 데이터 입출력 구현 C



## ➤ 데이터저장소

## ➤ 데이터베이스

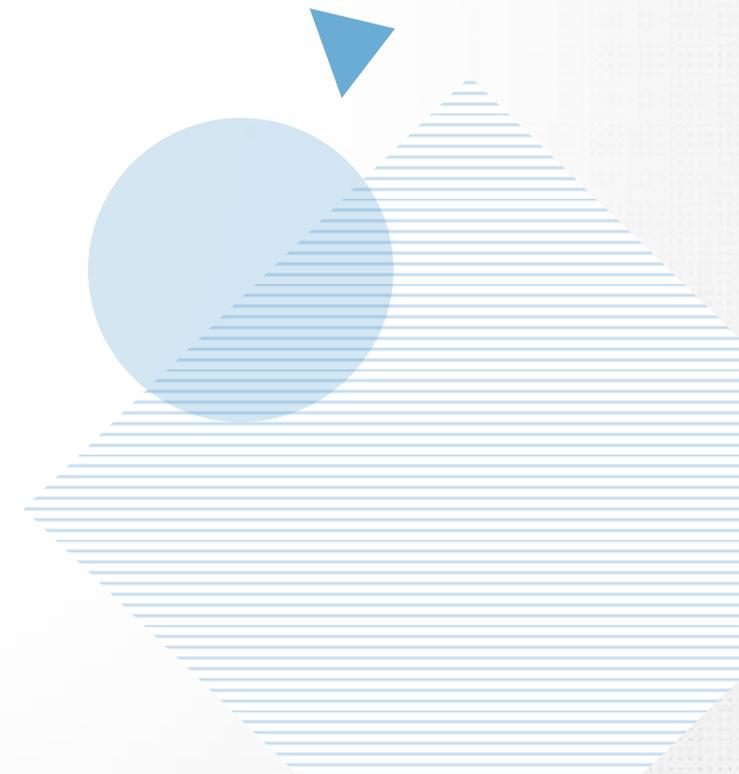
- 통합, 저장, 운영, 공유

## ➤ 데이터베이스 특징

- 실시간접근성, 내용에 의한 참조, 동시 공유, 계속적인 변화

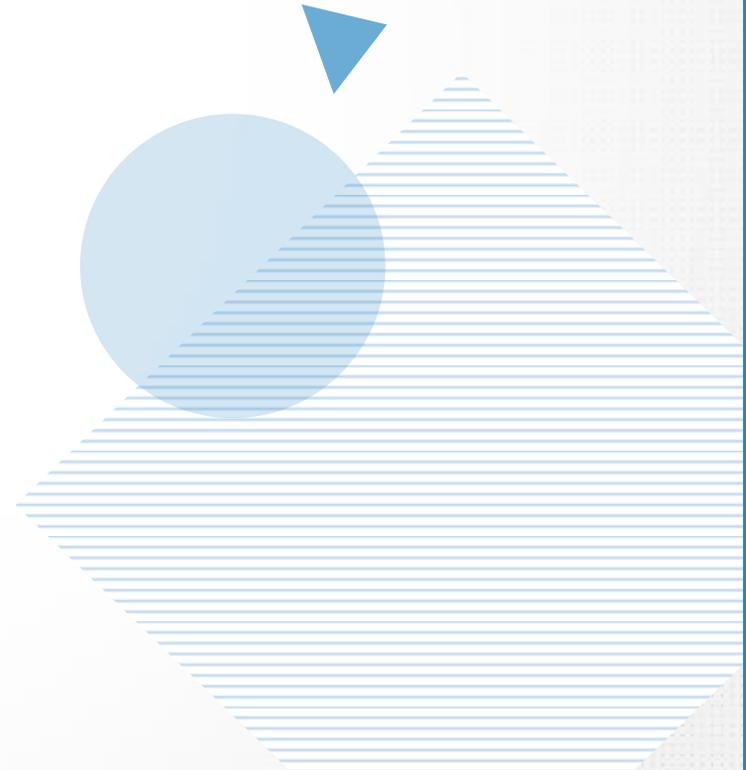
## ➤ DBMS의 장·단점

장점	단점
<ul style="list-style-type: none"><li>• 데이터 중복이 최소화된다.</li><li>• 데이터를 동시 공유할 수 있다.</li><li>• 데이터의 독립성이 확보된다.</li><li>• 데이터의 일관성이 유지된다.</li><li>• 데이터의 부결성이 유지된다.</li><li>• 데이터의 보안이 향상된다.</li><li>• 표준화 할 수 있다.</li></ul>	<ul style="list-style-type: none"><li>• 데이터베이스 전문가가 부족하다.</li><li>• 초기 구축비용이 많이 든다.</li><li>• 서버의 부담이 있다(대용량 디스크의 집중적인 처리로 과부하가 발생한다).</li><li>• 대용량의 기억장치가 필요하다.</li><li>• 백업과 회복방법이 복잡하다.</li></ul>

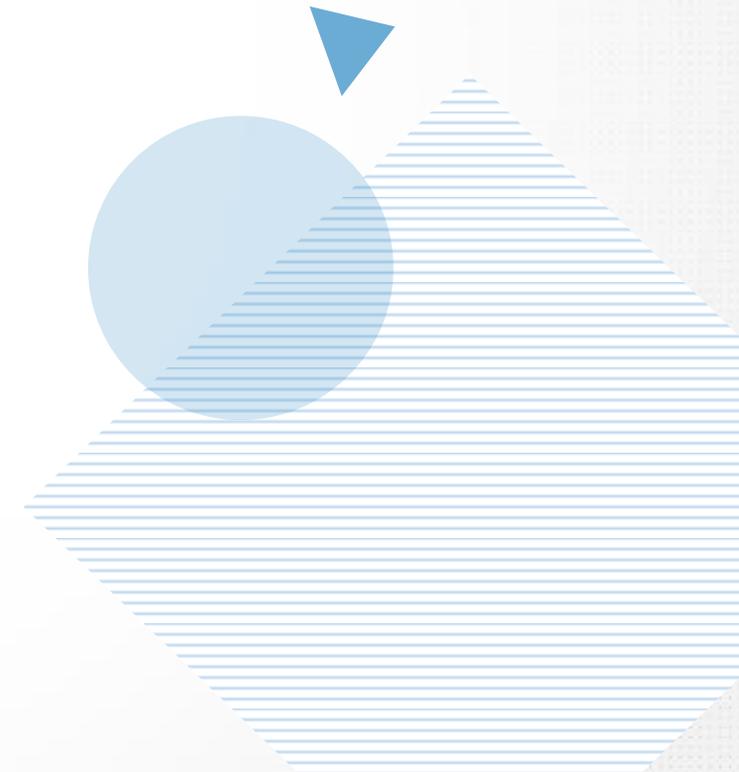


## ➤ DBMS

- 정의기능
- 조작기능
- 제어기능

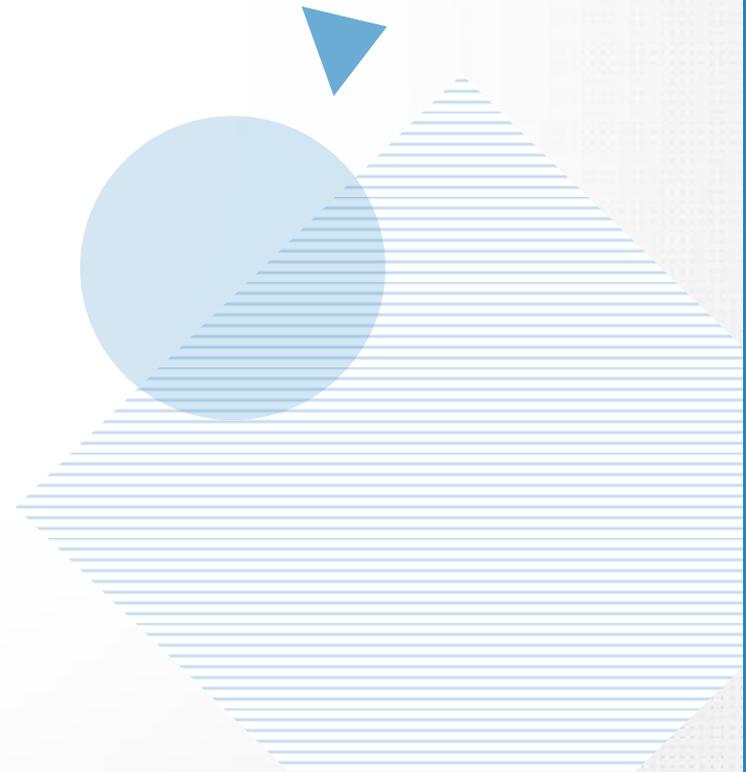


- 정의
- 외부 스키마
- 개념 스키마
- 내부 스키마



## 데이터 베이스 설계

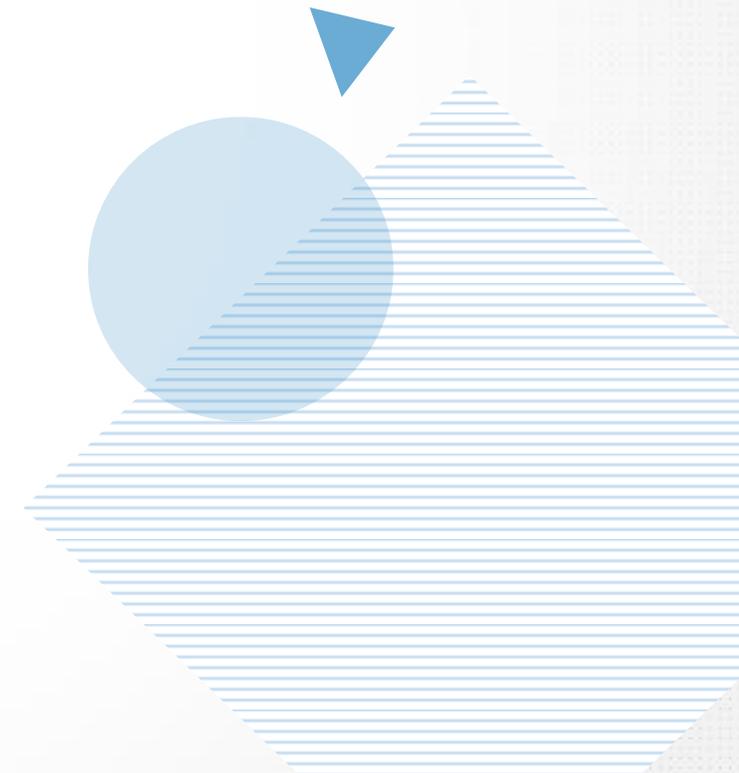
- 요구조건 분석
- 개념적 설계
- 논리적 설계
- 물리적 설계
- 데이터베이스 구현



## 1. 데이터베이스 구성의 장점이 아닌 것은?

- ① 데이터 중복 최소화
- ② 여러 사용자에게 의한 데이터 공유
- ③ 데이터 내용의 일관성 유지
- ④ 데이터 간의 종속성 유지

정답4

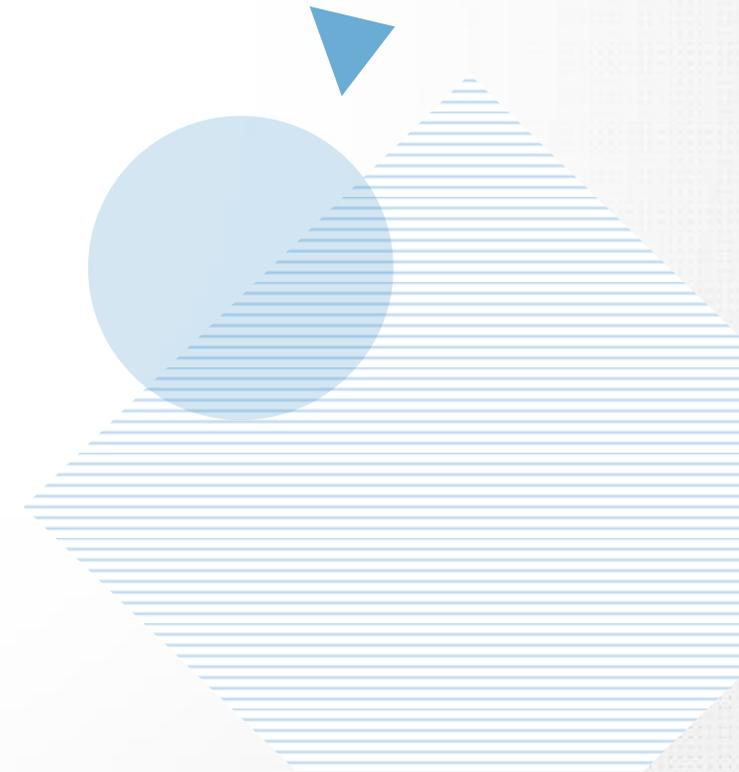


# 제2과목 소프트웨어 개발

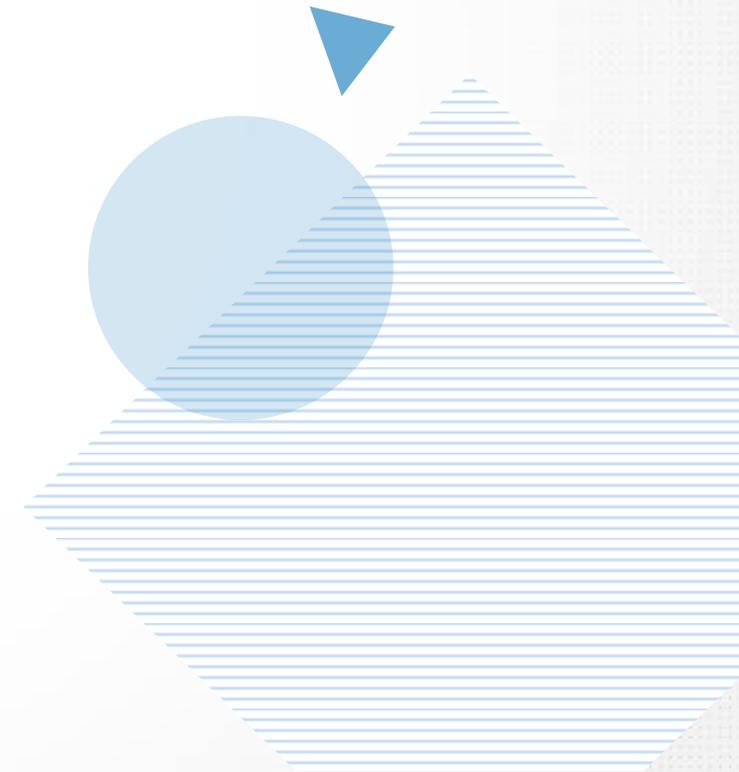
## 04 데이터 입출력 구현 D



- SQL(Structured Query Language)
  - DDL
  - DML
  - DCL



- 데이터 접속(Data Mapping)
  - SQL Mapping
  - ORM
- 트랜잭션(Transaction)



- 절차형 SQL
  - 프로시저
  - 트리거
  - 사용자 정의함수
- 절차형 SQL의 테스트와 디버깅
- 쿼리 성능 최적화

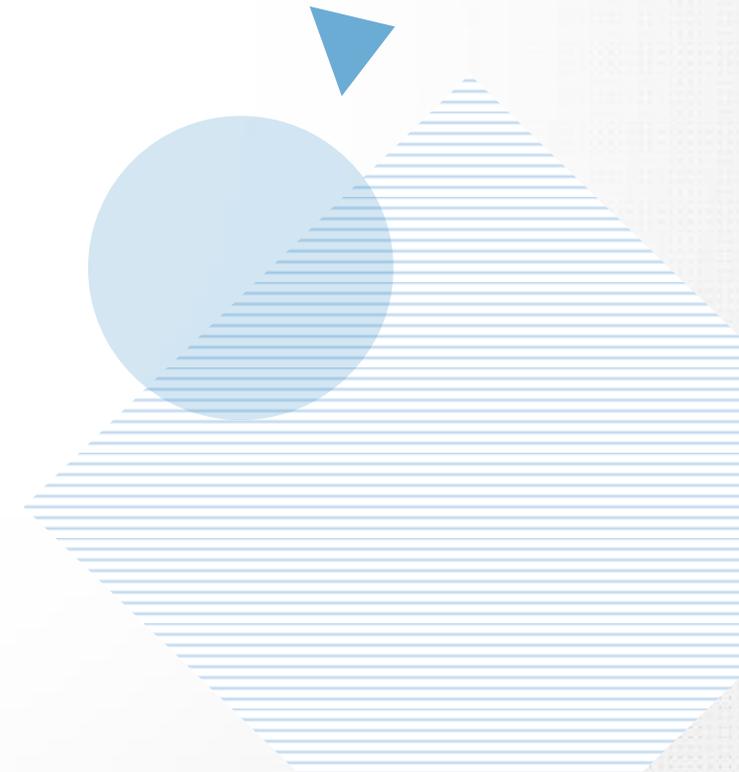


▶ 다음 데이터베이스 트랜잭션의 특성 중 아래 내용에 해당하는 것은?

