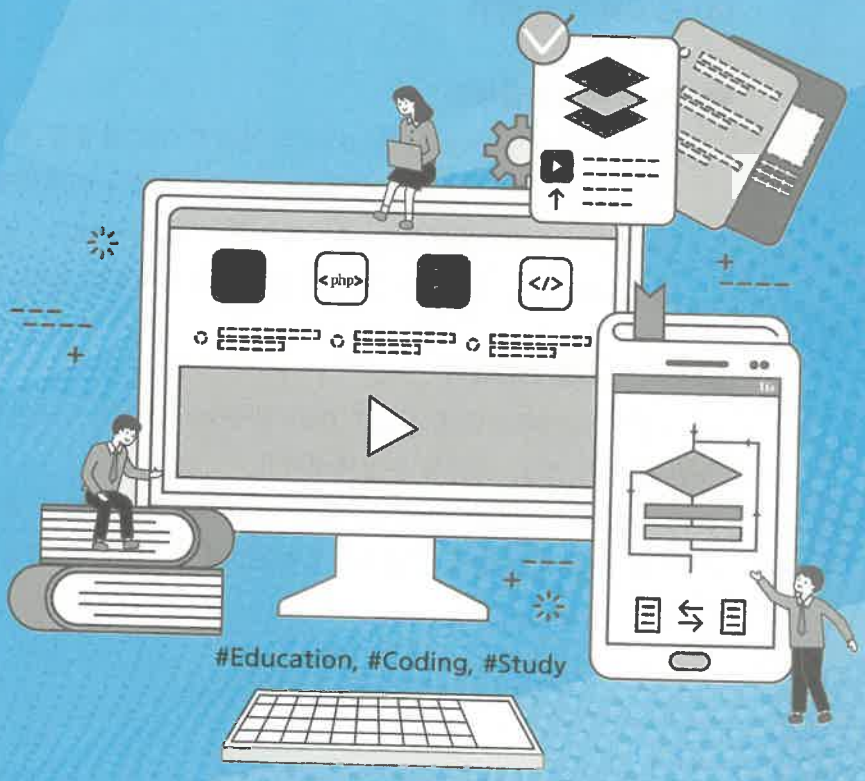
I

응용SW 기초기술 활용



CHAPTER

01 운영체제 기초 활용

1 운영체제의 특징 파악 ★★

(1) 운영체제(Operating System)의 개념

운영체제는 사용자로 하여금 컴퓨터의 하드웨어를 보다 쉽게 사용할 수 있도록 인터페이스를 제공해 주는 소프트웨어이다.

잠깐! 알고가기

인터페이스(Interface)
서로 다른 두 개의 시스템, 장치 사이에서 정보나 신호를 주고받는 경우의 접점이나 시스템이다.

(2) 운영체제의 특징

- 운영체제는 한정된 시스템 자원을 효과적으로 사용할 수 있도록 관리 및 운영함으로써 사용자에게 편리성을 제공한다.
- 운영체제는 컴퓨터 시스템과 사용자 간의 인터페이스(Interface) 기능을 담당한다.

(3) 운영체제의 종류

1 윈도즈 계열 운영체제

- 윈도즈는 MS-DOS의 멀티태스킹 기능과 GUI 환경을 제공하는 응용 프로그램으로서, 마이크로소프트(Microsoft)가 개발한 운영체제이다.
- 마이크로소프트에서 1995년도에 윈도즈 95(Windows 95)를 발표한 이후에 98, ME, XP, 7, 8, 10 등의 버전으로 지속적으로 출시되고 있다.

잠깐! 알고가기

멀티태스킹(Multitasking)
다중작업이라고 말하며 다수의 작업(혹은 프로세스)이 중앙 처리 장치(CPU)와 같은 공용자원을 나누어 사용하는 작업이다.

GUI(Graphical User Interface)
그래픽 환경을 기반으로 한 마우스나 전자펜을 이용하는 사용자 인터페이스이다.

2 유닉스 계열 운영체제

- 유닉스는 1960년대 AT&T Bell 연구소, MIT 및 General Electric 사가 공동으로 연구, 개발한 운영체제이다.
- 초기 운영체제는 Multics이고, C언어로 재이식되어 대중화 기반을 마련하였고, 1970년대에 AT&T가 본격적으로 유닉스 시스템을 판매하게 되었다.