시스템이 가지고 있는 고정요소는 트랜잭션 수행 전과 트랜잭션 수행 완료 후에 같아야 한다.

- ① 원자성(Automicity)
- ② 일관성(Consistency)
- ③ 격리성(Isolation)
- ④ 영속성(Durability)

정답 2



- 데이터베이스 프로시저 쿼리 성능 최적화를 위해서 개발자가 고려할 사항으로 가장 거리가 먼 것은?
- ① 개발자는 SQL 특성을 충분히 이해하고 SQL문을 적절히 구사할 수 있는 기본적인 능력을 갖추어야 한다.
  - ② 개발자는 SQL 작성 시 옵티마이저의 일련의 행위에 대해서는 몰라도 된다.
  - ③ 구문분석 단계에서 옵티마이저의 실행계획에 따라서 실행 속도의 차이는 크게 날 수 있다.
  - ④ 옵티마이저의 실행계획이 비정상적이라면 개발자는 Hint같은 조건을 부여하여 실행계획을 수정할 수 있다.

정답 2

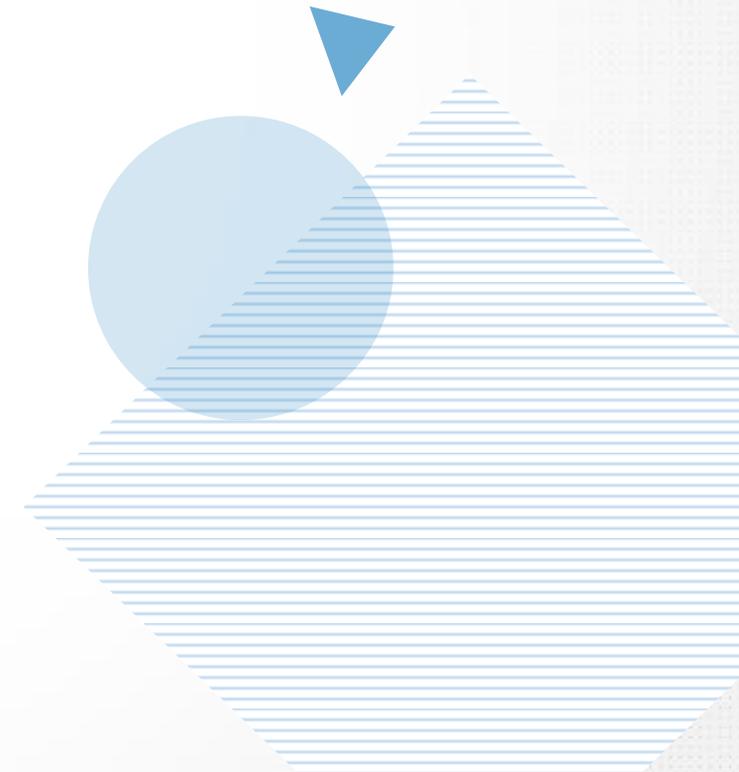
# 제2과목 소프트웨어 개발

## 05 통합 구현



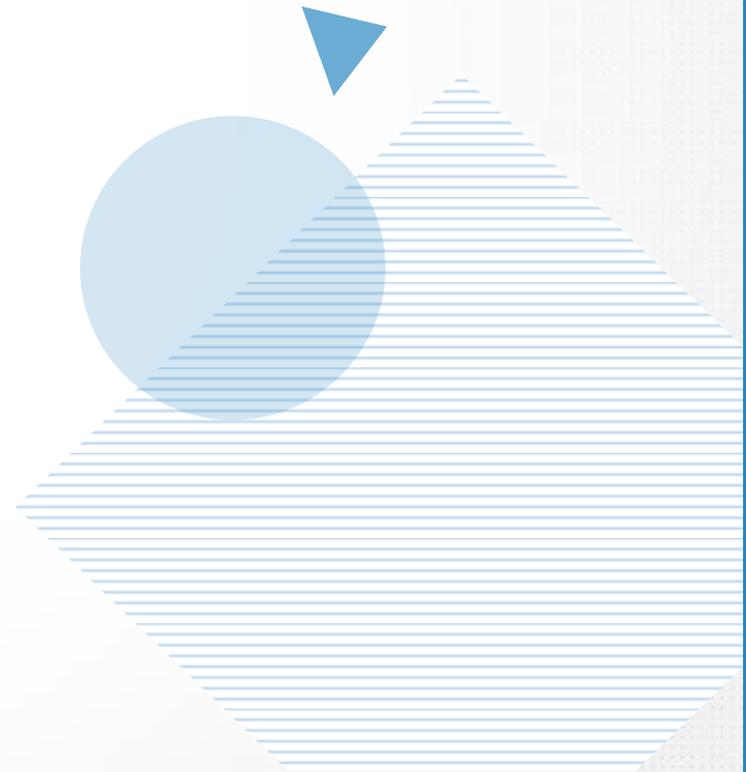
## ③ 단위 모듈 구현

- 단위 모듈(Unit Module)의 개요
  - 단위기능 명세서 작성 - 입출력기능 구현 - 알고리즘 구현
- 단위 기능 명세서 작성
- 입·출력 기능 구현



## ➤ 알고리즘 구현

- 디바이스 드라이버 모듈
- 네트워크 모듈
- 파일 모듈
- 메모리 모듈
- 프로세스 모듈



## ▶ 단위 모듈 테스트

- 화이트 박스 테스트
- 블랙박스 테스트

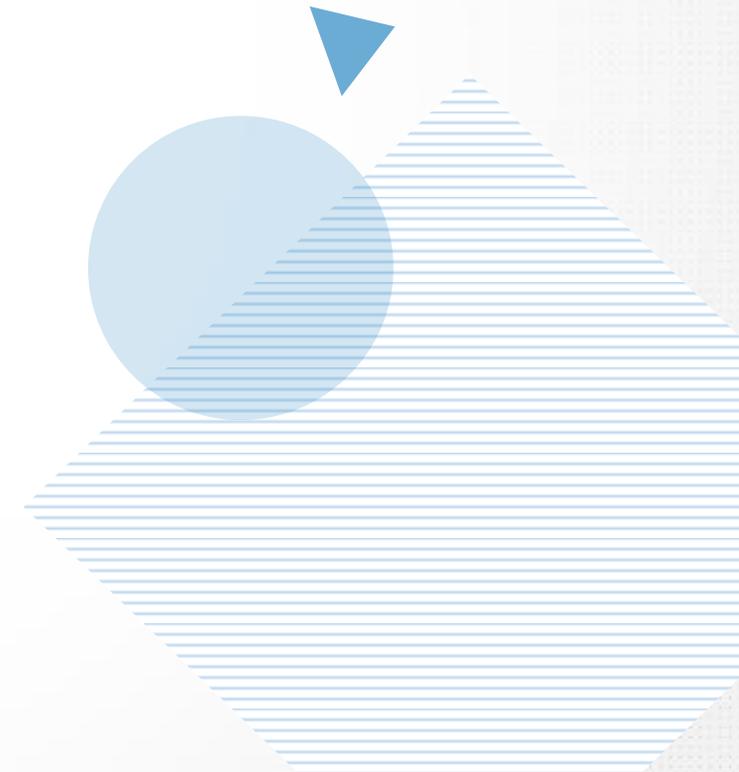
## ▶ 테스트 케이스(Test Case)

- 식별자, 텍스트 항목, 입력 명세, 출력명세, 환경설정, 특수절차요구, 의존성기술

## ▶ 테스트 프로세스

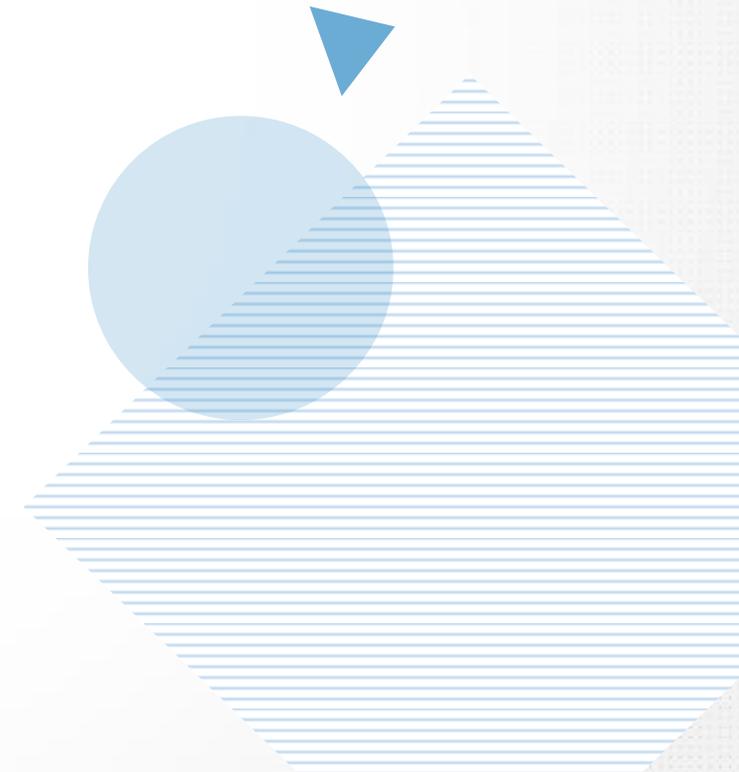
- 계획및 제어단계, 분석 및 설계단계, 구현 및 실현단계, 평가단계, 완료단계

- 통합 개발 환경(IDE; Integrated Development Environment)
  
- 빌드 도구
  - Ant
  - Maven
  - Gradle



## ➤ 기타 협업 도구

- 프로젝트 및 일정관리
- 정보 공유 및 커뮤니케이션
- 디자인
- 기타



- ▶ 다음 중 단위모듈 구현 시 고려할 사항으로 틀린 것은?
- ① 응집도는 높고, 결합도는 낮추는 방향으로 구현을 한다.
  - ② 개별 모듈을 먼저 구현 후 공통 모듈은 맨 나중에 구현한다.
  - ③ 단위모듈의 구현 시 디버깅은 IDE도구를 활용하여서 수행한다.
  - ④ 단위모듈의 테스트는 화면(UI)이 있다면 화면 중심으로 테스트 한다.

## 정답 2

단위 모듈 구현시 공통 모듈을 먼저 구현하고 이를 재사용하여 각각의 단위 모듈을 구현한다.



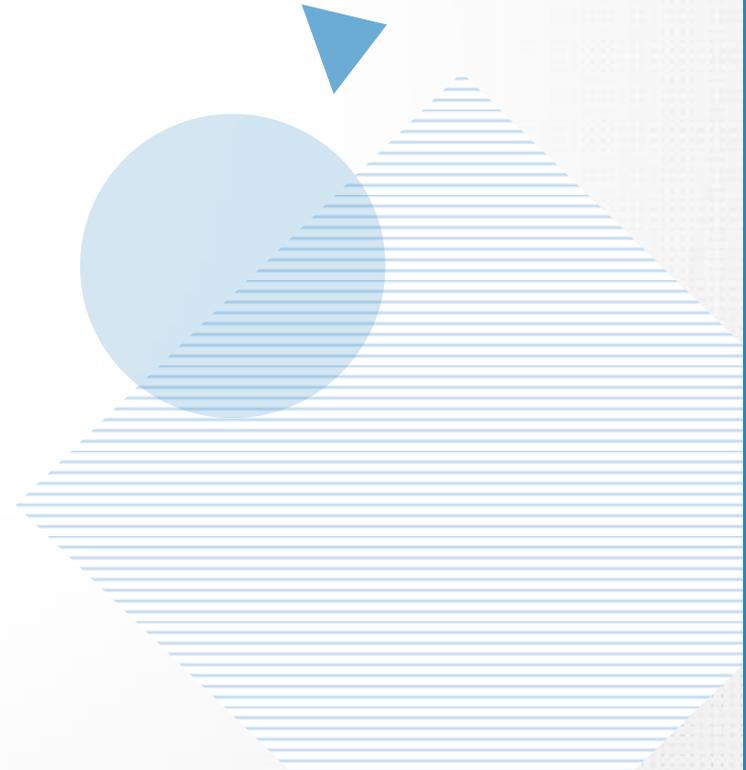
# 제2과목 소프트웨어 개발

## 06 제품 소프트웨어 패키징A



## 소프트웨어 패키징

- 소프트웨어 패키징의 개요
- 패키징 시 고려사항
- 패키징 작업 순서
  - 온라인 배포
  - 오프라인 배포

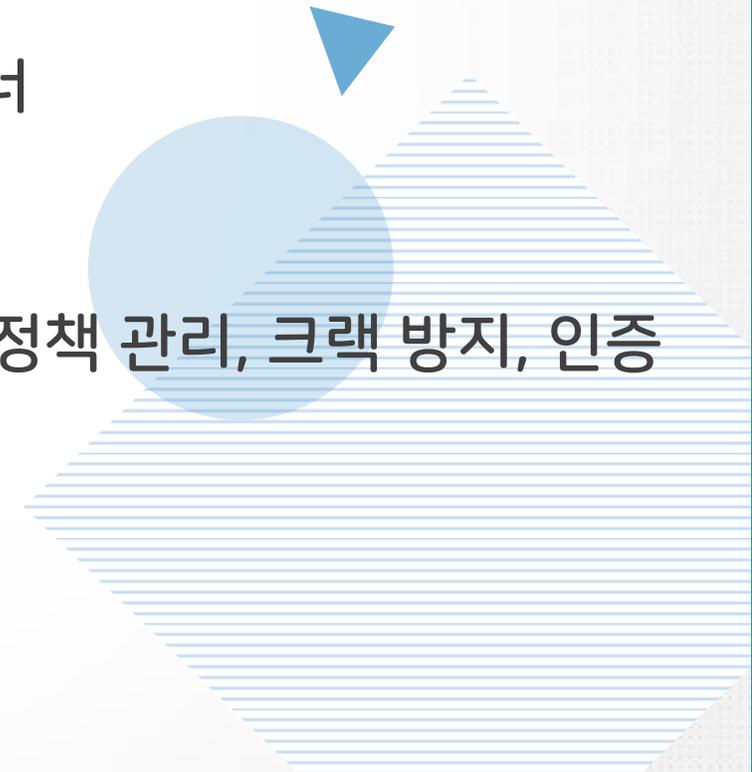


- 릴리즈 노트(Release Note)의 개요
- 릴리즈 노트 초기 버전 작성 시 고려사항
  - 머리말, 개요, 목적, 문제 요약, 재현항목, 수정/개선 내용, 사용자 영향도, SW 지원 영향도, 노트, 면책 조항, 연락처
- 릴리즈 노트 추가 버전 작성 시 고려사항
- 릴리즈 노트 작성 순서
  - 모듈식별-릴리지 정보 확인-릴리즈 노트 개요 작성-영향도 체크-정식 릴리즈 노트 작성- 추가 개선 항목 식별



# 🔗 디지털 저작권 관리(DRM)

- 저작권의 개요
- 디지털 저작권 관리(DRM; Digital Right Management)의 개요
- 디지털 저작권 관리의 흐름도
  - 클리어링 하우스, 콘텐츠 제공자, 패키지
  - 콘텐츠 분배자, 콘텐츠 소비자, DRM 컨트롤러, 보안컨테이너
- 디지털 저작권 관리의 기술 요소
  - 암호화, 키 관리, 암호화 파일 생성, 식별 기술, 저작권 표현, 정책 관리, 크랙 방지, 인증



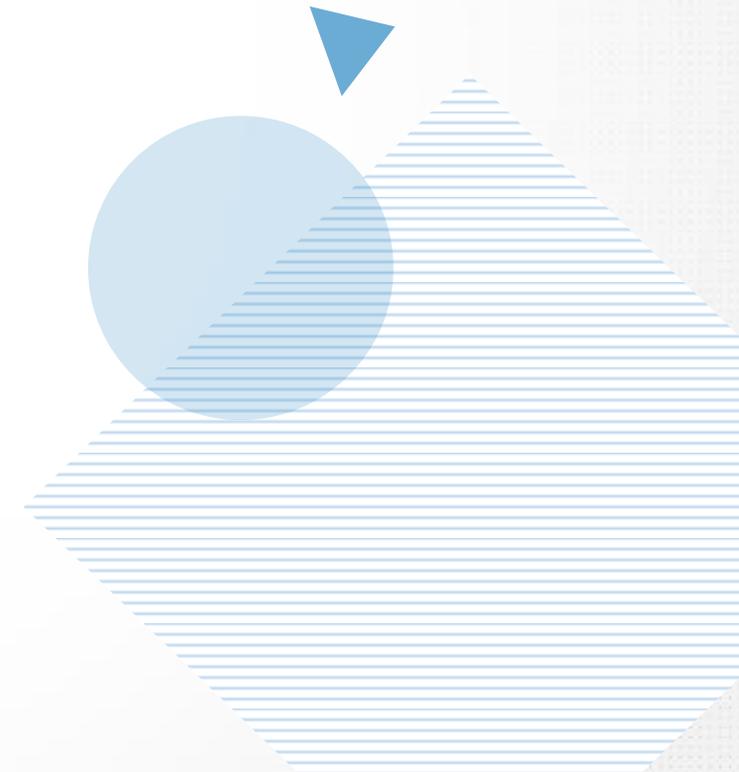
- ▶ 애플리케이션 패키징 시 고려할 사항으로 가장 거리가 먼 것은?
- ① 사용자 운영체제, 시스템 사양(CPU, Memory, Disk) 등 최소 사용 환경을 고려한다.
  - ② 직관적인 화면을 고려하여 매뉴얼과 일치시켜 패키징 작업을 한다.
  - ③ 애플리케이션 패키징 변경 및 개선 관리를 위하여 변경 사항을 기록한다.
  - ④ 애플리케이션 개발자의 개발환경, 성향, 유지보수 일정 등을 고려하여 패키징한다.