학습 Point

유닉스 운영체제의 특징은 윈도즈 운영체제 특징과 비교해서 시험에 출제되고 있기 때문에 각각의 운영체제별로 특징을 정확하게 알고 있어야 합니다.

3 리눅스 계열 운영체제

- 리눅스는 유닉스의 호환 커널이다.
- 1991년 리누스 토발즈(Linus Tovalds)는 '프리(Free) 소프트웨어' 정책을 가지고, 자유롭고 재배포가 가능한 운영체제인 리눅스를 만들었다.
- 리눅스는 데비안, 레드햇, Fedora, Ubuntu, Cent OS와 같이 다양하게 출시되고 있다.

▼ 유닉스와 리눅스의 차이점

차이점 항목	리눅스	유닉스
비용	• 대부분 무료이며 지원 정책별로 일부 유료 서비스 제품 존재	• 대부분 유료 제품
사용자	• 개발자, 일반 사용자	• 메인프레임 및 워크스테이션 등 대형 시스템 관리자
배포	• 오픈 소스 개발	• 사업자에 의해 배포 • 비용 수반됨
사용자 편의	• GUI 제공, 파일 시스템 지원, Bash 쉘 사용	• 커맨드 기반이 주였으나 GUI도 제공하는 추세 • 파일 시스템 제공
활용	• 스마트폰, 태블릿 등 다양하게 사용	• 인터넷 서버, 워크스테이션 등 대형 시스템에 주로 사용

4 맥(Mac) 운영체제

- 맥 운영체제는 애플이 매킨토시용으로 개발한 그래픽 사용자 인터페이스(GUI) 운영체제이다.
- 애플사는 1999년 OS X로 업데이트를 하였다. 이후에는 클라이언트 버전, 서버 제품 등으로 제품군을 확대하였으며, 2017년 OS X 시에라, 2018년 모하비, 2019년 카탈리나 등을 지속적으로 발표하고 있다.

5 안드로이드(Android) 운영체제

- 안드로이드는 휴대 전화를 비롯한 휴대용 장치를 위한 운영체제와 미들웨어, 사용자 인터페이스 그리고 표준 응용 프로그램(웹 브라우저, 이메일 클라이언트 등)을 포함하고 있는 운영체제다.
- 안드로이드는 리눅스 커널 위에서 동작하며, 자바와 코틀린 언어로 응용 프로그램을 작성할 수 있도록 하고, 컴파일된 바이트 코드를 구동할 수 있는 런타임 라이브러리를 제공한다.
- 안드로이드 소프트웨어 개발 키트(SDK)를 통해 응용 프로그램을 개발하는 데 필요한 각종 도구와 API를 제공한다.

잠깐! 알고가기

메인 프레임(Main Frame)
통계 데이터나 금융 관련 전산 업무, 전사적 자원 관리와 같이 복잡한 작업을 처리하는 컴퓨터이다.

워크 스테이션(Work Station)
과학기술 연산, 공학 설계, 통계 처리, 금융 자료 분석, 컴퓨터 그래픽스 등 전문 분야의 작업을 염두에 둔 고성능 개인용 컴퓨터이다.

Bash 쉘
쉘(Shell)은 커널(Kernel)과 사용자 간의 인터페이스로 사용자로부터 명령을 받아 그것을 해석하고 프로그램을 실행한다. BASH 쉘은 유닉스 및 리눅스에서 사용하는 쉘이다.

잠깐! 알고가기

매킨토시(Macintosh)
애플이 디자인, 개발, 판매하는 개인용 컴퓨터의 제품 이름이다. 줄여서 맥(Mac)이라고 한다.

커널(Kernel)
운영체제의 핵심으로 컴퓨터 자원을 사용자 프로그램이 사용할 수 있도록 관리하는 프로그램이다.

자바(Java)
썬 마이크로시스템즈의 제임스 고슬링이 중심이 되어 개발한 객체 지향적 프로그래밍 언어이다.

코틀린(Kotlin)
코틀린은 Jetbrain 사가 공개한 자바 플랫폼에서 돌아가는 프로그래밍 언어이다.

API(Application Programming Interface)
응용 프로그램에서 사용할 수 있도록, 운영체제나 프로그래밍 언어가 제공하는 기능을 제어할 수 있게 만든 인터페이스이다.

(4) 운영체제 설치 및 운용

1 운영체제의 선택 및 설치

학습 Point

BitLocker는 시험에 출제되었습니다. 꼭 확인해주세요!

① 윈도즈 운영체제의 선택 및 설치

- 개발환경 구축을 위해서는 최소 Pro 또는 Workstation급으로 설치해야 한다.
- 마이크로소프트 공식 홈페이지에 접속하여 최신 윈도즈 정보를 파악한 뒤 설치 여부를 판단한다.
- 라이선스 정책을 확인 및 데이터 유실 방지를 위한 백업 수행

▼ 윈도즈 운영체제 선택을 위한 기준

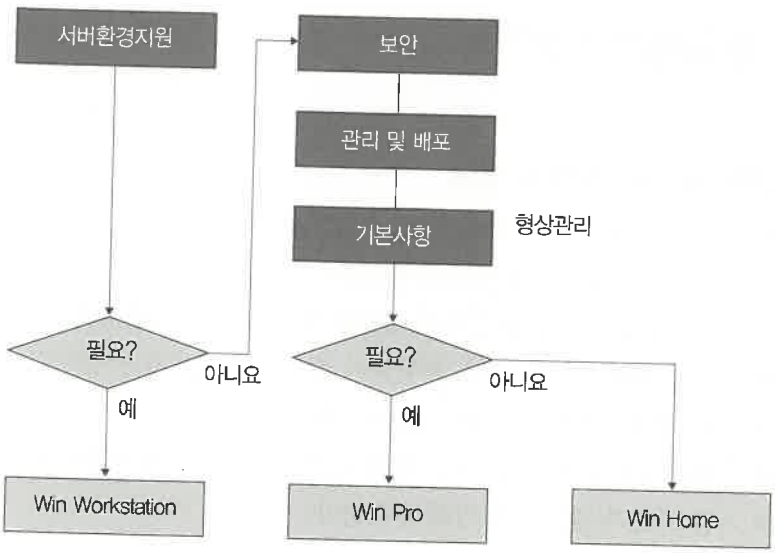
기능	설명
생산성 및 사용자 환경	• 코타나, 잉크, 태블릿 모드, 펜, 터치 등의 주요 기능의 필요 여부
보안	• 윈도즈 정보 보호 기능 및 비트락커(BitLocker) 기능 필요 여부
관리 및 배포	• 그룹 정책의 지원 여부 • 비즈니스용 윈도즈 스토어 지원 여부 • 할당된 액세스 지원 여부 • 다이내믹 권한 설정 지원 여부 • 비즈니스용 업데이트 필요 여부 • 공유되는 PC 구성의 지원 여부
기본 사항	• 도메인 가입 지원의 필요 여부 • 싱글 사인 온(Single Sign On) 기능 필요 여부 • 엔터프라이즈 모드 지원 필요 여부 • 원격 데스크톱을 통한 지원 필요 여부 • 클라이언트 하이프 브이(Hype-V) 기능 필요 여부
서버 환경 지원	• 다중 CPU 지원 필요 여부 • 대용량 메모리 지원 여부 • 데이터 안정성 중요도 및 비휘발성 메모리 지원 여부 • 강력한 복원 기능 필요 여부

잠깐! 알고가기

비트락커(BitLocker)
- 마이크로소프트 윈도즈 운영체제에 포함된 완전한 디스크 암호화 기능으로 볼륨 전체의 암호화를 제공함으로써 중요 자료를 보호하도록 설계되어 있다.
- 기본적으로 비트락커는 128비트 키의 CBC 모드에서 AES 암호화 알고리즘을 사용한다.

싱글 사인 온(Single Sign On)
- 한 번의 로그인을 통해 여러 다른 사이트들을 자동적으로 접속하여 이용하는 기법이다.

하이프 브이(Hype-V) 기능
- 윈도즈 서버 가상화(Windows Server Virtualization)를 위한 하이퍼바이저 기반의 가상화 기능이다.



▲ 운영체제 선택 프로세스

② 리눅스/유닉스 운영체제의 선택 및 설치

- 리눅스와 유닉스는 매우 다양하므로 운영체제별 웹 사이트에 접속하여 지원 내용을 확인하여 목적에 적합한 버전의 운영체제를 선택한다.
- 버전 명을 알면 패치 정보와 제공되는 기능상의 차이를 알 수 있다.
- 사용하려는 버전에서 제공하는 기능을 확인한 이후 운영체제를 선택해서 설치해야 한다.