정답 4

▶ 다음 중 디지털 저작권 관리의 기술 요소가 아닌것은 ?

- ① 암호화
- ② 식별 기술
- ③ 방화벽
- ④ 정책 관리

정답 3



# 제2과목 소프트웨어 개발

## 07 제품 소프트웨어 패키징B



# 🔗 소프트웨어 설치 매뉴얼

- 소프트웨어 설치 매뉴얼의 개요
- 서문
  - 문서 이력, 설치 매뉴얼의 주석, 설치도구의 구성, 설치 환경 체크 항목
- 기본 사항
  - 소프트웨어 개요, 설치 관련 파일, 설치 아이콘, 프로그램 삭제, 관련 추가정보

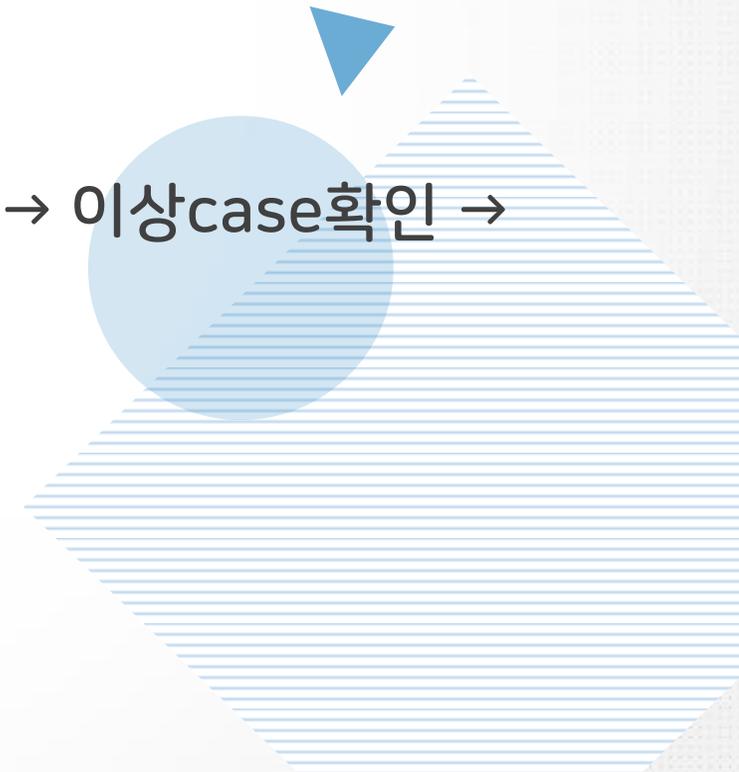


### ➤ 설치 매뉴얼 작성 방법

- 설치화면, 설치이상 메시지 설명, 설치 완료 및 결과, FAQ
- 설치 시 점검 사항, 네트워크 환경 및 보안, 고객 지원 방법, 준수 정보 & 제한 보증

### ➤ 설치 매뉴얼 작성 순서

- 기능 식별 → UI분류 → 설치파일확인 → Uninstall절차 확인 → 이상case확인 → 최종 매뉴얼 적용



- 소프트웨어 사용자 매뉴얼의 개요
- 서문
  - 문서 이력, 설치 매뉴얼의 주석, 기록 보관 내용
- 기본 사항
  - 소프트웨어 개요, 소프트웨어 사용 환경, 소프트웨어 관리, 모델버전별 특징, 기능 인터페이스 특징, 소프트웨어 구동 환경

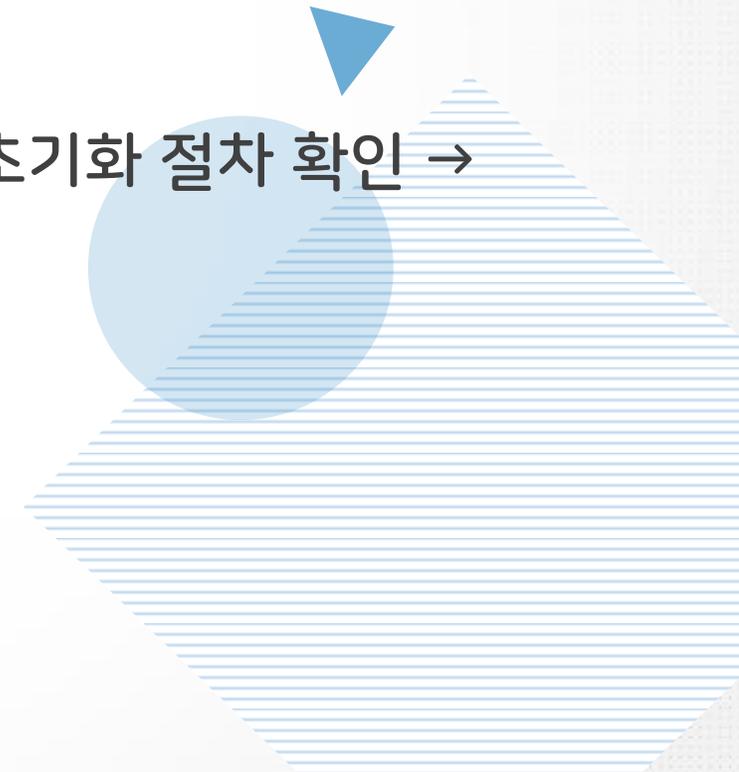


## ➤ 사용자 매뉴얼 작성 방법

- 사용자화면, 주요기능 분류, 응용 프로그램 및 설정, 장치 연동, 네트워크 환경, Profile 안내, 고객 지원 방법, 준수정보 제한 보증

## ➤ 사용자 매뉴얼 작성 순서

- 기능 식별 → 사용자 화면 분류 → 사용자 환경 파일 확인 → 초기화 절차 확인 → 이상case확인 → 최종 매뉴얼 적용



## ➤ 국제 표준 제품 관련 품질

### ▪ 국제 표준 제품 관련 품질 특성의 분류

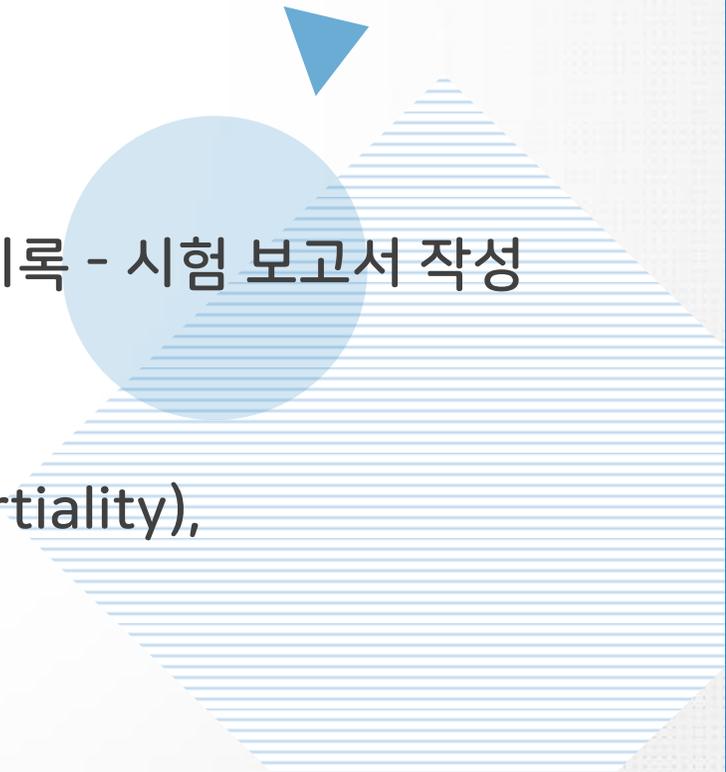
- 제품 품질 표준 : ISO/IEC 9123, ISO/IEC 14598, ISO/IEC 12119, ISO/IEC 25000
- 프로세스 품질 표준 : ISO/IEC 9000, ISO/IEC 12207, ISO/IEC 155054, ISO/IEC 15288, CMMI

### ▪ ISO/IEC 12119 평가 절차

- 제품 설명서 시험 - 사용자 문서 지침 - 실행 프로그램 시험 - 시험 기록 - 시험 보고서 작성

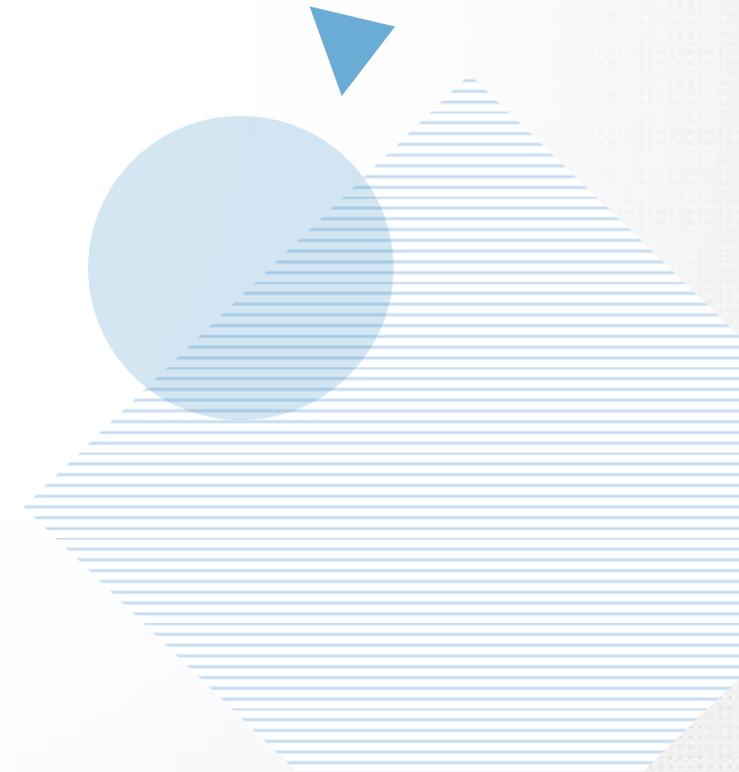
### ▪ ISO/IEC 14598 특징

- 반복성(Repeatability), 재현성(Reproducibility), 공정성(Impartiality), 객관성(Objectivity)



- ▶ 다음 중 제품소프트웨어 설치 매뉴얼에 들어갈 내용으로 가장 거리가 먼 것은?
- ① 제품 소프트웨어 사용 절차 및 사용 화면
  - ② 설치를 위한 환경 체크 항목(사용자 PC환경, 업그레이드 버전, 설치 폴더 등)
  - ③ 설치 파일(exe/dll/ini/chm 등) 및 설치 위치
  - ④ 문서 이력 정보