2 운영체제의 운용

▼ 운영체제의 운용

기능	설명
윈도즈 최신 업데이트 유지	• '윈도즈즈 업데이트'에서 최신 업데이트 정보를 확인 • 침입이 가능한 외부 공격 패턴에 대하여 마이크로 소프트에서 정기적인 업데이트를 수행 • 해당 기능 활성화를 통해 항상 최신 상태 유지
리눅스 최신 업데이트 유지	• 리눅스는 'Software & Updates' 창을 열어 업데이트 수행 • 최신 업데이트를 파악한 후 실행 업데이트 수행 • Shell을 사용할 경우 Ubuntu는 '#sudo apt update', '#sudo apt upgrade' 명령어 사용 • 리눅스마다 명령어가 서로 다른 부분이 있을 수 있으나 동일 기능이 제공되고 있으므로 매뉴얼을 통한 업데이트 명령어의 파악 필요
시스템 복원 기능 활성화 유지	• 예기치 않은 오류로 인한 시스템 재설치를 대비해서 복원 기능의 활성화와 주요 복원 시점 설정

2 운영체제 기본 명령어★★★

(1) 운영체제 기본 명령어 활용

- 운영체제를 제어하기 위한 방법은 CLI(Command Line Interface)와 GUI(Graphic User Interface)가 있다.
- CLI는 사용자가 직접 명령어를 입력하여 컴퓨터에게 명령을 내리는 방식이며, GUI는 마우스로 화면을 클릭하여 컴퓨터를 제어하는 방식이다.
- 마우스 기반의 제어 시스템인 GUI가 개발되면서 CLI의 사용 빈도가 감소하다가 최근 깃허브 등 오픈 소스 기반의 개발환경이 급격히 증가함으로써 CLI의 기본 개념과 명령어를 이해해야 한다.

잠깐! 알고가기

깃허브(GitHub)
분산 버전 관리 도구인 깃(Git)을 사용하는 프로젝트를 지원하는 웹 호스팅 서비스이다.

(2) 윈도즈 운영체제의 기본 명령어

1 CLI 기본 명령어

- CLI 명령어를 입력하기 위해서는 커맨드 창이 필요하다.
- 프로그램 및 파일 검색에서 'CMD'를 입력하거나 윈도즈 보조 프로그램에서 '명령 프롬프트'를 선택하여 커맨드 창을 호출할 수 있다.

▲ 커맨드 창

▼ CMD CLI 기본 명령어

명령어	설명
CD	현재 디렉터리 이름을 보여주거나 변경
COPY	하나 이상의 파일을 다른 위치로 복사

핵인싸 퀴즈

1 ( )은/는 데이터를 정의하는 언어로서 데이터를 담는 그릇을 정의하는 언어이다.

2 ( )은/는 DDL 대상 오브젝트 중 하나로 속성의 데이터 타입과 크기, 제약조건 등의 정보를 나타낸다.

3 ( )은/는 DDL 대상 오브젝트 중 하나로 DBMS 특성과 구현을 고려한 데이터 구조이다.

정답 1. 데이터 정의어(DDL) 2. 도메인(Domain) 3. 스키마(Schema)

명령어	설명
DATE	날짜를 보여주거나 설정
DEL	하나 이상의 파일을 지움
DIR	디렉터리에 있는 파일, 하위 디렉터리 목록을 보여줌
EXIT	CMD 프로그램을 종료
FIND	파일에서 텍스트 문자열을 찾음
FINDSTR	파일에서 문자열을 찾음
HELP	Windows 명령어에 관한 도움말을 제공
MKDIR	디렉터리를 생성
MOVE	하나 이상의 파일을 한 디렉터리에서 다른 디렉터리로 이동
RENAME	파일 이름을 변경
REPLACE	파일을 대체
RMDIR	디렉터리를 지움

- 명령어는 시스템 제어, 파일 제어, 실행, I/O, 실행 중 프로그램 중지, 진단 및 검증과 같이 다양하다.
- 명령어는 Help를 커맨드 창에 입력함으로써 검색할 수 있다.