정답 1



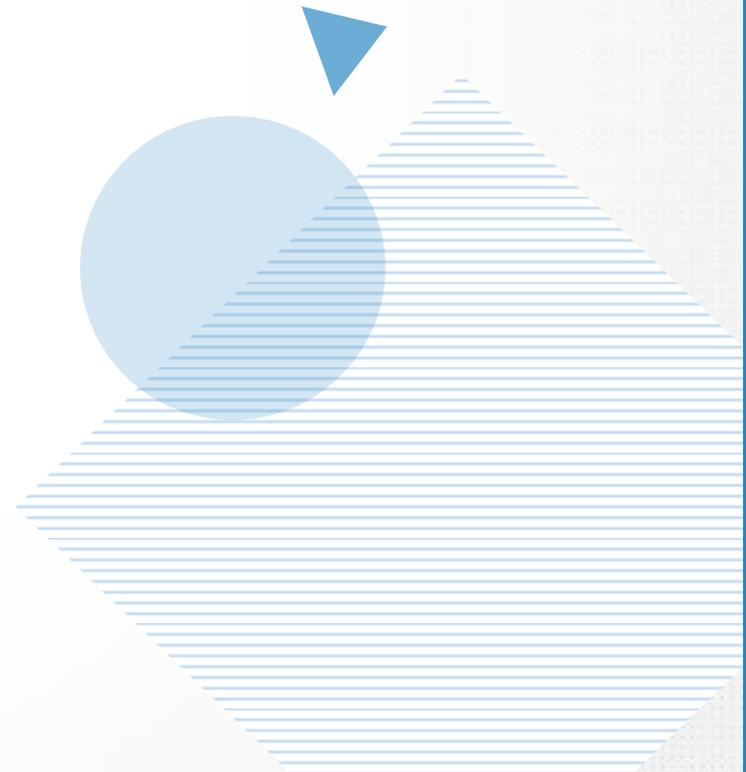
# 제2과목 소프트웨어 개발

## 08 제품 소프트웨어 패키징C



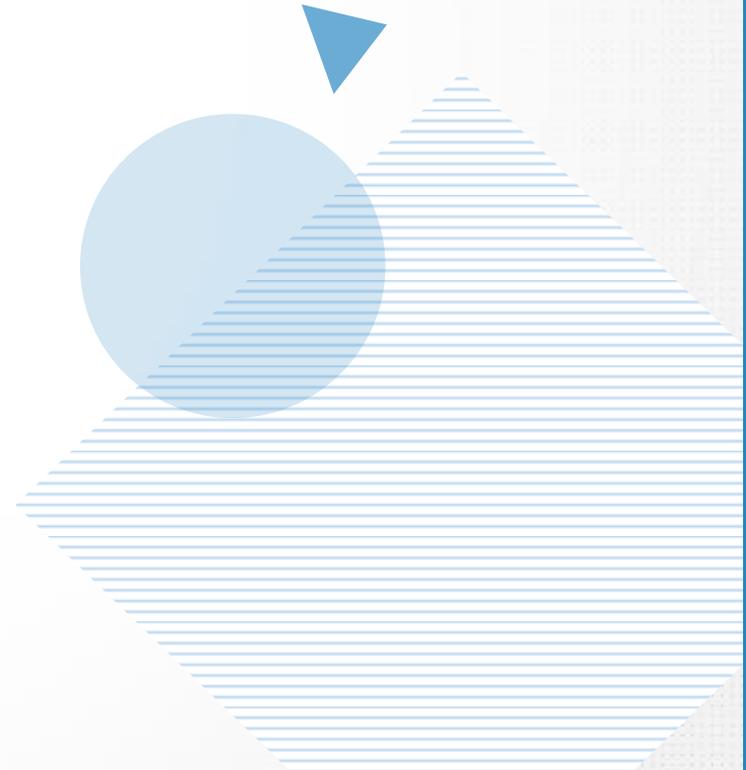
## 소프트웨어 버전 등록

- 소프트웨어 패키징의 형상 관리
- 형상관리의 중요성
- 형상 관리 기능
  - 형상 식별, 버전 제어, 형상 통제, 형상 감사, 형상 기록



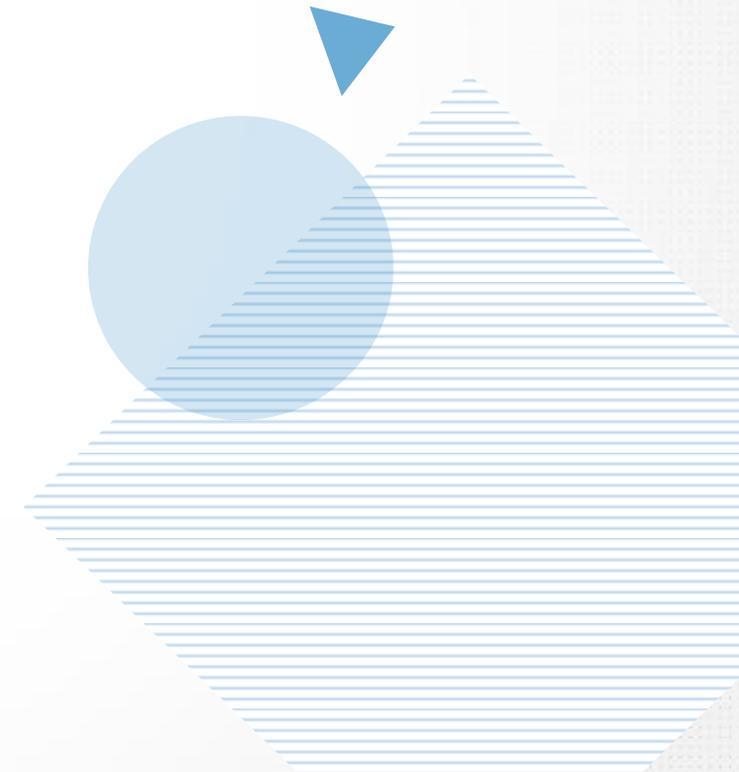
## 소프트웨어 버전 등록

- 소프트웨어의 버전 등록 관련 주요 용어
  - 저장소, 가져오기, 체크아웃, 체크인, 커밋, 동기화
  
- 소프트웨어 버전 등록 과정
  - 가져오기 → 인출 → 예치 → 동기화 → 차이



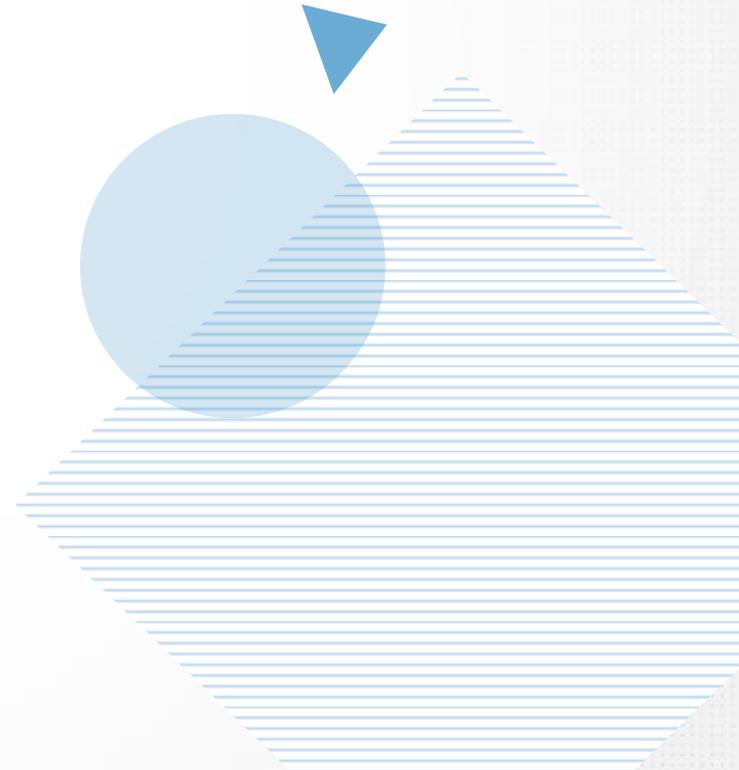
## ➤ 소프트웨어 버전관리 도구 개요

- 소프트웨어 버전관리 도구란?
  - 제품소프트웨어의 신규 개발, 변경, 개선과 관련된 수정 내역을 관리하는 도구
- 소프트웨어 버전관리 도구 유형
  - 공유 폴더 방식
  - 클라이언트/서버 방식
  - 분산 저장소 방식



## ◆ 소프트웨어 버전관리 도구별 특징

- CVS,
- SVN,
- RCS,
- Bitkeeper,
- Git,
- Clear Case



## 🔄 소프트웨어 버전관리 도구

### ➤ 소프트웨어 버전관리 도구 사용 시 유의점

- 형상관리 지침에 의거 버전에 대한 정보를 언제든지 접근할 수 있어야 함
- 개발자, 배포자 이외에 불필요한 사용자가 소스 접근할 수 없도록 해야 함
- 동일한 프로젝트 파일에 대해서 여러 개발자가 동시 개발할 수 있어야 함
- 에러 발생 시 최대한 빠른 시간 내에 복구 필요

### ➤ 제품소프트웨어에 소프트웨어 버전관리 도구 활용방안

- 공동 개발 및 작업 관리
- 버전 백업 및 복구
- 여러 버전 솔루션 작업



## 빌드 자동화 도구

### ➤ 소프트웨어 빌드 자동화 도구 개요

- 소프트웨어 빌드 자동화 도구란? 저장소에 있는 소스를 자동으로 읽어서 빌드를 하여 실행 파일을 만드는 도구

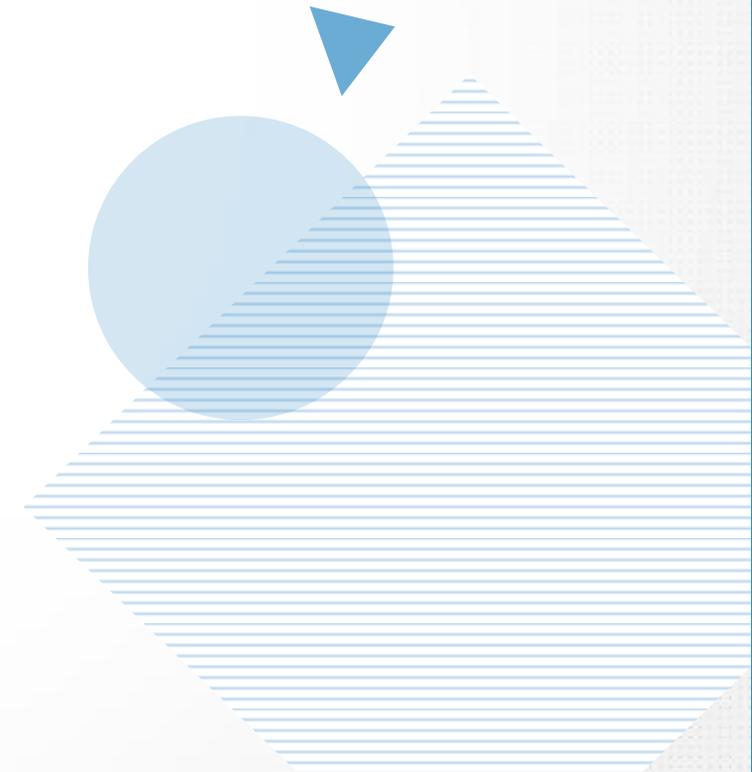
### ➤ 온라인 빌드 자동화 도구, 젠킨스(Jenkins)

- 빌드 자동화 도구로서 가장 많이 활용되는 도구이다.
- 지속적 통합관리를 가능하게 한다, 다양한 버전관리 도구를 지원한다.
- 임의의 셸 스크립트와 윈도우 배치 명령까지 실행시킬 수 있다.

## 🔗 빌드 자동화 도구

### ➤ 안드로이드 환경에 적합한 도구, 그레들(Gradle)

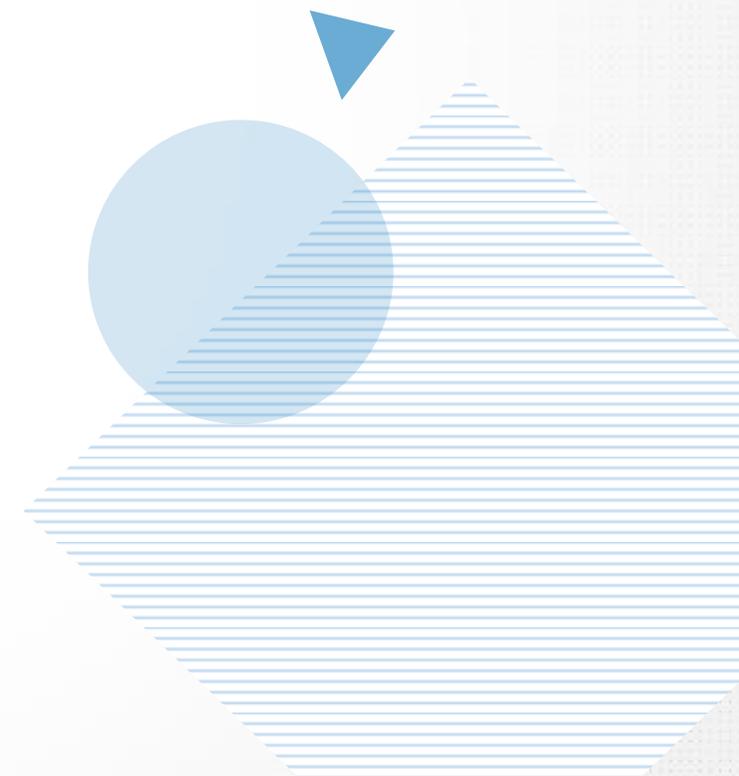
- 그레들(Gradle)은 여러가지 언어의 빌드 환경을 구성할 수 있다.
- 안드로이드 개발 환경에서 빌드 자동화 도구로 사용된다.
- 그레들 스크립트는 groovy를 사용해서 만든 DSL이다.
- 모든 그레들 빌드는 하나 이상의 projects로 구성된다.



▶ 다음 중 소프트웨어 버전관리 도구의 기능으로 가장 거리가 먼 것은?

- ① 소스 코드 형상 관리
- ② 소스 오류 관리
- ③ 소스 원복 관리
- ④ 소스 백업 관리

정답 2



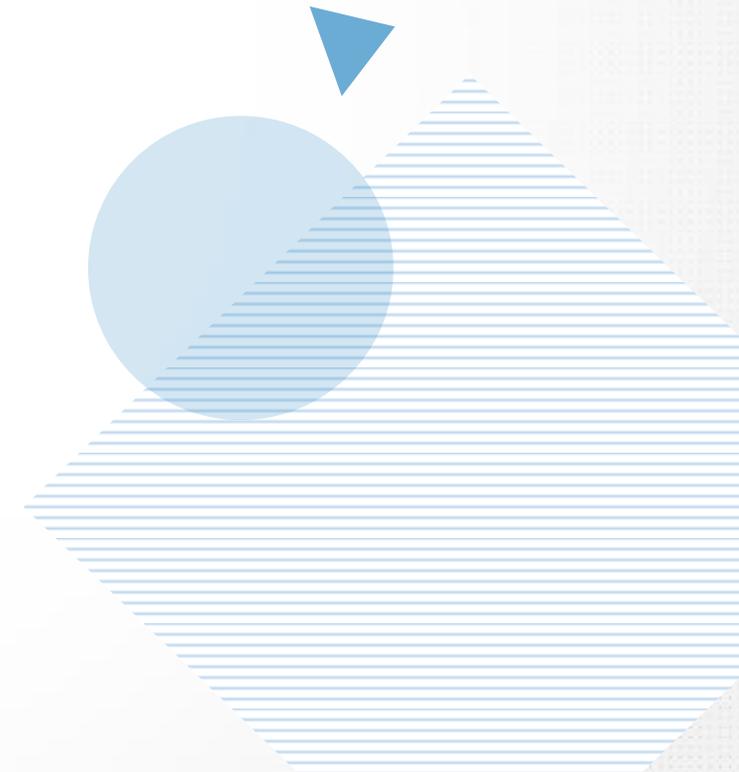
# 제2과목 소프트웨어 개발

## 09 애플리케이션 테스트 관리 A



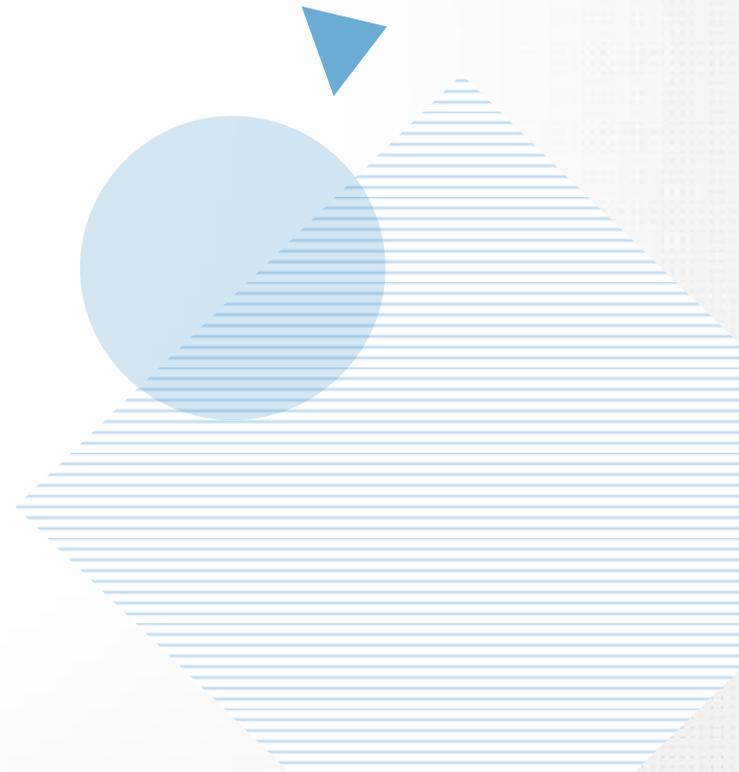
## 애플리케이션 테스트

- 애플리케이션 테스트의 개념
- 애플리케이션 테스트의 필요성
- 애플리케이션 테스트의 기본 원리



## 🔄 애플리케이션 테스트의 분류

- 프로그램 실행 여부에 따른 테스트
  - 정적 테스트
  - 동적 테스트
- 테스트 기반(Test Bases)에 따른 테스트
  - 명세 기반 테스트
  - 구조 기반 테스트
  - 경험 기반 테스트



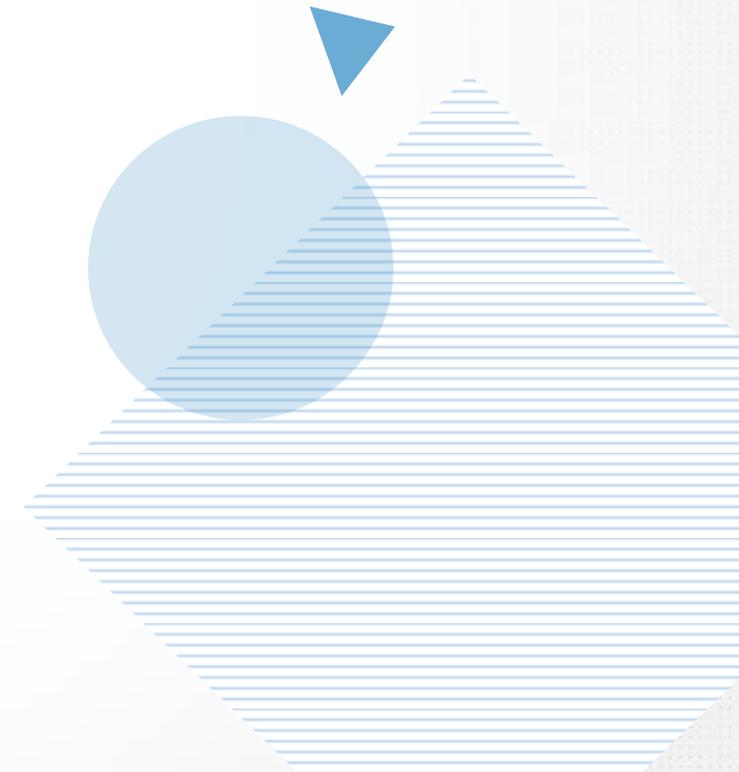
# 애플리케이션 테스트의 분류

## ▶ 시각에 따른 테스트

- 검증 테스트
- 확인 테스트

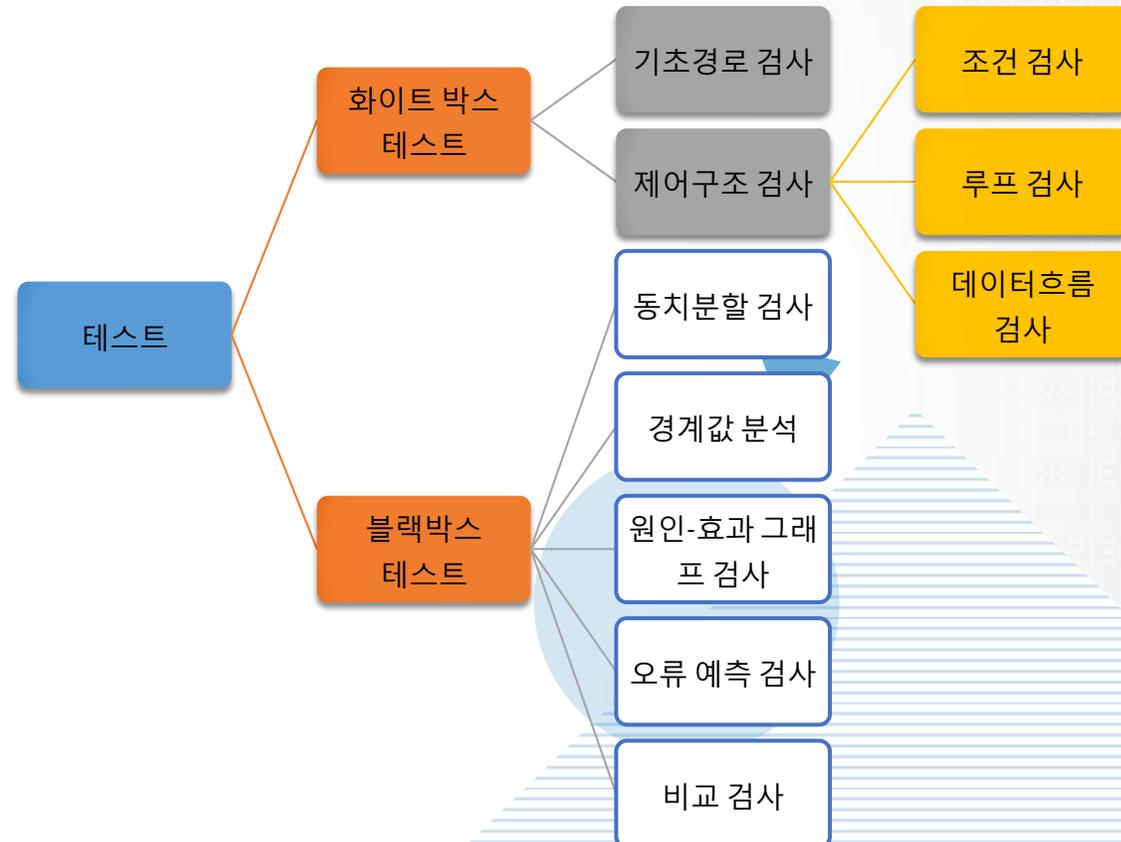
## ▶ 목적에 따른 테스트

- 회복 테스트, 안전 테스트, 강도 테스트
- 성능 테스트, 구조 테스트, 회귀 테스트
- 병행 테스트



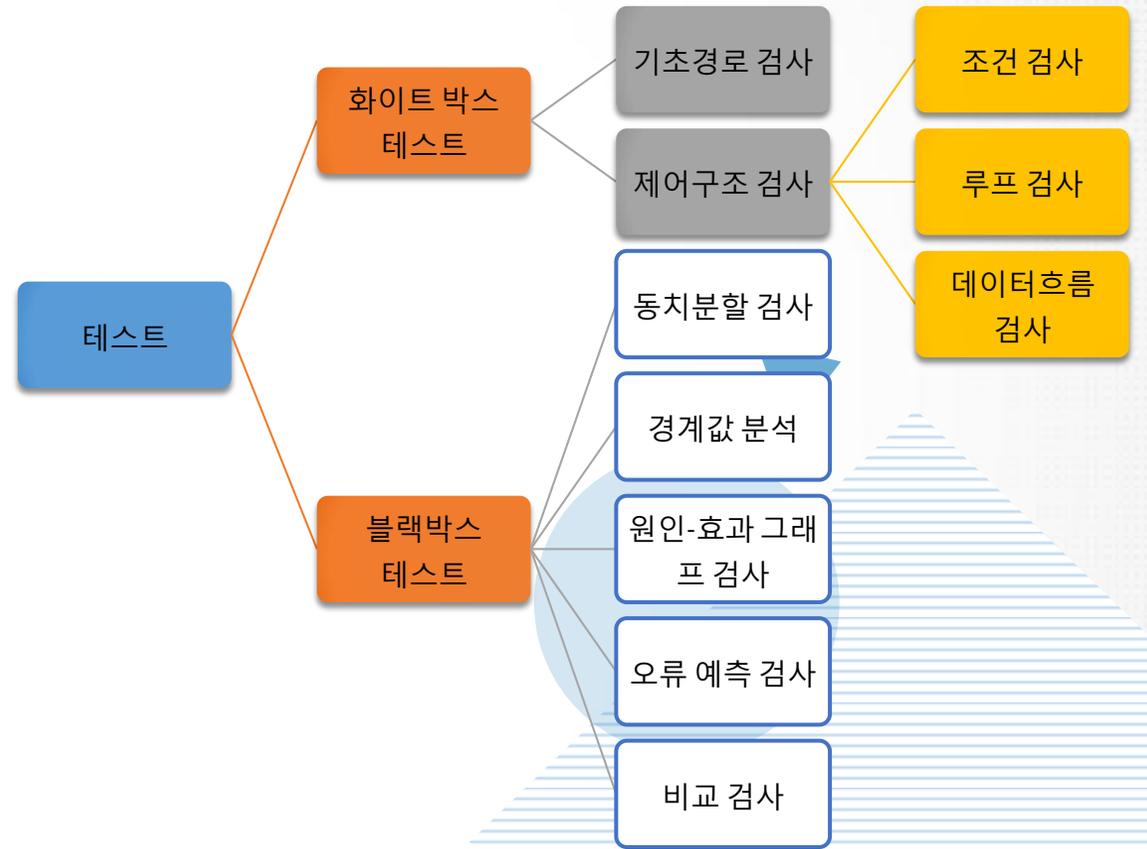
# 테스트 기법에 따른 애플리케이션 테스트

## 화이트박스 테스트(White Box Test)



# 테스트 기법에 따른 어플리케이션 테스트

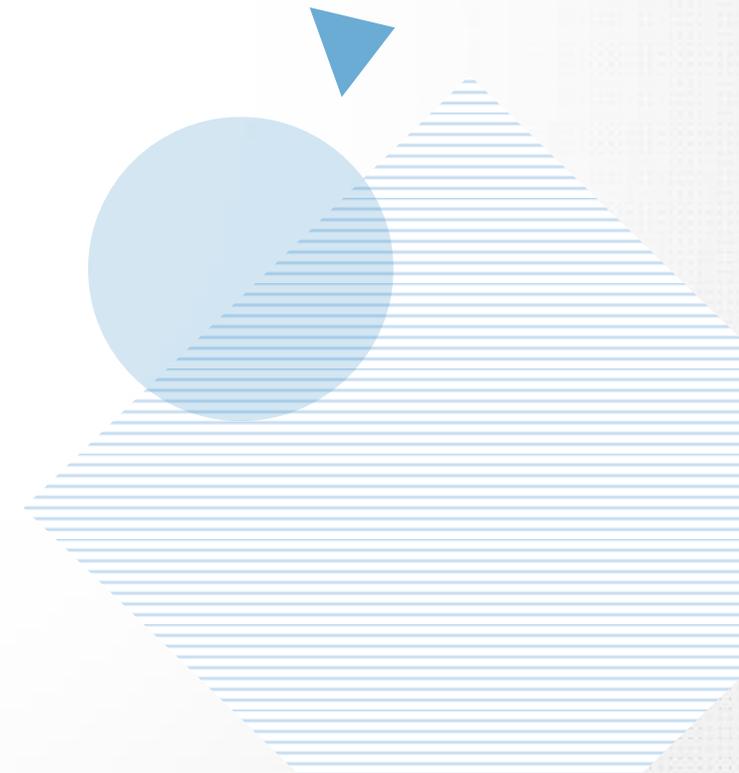
## ▶ 블랙박스 테스트(Black Box Test)



# 1. 블랙 박스 테스트로 발견하기 어려운 오류는?

- ① 인터페이스 오류
- ② 행위나 성능 오류
- ③ 잘못되거나 누락된 기능
- ④ 모듈 내의 논리 구조상의 오류

정답 4



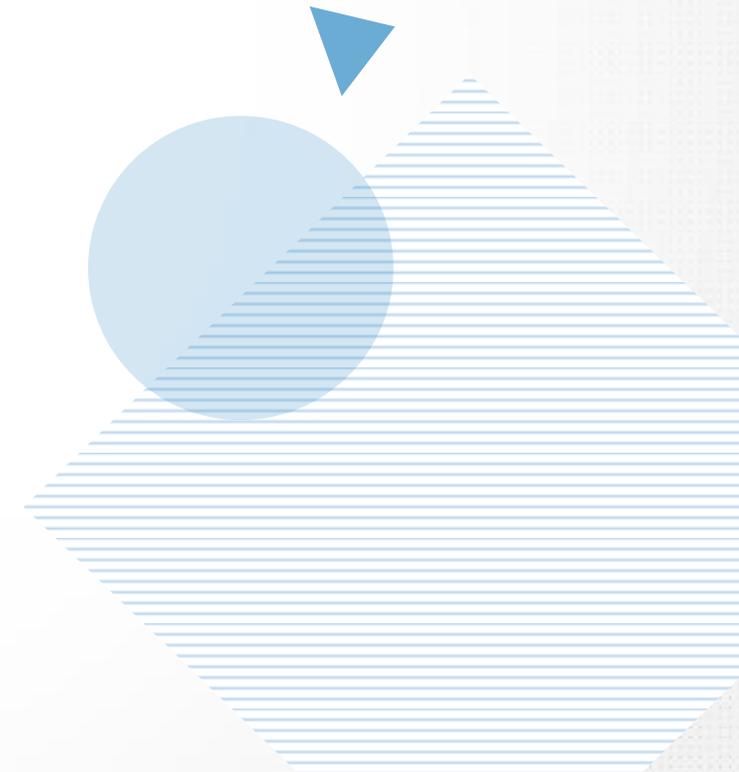
# 제2과목 소프트웨어 개발

## 10 애플리케이션 테스트 관리 B



## 개발 단계에 따른 애플리케이션 테스트

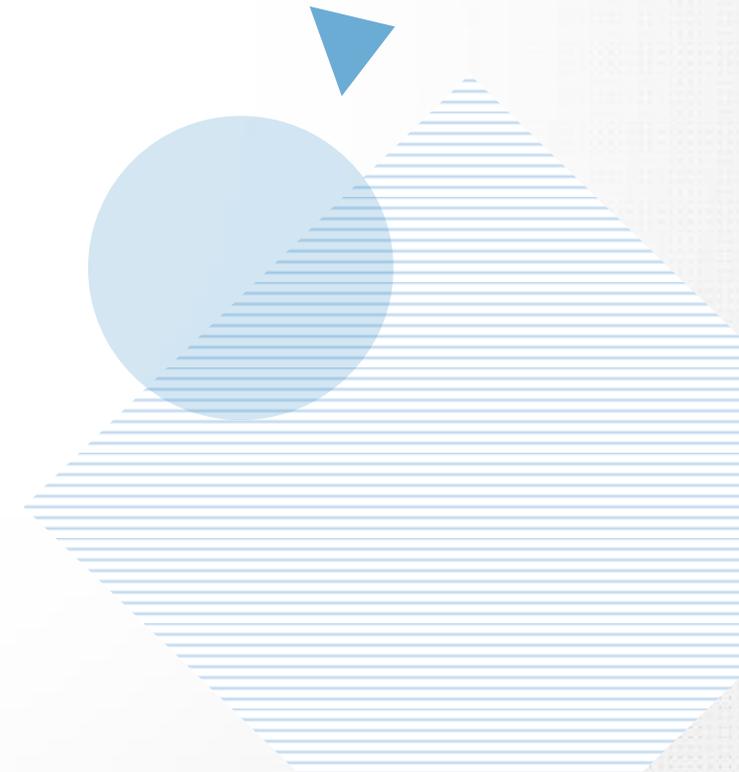
- 단위 테스트(Unit Test)
- 통합 테스트(Integration Test)
- 검증 테스트
- 시스템 테스트



# 개발 단계에 따른 애플리케이션 테스트

## ▶ 통합 테스트(Integration Test)

- 비점진적 통합방식
  - 빅뱅 통합검사
- 점진적 통합방식
  - 하향식 통합 검사
  - 상향식 통합검사
  - 혼합식 통합검사



# 개발 단계에 따른 애플리케이션 테스트

## ▶ 검증 테스트

- 형상 검사
- 알파 검사
- 베타 검사

## ▶ 시스템 테스트

- 복구 검사
- 보안 검사
- 강도 검사
- 성능 검사



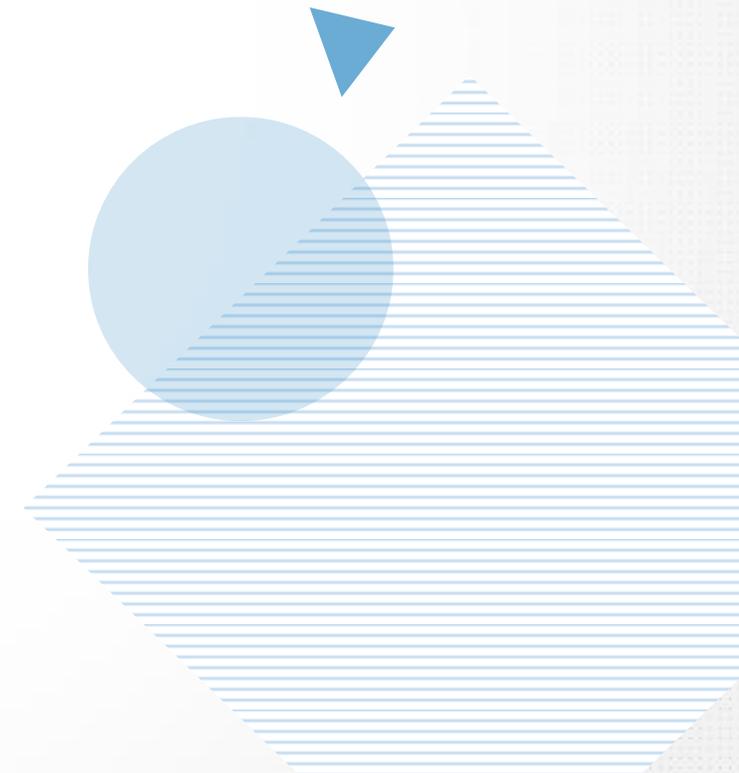
# 애플리케이션 테스트 프로세스

## ➤ 애플리케이션 테스트 프로세스

- 테스트 계획 → 테스트 분석 및 디자인 → 테스트 케이스 및 시나리오 작성 → 테스트 수행-테스트 결과 평가 및 리포팅 → 결함추적 및 관리
- 테스트 계획서, 테스트 케이스, 테스트 시나리오, 테스트 결과서

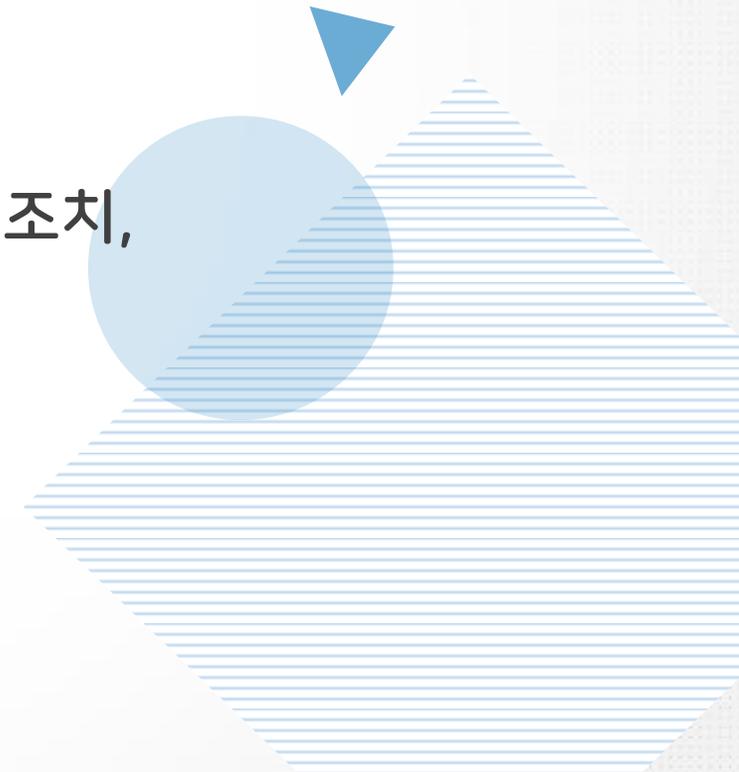
## ➤ 테스트 계획

- 테스트 시작 조건
- 테스트 종료조건

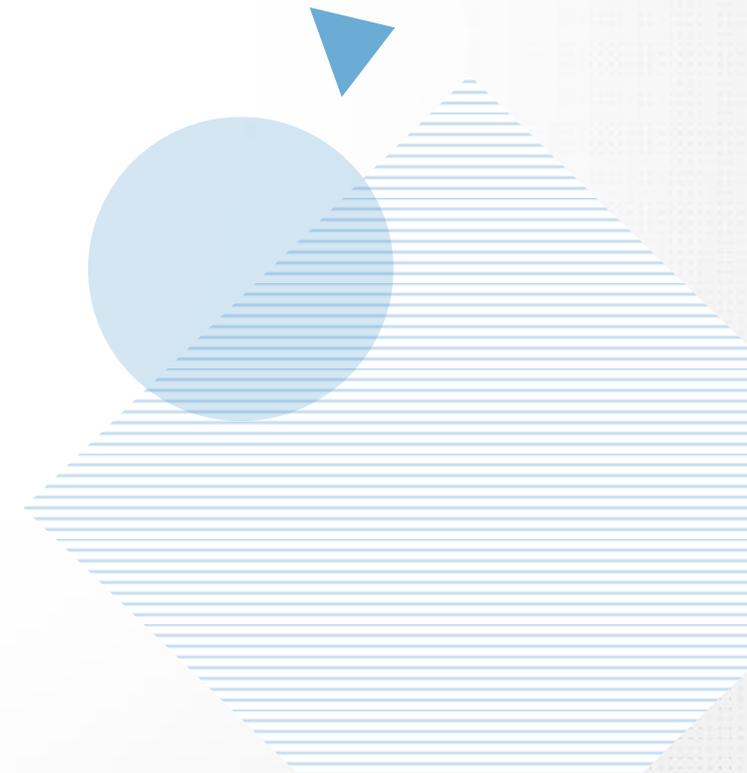


## 애플리케이션 테스트 프로세스

- 테스트 분석 및 디자인
- 테스트 케이스 및 시나리오 작성
- 테스트 수행
- 테스트 결과 평가 및 리포팅
- 결함 추적 및 관리
  - 에러 발견, 에러 등록, 에러 분석, 결함 확정, 결함 할당, 결함 조치, 결함 조치 검토 및 승인