2 GUI 기본 명령어

- 윈도즈 내에서 파일을 이동하고 프로그램을 실행하는 것 등 모든 것이 GUI 명령에 해당한다.
- 메모리나 디스크 제어 등이 필요할 경우에는 제어판에서 필요 기능을 선택하여 명령을 내릴 수 있다.

(3) 리눅스/유닉스 계열 운영체제의 기본 명령어

- 리눅스와 유닉스 명령어는 쉘에서 입력할 수 있다.
- 쉘의 주요 기능은 다음과 같다.

▼ 쉘 주요 기능

기능	설명
명령어 실행	• 명령어의 모임인 별칭(Alias), 쉘 함수(Function) 등을 찾아 실행
명령어 자동완성	• 쉘은 명령어 자동완성 기능을 사용하여 입력한 명령어를 기록
설정 기능	• 파이프, 입/출력 리다이렉션, 백그라운드 프로세싱 설정
환경 초기화	• TERM 변수를 사용하여 서로 다른 터미널 환경을 초기화 • 쉘은 사용자 환경 정의 파일을 가지고 사용자의 환경을 초기화

- Bourne 계열은 Bash, Korn, Bourne 쉘 등이 존재하고, C 계열은 Csh, Tcsh, Zsh, Ash와 같은 쉘이 있다.

잠깐! 알고가기

쉘(Shell)
운영체제상에서 다양한 운영체제 기능과 서비스를 구현하는 인터페이스를 제공하는 프로그램이다.

• Ksh 쉘은 유닉스에서, Bash 쉘은 리눅스에서 가장 보편적으로 사용되고 있다.

1 CLI 기본 명령어

• 리눅스는 최상위 유저를 CLI 환경에서 #으로 표시하며, 일반 유저를 $로 표시한다.

• 명령어에 대한 도움말은 --help, -h를 명령어 뒤에 붙임으로써 확인할 수 있다.

학습 Point

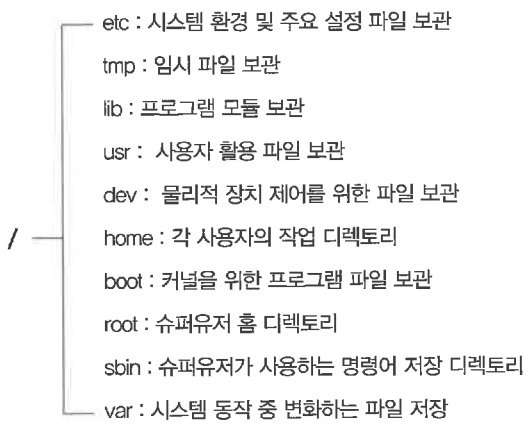
CLI 기본 명령어는 단답형으로 문제에 나오기 좋습니다. 주요 명령어를 알고 가시길 추천합니다!

▼ 리눅스/유닉스 CLI 기본 명령어

명령어	의미	설명
pwd	Print Working Directory	• 현재 작업 중인 디렉터리 정보 출력
cd	Change Directory	• 경로를 이동
ls	List	• 디렉터리 목록 확인
cp	Copy	• 파일 혹은 디렉터리를 복사 • 디렉터리를 복사할 경우 -r 옵션을 필요
mv	Move	• 파일 혹은 디렉터리 이동 • 이름을 변경하는 용도로도 사용
mkdir	Make Directory	• 디렉터리 생성 • -p 옵션을 주면 하위 디렉터리까지 한 번에 생성 가능
rm	Remove	• 파일이나 디렉터리를 삭제 • 디렉터리 삭제 시 -r 옵션을 주어야 함

• 명령어는 파일 디렉터리 관리, 유저 관리, 권한 관리, 프로세스 관리, 통신 관련 등으로 구분할 수 있다.

• 최상위 디렉터리는 '/'이며, root는 최상위 디렉터리 아래의 root 계정의 홈 디렉터리임을 의미한다.



▲ 리눅스에서 디렉터리 구조

2 GUI 기본 명령어

- 리눅스의 GUI는 윈도즈와 같이 기본 설정이 아닌 경우가 많아 버전별로 별도의 설치방법에 따라 GUI 환경을 설치해야 한다.
- 설치 뒤에는 GUI 환경과 CLI 환경을 이동하는 명령어를 사용하여 두 환경을 이동할 수 있다.