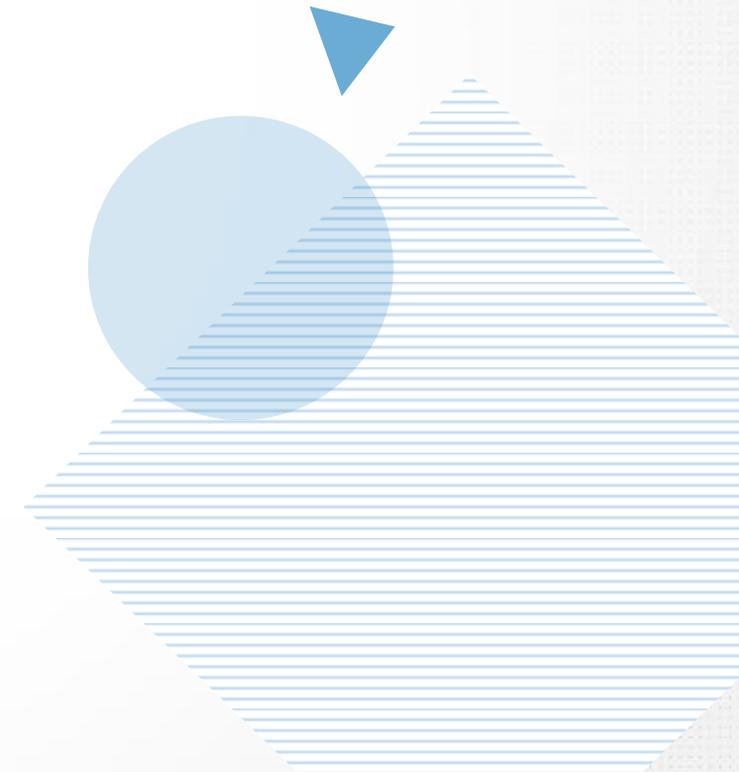


- 테스트 케이스(Test Case)
- 테스트 케이스 작성 순서
  - ① 테스트 계획 검토 및 우선순위 결정
  - ② 위험 평가 및 우선 순위 결정
  - ③ 테스트 요구사항 정의
  - ④ 테스트 구조 설계 및 테스트 방법결정
  - ⑤ 테스트 케이스 정의
  - ⑥ 테스트 케이스 타당성 확인및 유지보수



## 테스트 케이스/테스트 시나리오/테스트 오라클

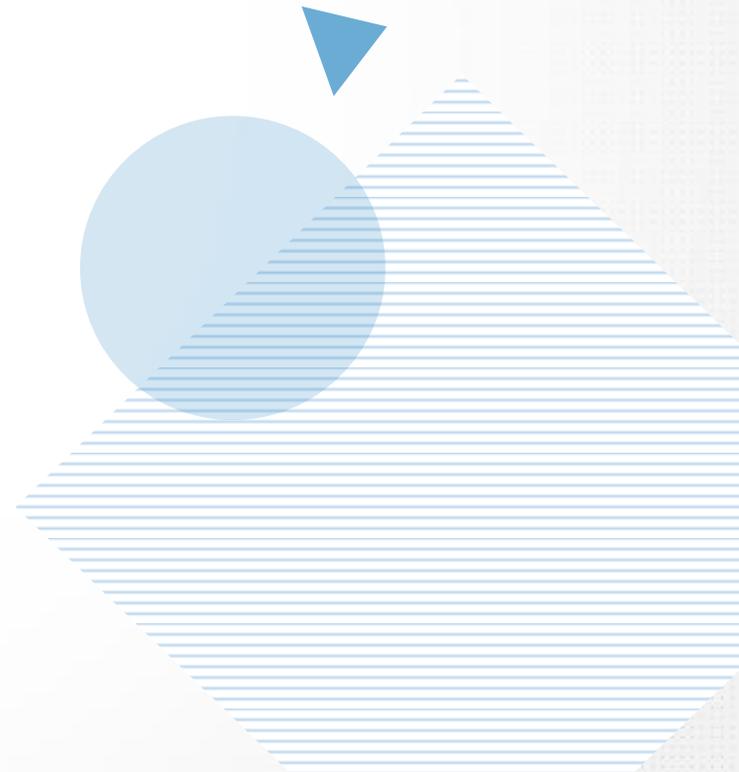
- 테스트 시나리오(Test Scenario)
- 테스트 시나리오 작성 시 유의 사항
- 테스트 오라클(Test Oracle)
  - 제한된 검증, 수학적 기법, 자동화 가능
- 테스트 오라클의 종류
  - 참 오라클, 샘플링 오라클, 추정 오라클, 일관성 검사 오라클



1. 다음 중 테스트 오라클의 종류가 아닌 것은?

- ① 참 오라클
- ② 거짓 오라클
- ③ 샘플링 오라클
- ④ 추정 오라클

정답 2



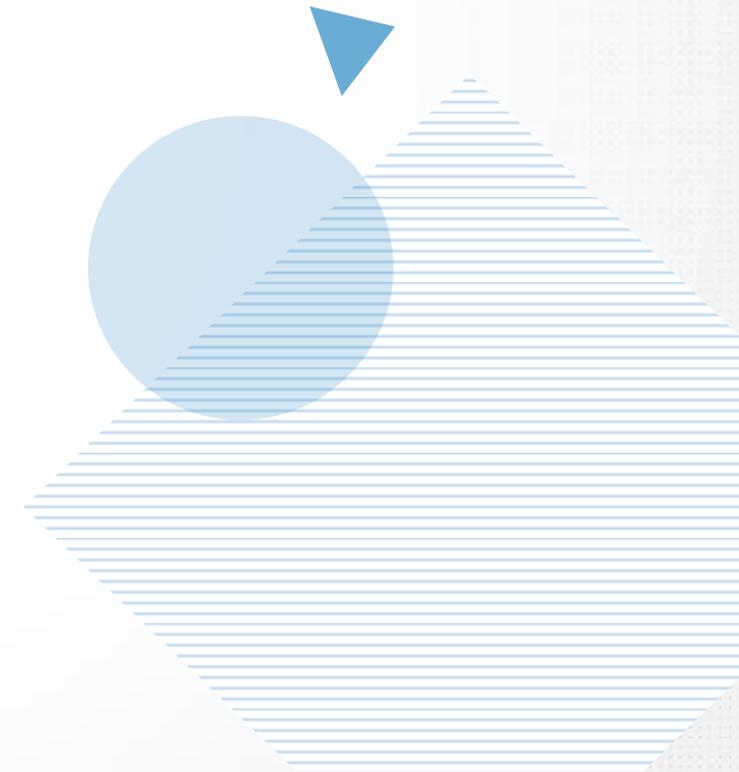
# 제2과목 소프트웨어 개발

## 11 애플리케이션 테스트 관리 C



## 🔗 테스트 자동화 도구

- 테스트 자동화의 개념
- 테스트 자동화 도구의 장점/단점
  - 장점
  - 단점
- 테스트 자동화 수행 시 고려사항



## 테스트 자동화 도구

### ▶ 테스트 자동화 도구의 유형

- 정적 분석 도구, 테스트 실행 도구, 성능 테스트 도구
- 테스트 통제 도구, 테스트 하네스 도구

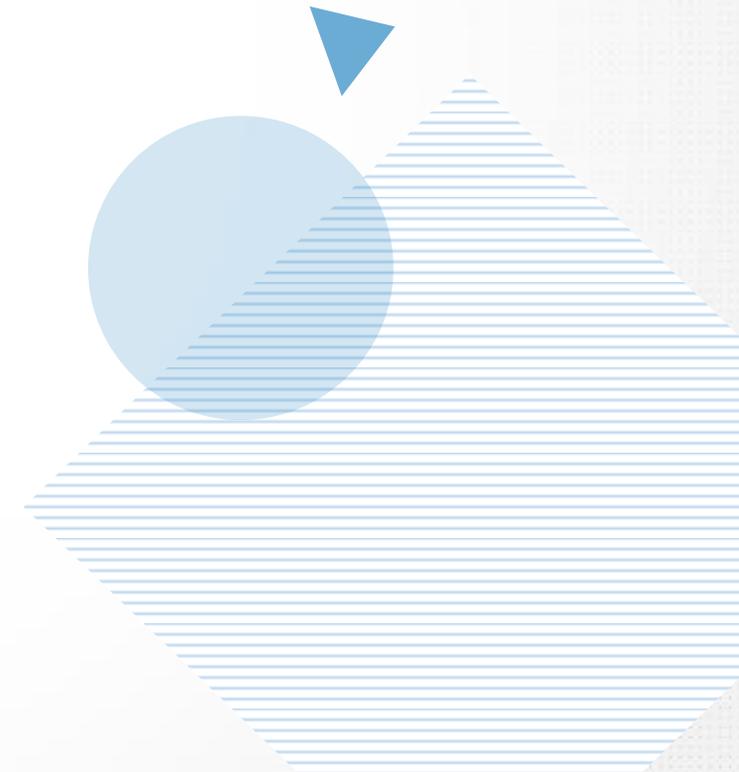
### ▶ 테스트 수행 단계별 테스트 자동화 도구

- 테스트 계획 - 요구사항 관리
- 테스트 분석/설계 - 테스트 케이스 생성
- 테스트 수행 - 테스트 자동화, 정적 분석, 동적 분석, 성능 테스트, 모니터링
- 테스트 관리 - 커버리지 분석, 형상 관리, 결함 추적/관리



## 결함 관리

- 결함(Fault)의 정의
- 결함 관리 프로세스
  - 결함 관리 계획 → 결함 기록 → 결함 검토 → 결함 수정 → 결함 재확인 →
  - 결함 상태 추적 및 모니터링 활동 → 최종 결함 분석 및 보고서 작성
- 결함 상태 추적
  - 결함 분포, 결함 추세, 결함 에이징



## ➤ 결함 추적 순서

- 결함 등록 → 결함 검토 → 결함 할당 → 결함 수정 → 결함 조치 보류 → 결함 종료 → 결함 해제

## ➤ 결함 분류

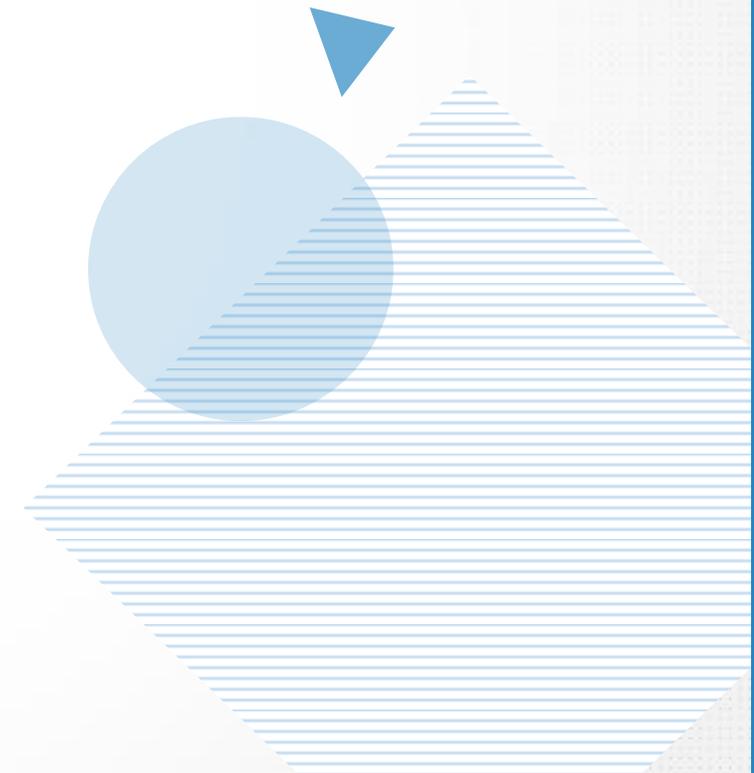
- 시스템 결함, 기능결함, GUI결함, 문서결함

## ➤ 결함 심각도

- High, Medium, Low

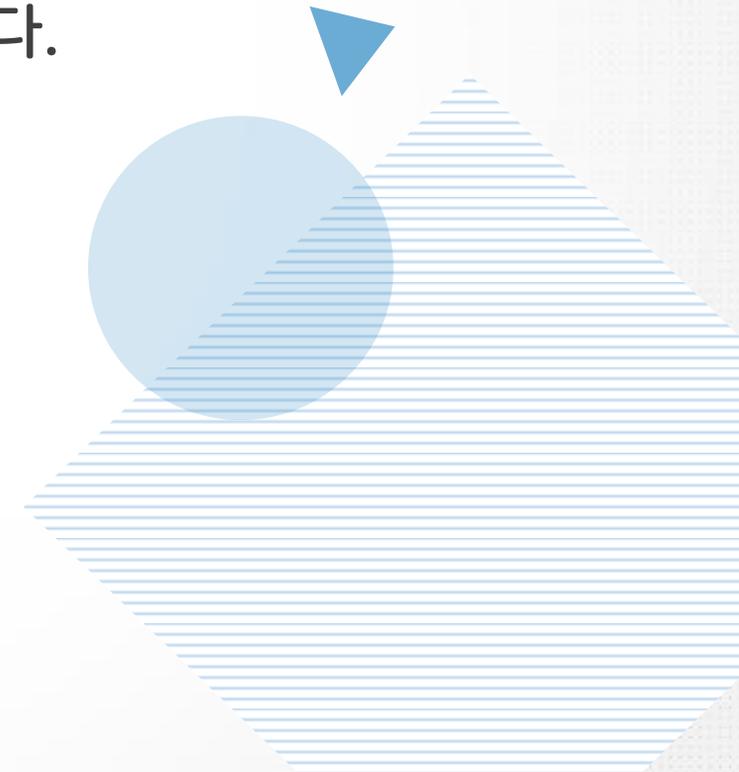


- 결함 우선순위
- 결함 관리 도구
  - Mantis, Trac, Redmine, Bugzilla



- ▶ 다음 테스트 케이스 작성 절차에 대한 설명 중 가장 적합하지 않은 것은?
- ① 테스트 계획 검토 및 자료를 확보한다.
  - ② 결함 해결에 있어 상대적 중요성을 지니며 위험 평가를 통하여 테스트 우선순위 결정을 한다.
  - ③ 테스트 절차, 장비, 도구 및 문서화 방법은 개발자가 결정한다.
  - ④ 요구사항을 테스트할 수 있는 테스트 케이스 도출을 한다.

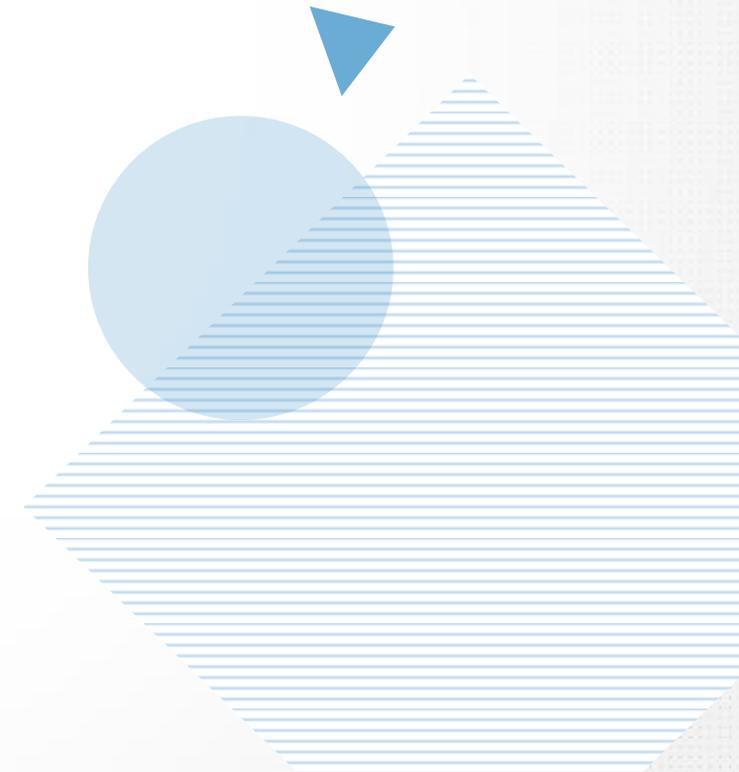
정답 3



▶ 다음은 애플리케이션 통합 테스트 종료 기준이다. 가장 적합하지 않은 것은?

- ① 통합 테스트 수행율
- ② 통합 테스트 합격율
- ③ 통합 테스트 결함 조치율
- ④ 통합 테스트 환경 구축 완료율

정답 4



▶ 다음 중 테스트 자동화에 대한 설명으로 틀린것은?

- ① 테스트 자동화는 사람이 하던 반복적인 테스트 절차를 자동화 하는 것이다.
- ② 테스트 자동화를 수행하면 휴먼 에러를 줄일 수 있다.
- ③ 자동화를 수행하면 테스트 인력 및 시간을 단축할 수 있다.
- ④ 테스트 자동화 도구는 모두 무료이므로 추가 비용이 들지 않는다.

정답 4



# 제2과목 소프트웨어 개발

## 12 애플리케이션 테스트 관리 D

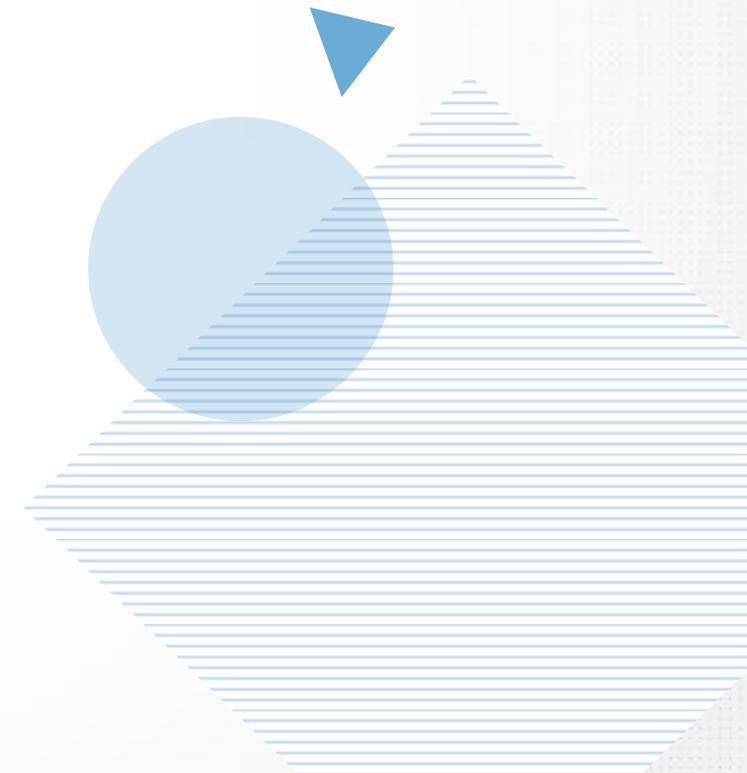


## ➤ 애플리케이션 성능

- 처리량
- 응답시간
- 경과시간
- 자원 사용률

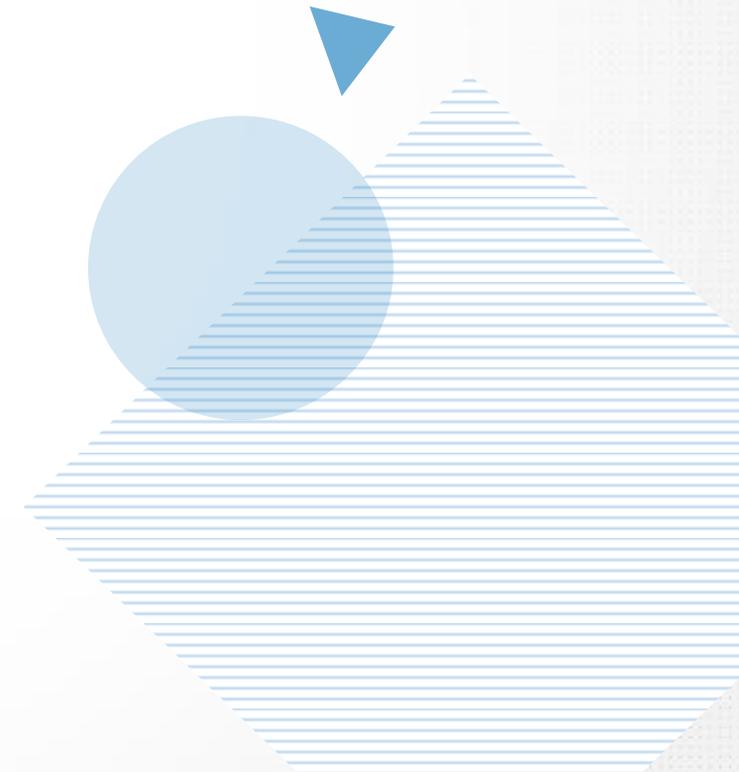
## ➤ 성능 테스트 도구

- Jmeter
- LoadUI
- OpenSTA



## 🔄 애플리케이션 성능 분석

- 시스템 모니터링(Monitoring) 도구
  - Scouter
  - Zabbix
- 애플리케이션 성능 저하 원인 분석



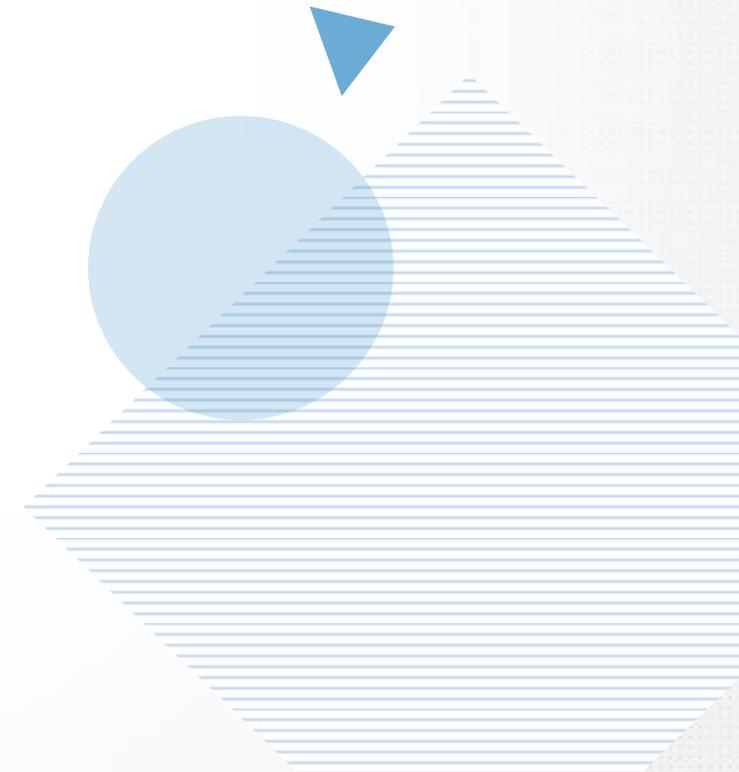
## 🔄 애플리케이션 성능 개선

### ➤ 소스 코드 최적화

- 클린코드, 나쁜코드
- 클린코드 작성원칙 : 가독성, 단순성, 의존성 배제, 중복성 최소화, 추상화

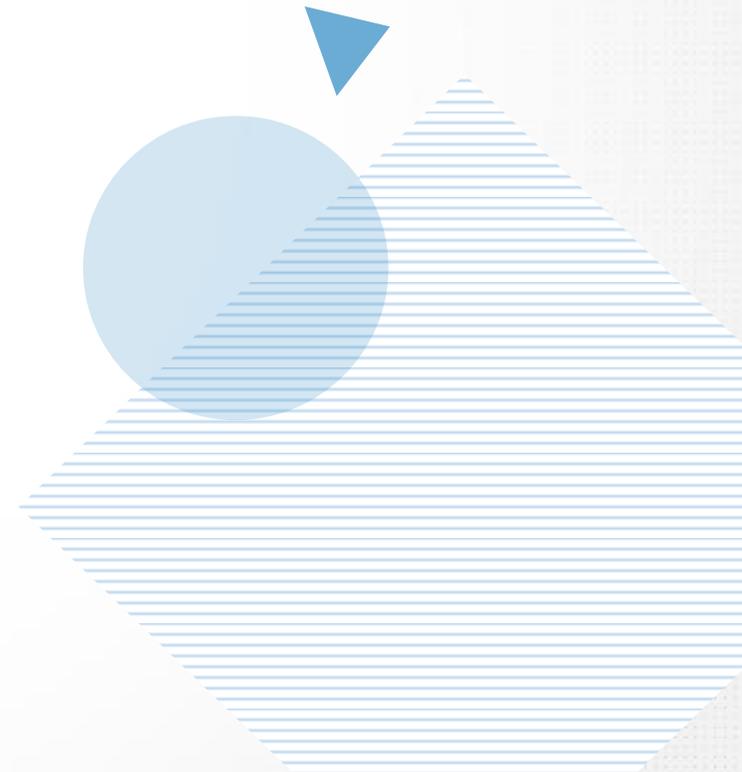
### ➤ 소스 코드 최적화 유형

- 클래스 분할 배치, 느슨한 결합, 코딩 형식 준수



# 🔄 애플리케이션 성능 개선

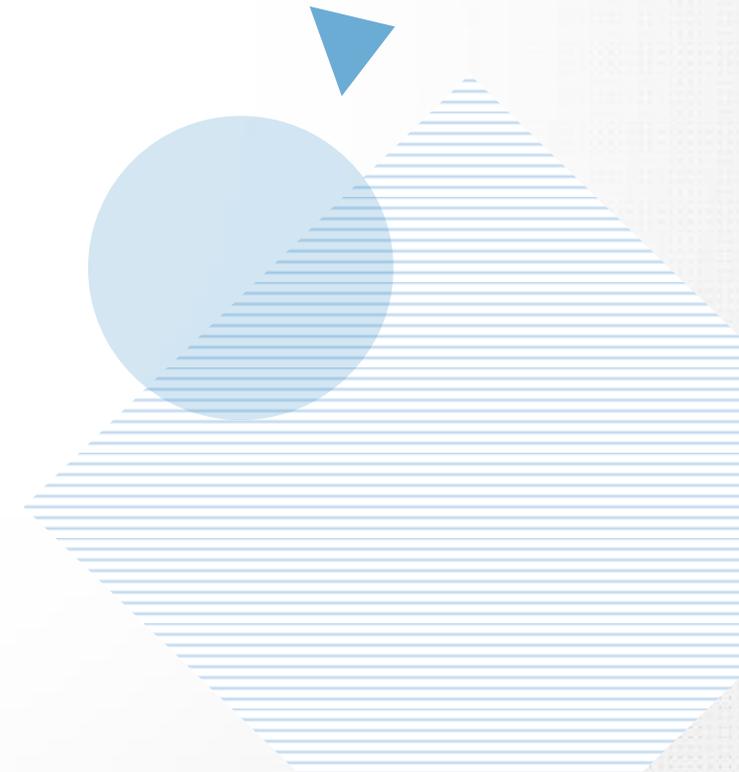
- 소스 코드 품질 분석 도구
  - 정적 분석도구
  - 동적 분석도구
- 소스 코드 품질 분석 도구의 종류
  - Pmd, cppcheck, sonarQube,
  - checkstyle, ccm, cobertura,
  - avalanche, valgrind



# 1. 클린코드 작성 원칙에 대한 설명으로 틀린 것은?

- ① 중복이 최대화된 코드를 작성한다.
- ② 누구든지 쉽게 이해하는 코드를 작성한다.
- ③ 다른 모듈에 미치는 영향이 최소화된 코드를 작성한다.
- ④ 공통된 코드를 사용한다.

정답 1



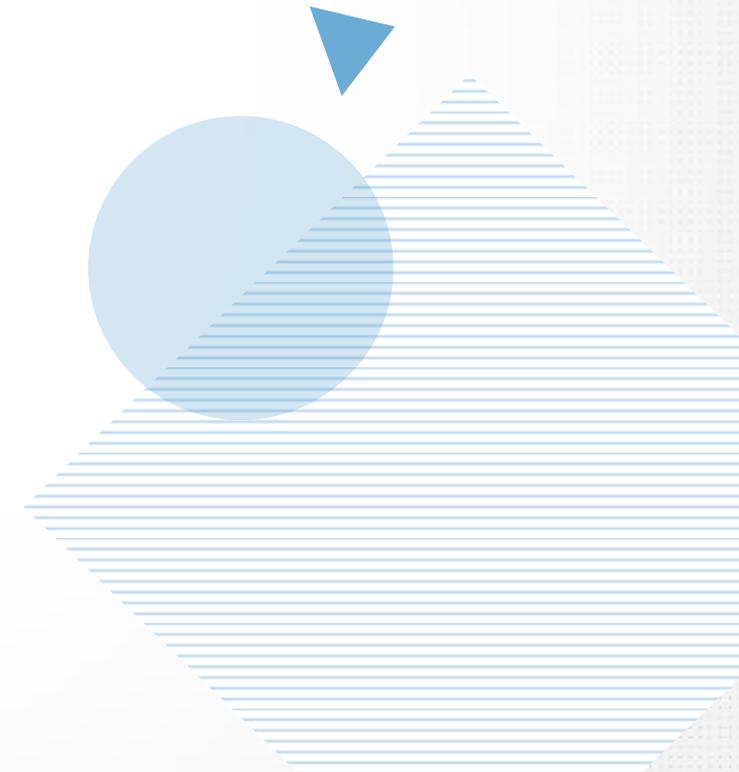
# 제2과목 소프트웨어 개발

## 13 인터페이스 구현 A



## ③ 모듈 간 공통 기능 및 데이터 인터페이스 확인

- 모듈 간 공통 기능 및 데이터 인터페이스의 개요
- 인터페이스 설계서
  - 일반적인 인터페이스 설계서
  - 정적 동적 모형을 통한 인터페이스 설계서
- 인터페이스 설계서별 모듈 기능 확인
  - 외부모듈, 내부모듈
- 모듈 간 공통 기능 및 데이터 인터페이스 확인