(4) 윈도즈와 리눅스/유닉스의 명령어 비교

- 앞서 언급된 윈도즈 CLI 명령어와 리눅스/유닉스의 CLI 명령어를 비교하여 공통점과 차이점을 확인한다.

▼ 윈도즈와 리눅스/유닉스의 명령어 비교

구분	윈도즈	리눅스/유닉스	설명
공통점	cd	cd	• 현재 디렉터리 이름을 보여주거나 변경
	rename	rename	• 파일 이름을 변경
	date	date	• 날짜를 보여주거나 설정
	exit	exit	• 프로그램을 종료
	find	find	• 파일에서 텍스트 문자열을 찾음
	mkdir	mkdir	• 디렉터리를 생성
	rmdir	rmdir	• 디렉터리를 지움
차이점	copy	cp	• 하나 이상의 파일을 다른 위치로 복사
	del	rm	• 하나 이상의 파일을 지움
	dir	ls	• 디렉터리에 있는 파일, 하위 디렉터리 목록을 보여줌
	help	--help -h	• 명령어에 관한 도움말을 제공
	move	mv	• 하나 이상의 파일을 한 디렉터리에서 다른 디렉터리로 이동

3 운영체제 핵심 기능☆

(1) 운영체제 핵심 기능

- 운영체제는 중앙 처리 장치, 메모리, 스토리지, 주변 기기 등을 관리한다.
- 주기억장치와 메모리, 메모리와 스토리지 사이의 속도 차로 인해 캐시메모리 등의 관리 기법들이 개발되었다.
- 초기에는 메모리 용량에 제한이 많아 소프트웨어 개발 시 메모리 관리가 매우 중요했으나 최근에는 운영체제가 대부분 자동관리하여 사용이 편리해졌다.

잠깐! 알고가기

캐시메모리(Cache Memory) 속도가 빠른 장치와 느린 장치 사이에서 속도 차에 따른 병목 현상을 줄이기 위한 범용 메모리이다.

1 메모리 관리

- 메모리 관리는 프로그램의 실행이 종료될 때까지 메모리를 가용한 상태로 유지 및 관리하는 기능이다.
- 메모리에 있는 프로그램은 중앙 처리 장치인 CPU로 이동하여 처리된다.

① 메모리 관리 기본 사항

▼ 메모리 관리 기본 사항

기본사항	설명
가상 메모리 (Virtual Memory)	• 각 프로그램에 실제 메모리 주소가 아닌 가상의 메모리 주소를 부여하는 방식 • 가상 주소(Virtual Address), 물리 주소(Physical Address)가 있고 가상 주소의 범위를 가상 주소 공간, 물리 주소의 범위를 물리 주소 공간이라 함 • 가상 주소 공간은 메모리 관리 장치(MMU)에 의해서 물리 주소로 변환
메모리 관리 장치 (MMU)	• CPU가 메모리에 접근하는 것을 관리하는 컴퓨터 하드웨어 부품 • 가상 메모리 주소를 실제 메모리 주소로 변환 • 메모리 보호, 캐시 관리, 버스 중재 등의 역할을 담당  ▲ MMU 역할 개념도
메모리 관리자	• 기억장치의 어느 부분이 사용 중인지 또는 아닌지를 조사하여 프로세스에게 필요할 때마다 기억장치를 할당 후 회수하는 작업 수행 • 실행 파일 심볼의 재배치 주소를 프로세스의 논리 주소로 연결시키는 작업 수행

핵인싸 퀴즈

4 DDL 명령어 중 데이터베이스 오브젝트를 생성하는 명령어는 ( )이다.

5 DDL 명령어 중 데이터베이스 오브젝트 내용을 삭제하는 명령어는 ( )이다.

6 ( )은/는 테이블의 기본키를 정의하는 제약조건이자 명령어로 유일하게 테이블의 각 행을 식별한다.

정답 4. CREATE 5. TRUNCATE 6. PRIMARY KEY

② 메모리 관리 기법

- 운영체제의 역할에서 메모리 관리는 매우 큰 역할을 차지하는 이유는 메모리가 매우 고가의 자원이고, 시스템에서 중요한 역할을 수행하기 때문이다.
- 메모리 관리 기법에는 반입 기법, 배치 기법, 할당 기법, 교체 기법 등이 있다.

▼ 메모리 관리 기법의 종류

기법	설명	세부 기법
반입 기법	• 주기억장치에 적재할 다음 프로세스의 반입 시기를 결정하는 기법 • 메모리로 적재 시기 결정(When)	• 요구 반입 기법 • 호출 반입 기법
배치 기법	• 디스크에 있는 프로세스를 주기억장치의 어느 위치에 저장할 것인지 결정하는 기법 • 메모리 적재 위치 결정(Where)	• 최초 적합 • 최적 적합 • 최악 적합
할당 기법	• 실행해야 할 프로세스를 주기억장치에 어떤 방법으로 할당할 것인지 결정하는 기법 • 메모리 적재 방법 결정(How)	• 연속 할당 기법 • 분산 할당 기법
교체 기법	• 재배치 기법으로 주기억장치에 있는 프로세스 중 어떤 프로세스를 제거할 것인지를 결정하는 기법 • 메모리 교체 대상 결정(Who)	• 프로세스의 Swap In/Out • FIFO, LRU, LFU

③ 윈도즈 메모리 관리

필요 이상의 메모리를 활용하여 프로그램을 중단시켜야 할 경우, [작업관리자]→[프로세스]의 '프로세스 끝내기'를 클릭하여 종료한다.

④ 리눅스/유닉스 메모리 관리

- top, vmstat –s, free, cat /proc/meminfo 명령어 등을 통해 메모리 상태를 점검한다.

▼ 리눅스/유닉스 메모리 관리 기본 명령어

명령어	설명
top	• 프로세스당 메모리와 CPU 사용량 조회 • 현재 시스템의 CPU와 RAM 사용량을 모니터링
vmstat –s	• proc 명령어와 같이 메모리 사용량 통계를 나타냄
free	• 메모리 사용량을 확인할 때 사용하는 단순하고 쉬운 명령어 • 커널에서 사용하는 공유 메모리와 버퍼의 양 표시
cat /proc/meminfo	• vmstat –s, free 명령어와 유사한 메모리 사용량을 확인 가능

- 메모리가 부족하면 스와핑 기법을 이용한다.

두음쌤 한마디

메모리 관리 기법의 종류

「반배할교」

반입기법 / 배치기법 / 할당기법 / 교체기법

→ 반 배치 때 할배 교감 오심

잠깐! 알고가기

요구 반입 기법
다음에 실행될 프로세스가 참조 요구가 있을 경우에 적재하는 기법이다.

호출 반입 기법
시스템의 요구를 예측하여 미리 메모리에 적재하는 방법으로 요구되는 페이지 이외 다른 페이지도 함께 적재한다.

최초 적합(First Fit)
프로세스가 적재될 수 있는 가용 공간 중에서 첫 번째 분할에 할당하는 방식이다.

최적 적합(Best Fit)
가용 공간 중에서 가장 크기가 비슷한 공간을 선택하여 프로세스를 적재하는 방식(공백 최소화 장점)이다.

최악 적합(Worst Fit)
프로세스의 가용 공간들 중에서 가장 큰 공간에 할당하는 방식이다.

연속 할당 기법
실행을 위한 각 프로세스를 주기억장치 공간 내에서 인접되게 연속하여 저장하는 방법이다.

분산 할당 기법
하나의 프로세스를 여러 개의 조각으로 나누어 주기억장치 공간 내 분산하여 배치하는 기법으로 주로 가상기억장치에서 사용한다. 페이징 기법, 세그먼테이션 기법, 페이징/세그먼테이션 기법 등이 있다.

잠깐! 알고가기

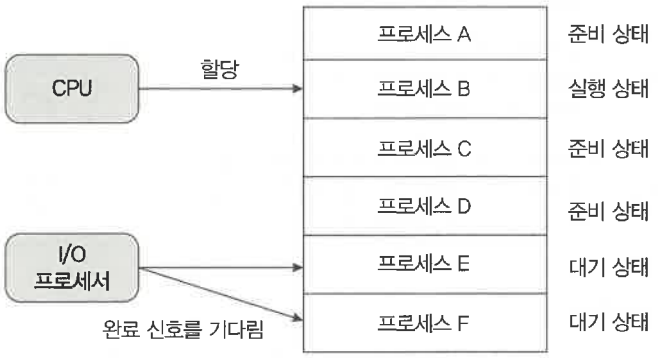
스와핑(Swapping) 기법
프로그램에 할당된 메모리 일부를 디스크에 저장하는 기법이다.