# 🔄 모듈 연계를 위한 인터페이스 기능 식별

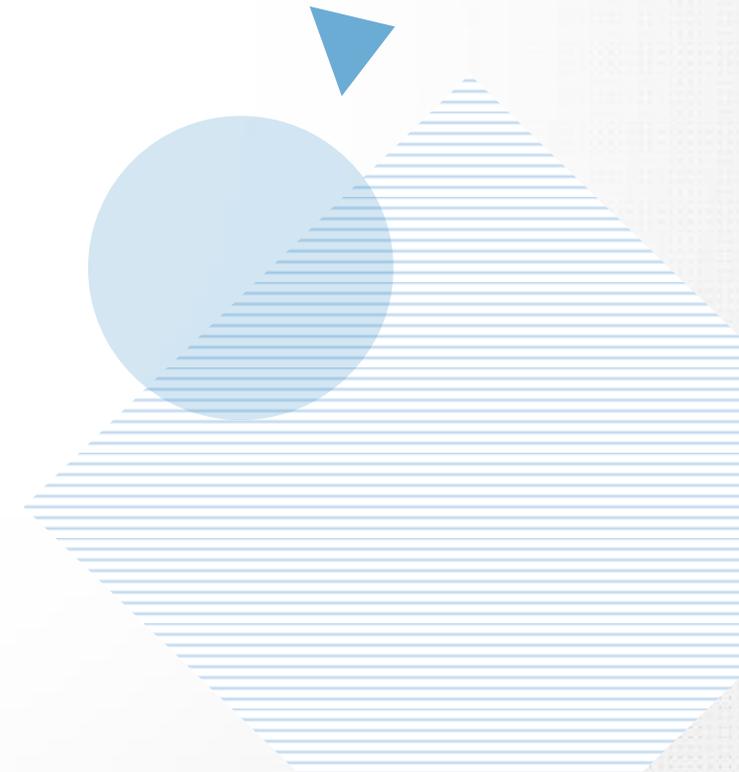
## ➤ 모듈 연계의 개요

- EAI
- ESB

## ➤ 모듈 간 연계 기능 식별

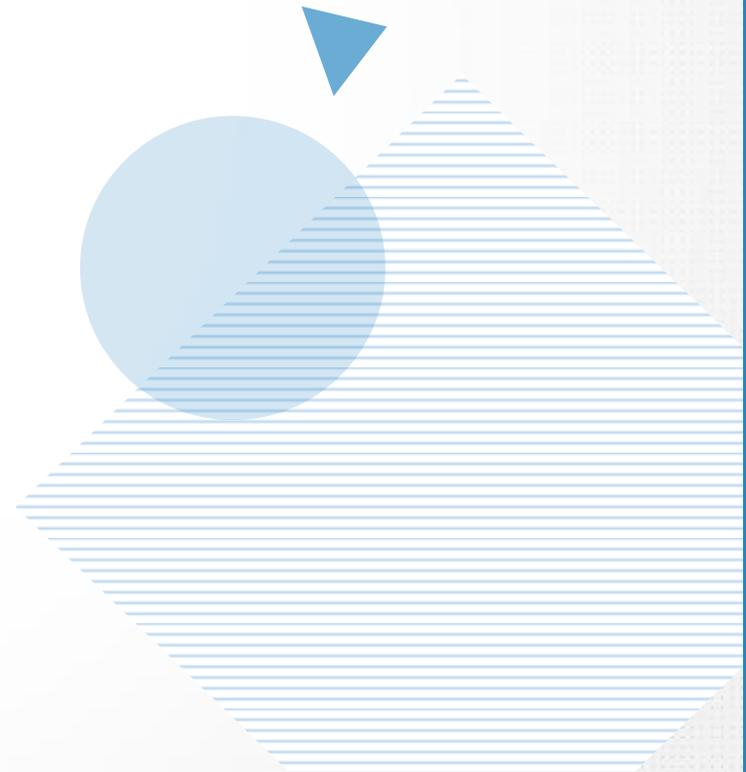
- 외부모듈
- 내부모듈

## ➤ 모듈 간 인터페이스 기능 식별



## 모듈 간 인터페이스 데이터 표준 확인

- 인터페이스 데이터 표준의 개요
- 데이터 인터페이스 확인
- 인터페이스 기능 확인
- 인터페이스 데이터 표준 확인

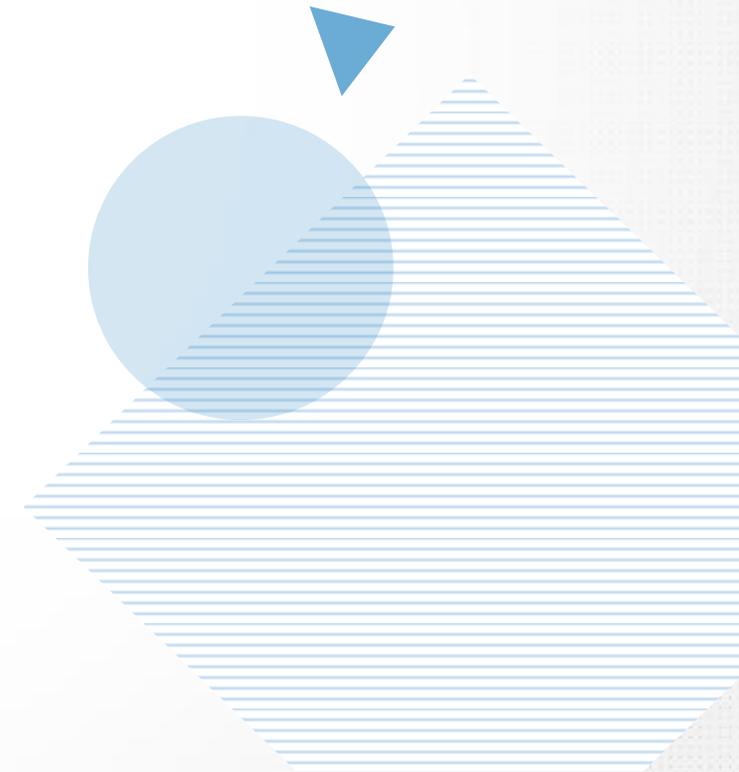


## 1. 다음 설명이 의미하는 것은?

“인터페이스를 위해 인터페이스가 되어야 할 범위의 데이터들의 형식과 표준을 정의하는 것으로, 기존에 있던 데이터중 공통의 영역을 추출하여 정의하는 경우도 있고 인터페이스를 위해 한쪽의 데이터를 변환하는 경우도 있다.”

- ① 인터페이스 요구사항 정의서
- ② 프로그램 명세서
- ③ 인터페이스 데이터 표준
- ④ 인터페이스 목록

정답 3



문혜영 교수

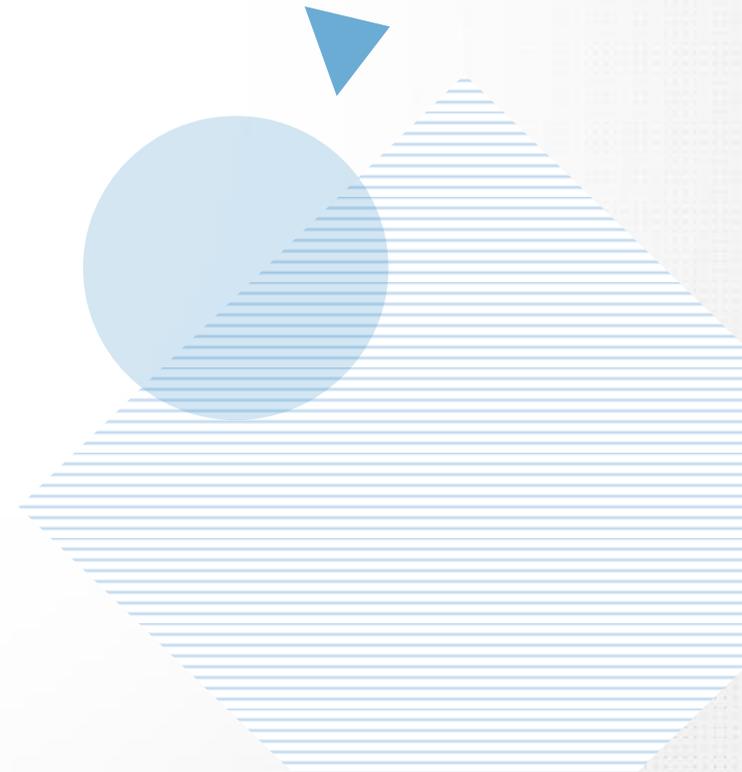
# 제2과목 소프트웨어 개발

## 14 인터페이스 구현 B



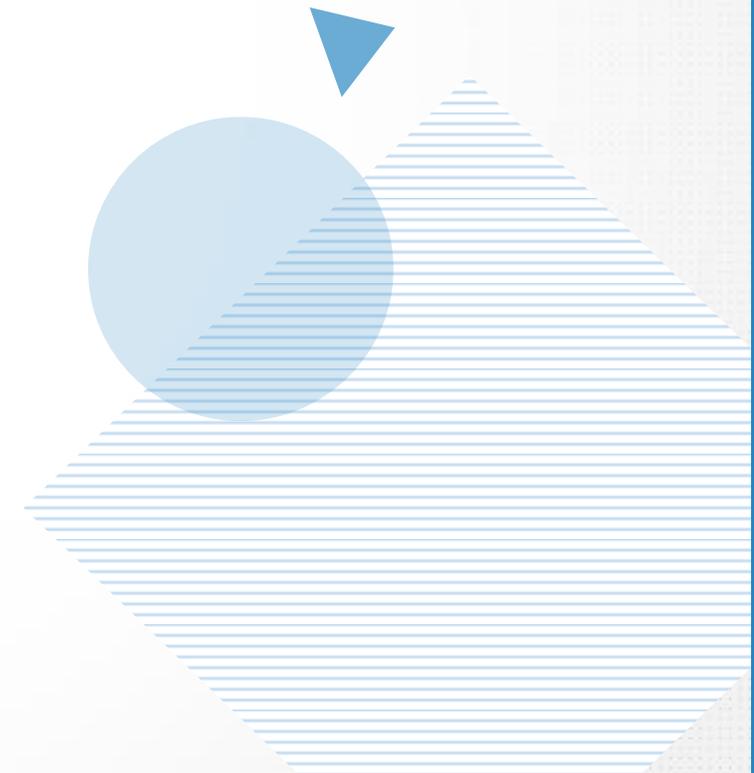
## 🔗 인터페이스 기능 구현 정의

- 인터페이스 기능 구현의 정의 개요
- 모듈 세부 설계서
  - 컴포넌트 명세서, 인터페이스 명세서
- 모듈 세부 설계서 확인
- 인터페이스 기능 구현 정의



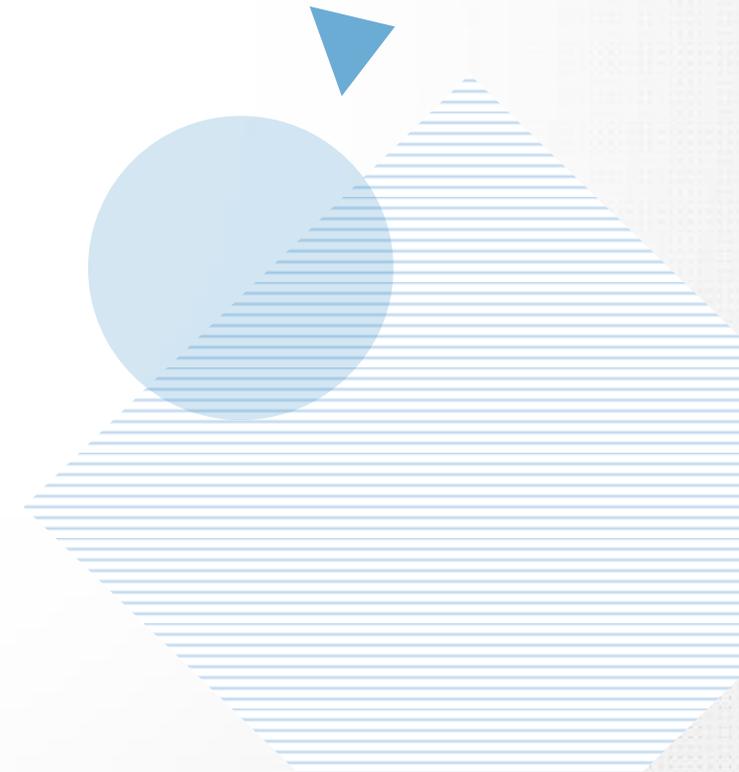
## ③ 인터페이스 구현

- 인터페이스 구현
- 데이터 통신을 이용한 인터페이스 구현
- 인터페이스 엔티티를 이용한 인터페이스 구현



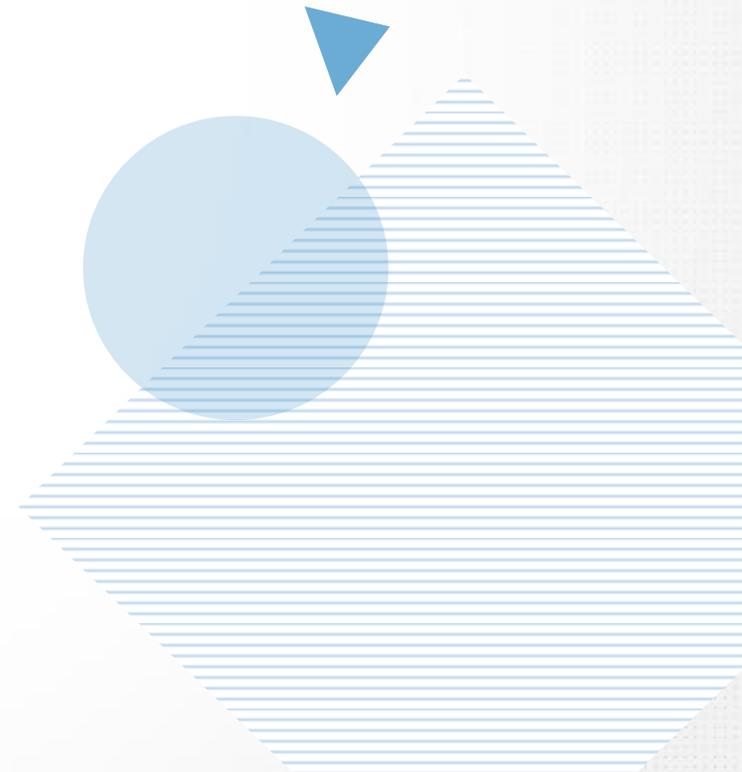
## ③ 인터페이스 예외 처리

- 인터페이스 예외 처리의 개요
- 데이터 통신을 이용한 인터페이스 예외 처리
  - 시스템 환경-
  - 송신 데이터-
  - 프로그램 자체 원인
- 인터페이스 엔티티를 이용한 인터페이스 예외 처리
  - 인터페이스 데이터 생성
  - 인터페이스 테이블에 입력
  - 인터페이스 데이터 전송

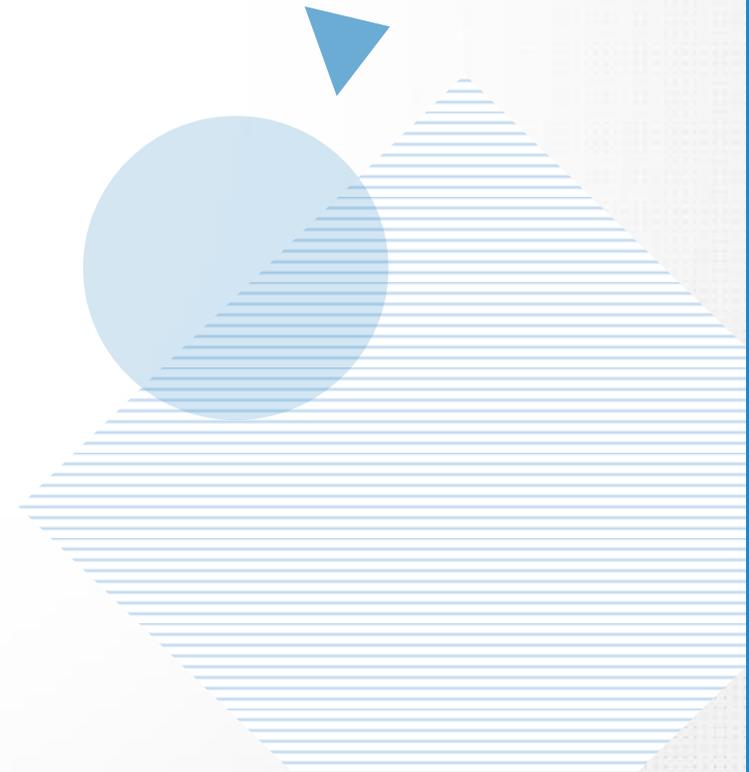


## 인터페이스 보안

- 인터페이스 보안의 개요
- 인터페이스 보안 취약점 분석
- 인터페이스 보안 기능 적용
  - 네트워크 영역
  - 애플리케이션 영역
  - 데이터베이스 영역

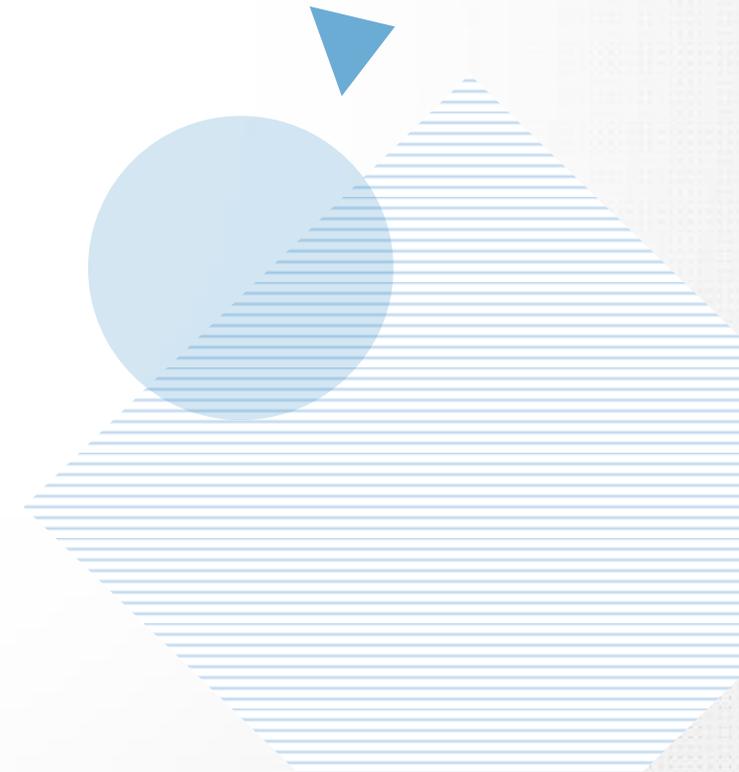


- 연계 테스트의 개요
- 연계 테스트 케이스 작성
- 연계 테스트 환경 구축
- 연계 테스트 수행
- 연계 테스트 수행 결과 검증



## 🔄 인터페이스 구현 검증

- 인터페이스 구현 검증의 개요
- 인터페이스 구현 검증 도구
  - xUnit, STAF, FitNesse, NTAF, Selenium, watir
- 인터페이스 구현 감시 도구
  - 인터페이스 동작상태-APM
  - 애플리케이션 성능 관리도구 - 스카우터, 제니퍼
- 인터페이스 구현 검증 도구 및 감시 도구 선택
- 인터페이스 구현 검증 확인
- 인터페이스 구현 감시 확인



## ③ 인터페이스 오류 확인 및 처리 보고서 작성

- 인터페이스 오류 확인 및 처리 보고서의 개요
- 인터페이스 오류 발생 즉시 확인
  - 오류 메시지 알람 표시, 오류 SMS발송, 오류 내역 이메일 발송
- 주기적인 인터페이스 오류 발생 확인
  - 인터페이스 오류 로그 확인, 인터페이스 오류 테이블 확인, 인터페이스 감시 도구 사용
- 인터페이스 오류 처리 보고서 작성



1. 다음 중 애플리케이션의 흐름 모니터링과 성능 예측을 통해 최적의 애플리케이션 상태를 보장 및 관리하는 것을 의미하는 용어는?

- ① APM
- ② EAI
- ③ API
- ④ ESB

정답 1