2 프로세스 관리

- 프로세스란 CPU에 의해 처리되는 사용자 프로그램, 시스템 프로그램, 즉 실행 중인 프로그램을 의미하며, 작업(Job) 또는 태스크(Task)라고도 한다.
- 프로세스에 대한 효과적인 관리를 통해 시스템을 최적의 상태로 관리할 수 있다.

① 프로세스 상태

- 하나의 프로세스는 여러 가지 이벤트에 의해 일련의 서로 구분되는 상태 변화를 겪는다.
- 생성 상태, 준비 상태, 실행 상태, 대기 상태, 완료 상태를 가질 수 있다.



▲ 프로세스 상태

▼ 프로세스 상태

프로세스 상태	설명
생성(Create) 상태	• 사용자에 의해 프로세스가 생성된 상태
준비(Ready) 상태	• CPU를 할당받을 수 있는 상태 • 준비 리스트: 각각 우선순위를 부여하여 가장 높은 우선순위를 갖는 프로세스가 다음 순서에 CPU를 할당받음
실행(Running) 상태	• 프로세스가 CPU를 할당받아 동작 중인 상태
대기(Waiting) 상태	• 프로세스 실행 중 입출력 처리 등으로 인해 CPU를 양도하고 입출력 처리가 완료까지 대기 리스트에서 기다리는 상태 • 대기 리스트: 우선순위가 존재하지 않음
완료(Complete) 상태	• 프로세스가 CPU를 할당받아 주어진 시간 내에 완전히 수행을 종료한 상태

② 프로세스 상태 전이

- 프로세스의 상태 전이는 하나의 작업이 컴퓨터 시스템에 입력되어 완료되기까지 프로세스의 상태가 준비, 실행 및 대기 상태 등으로 변하는 활동을 말한다.
- 활동상태란 프로세스가 기억장치를 할당받은 상태이다.

핵인싸 퀴즈

7 ( )은/는 테이블 내에서 같은 값을 가져서는 안 되는 항목을 지정하는 제약조건이자 명령어이다.

8 ( )은/는 데이터베이스에 저장된 자료들을 입력, 수정, 삭제, 조회하는 언어이다.

정답 7. UNIQUE 8. 데이터 조작어(DML)

두음쌤 한마디

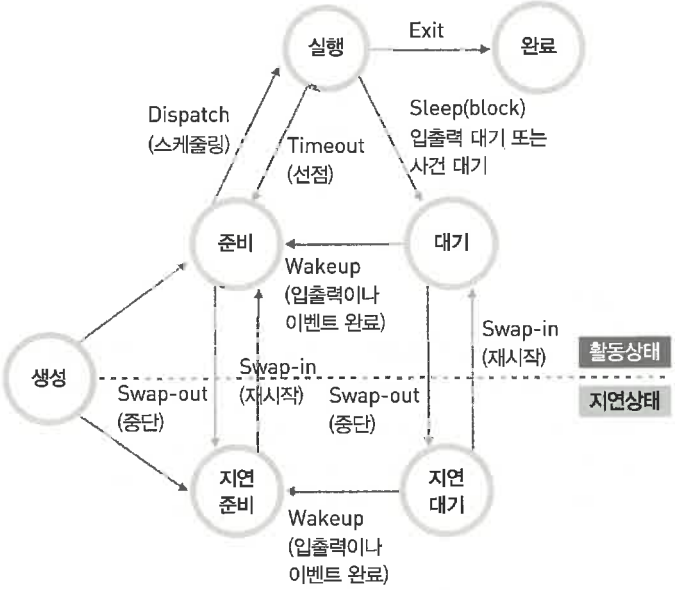
프로세스 상태

「생준 실대완」

생성 상태 / 준비 상태 / 실행 상태 / 대기 상태 / 완료 상태

→ 생존 준비를 위한 실행을 위해 대두와 완두콩을 준비

• 지연상태란 프로세스가 기억장치를 할당받지 못한 상태이다.



▲ 프로세스 상태 전이

▼ 프로세스 상태 전이

프로세스 상태 전이	설명
디스패치 (Dispatch)	• 준비 상태에 있는 여러 프로세스(Ready List) 중 실행될 프로세스를 선정(Scheduling)하여 CPU를 할당(Dispatching) → 문맥 교환 발생 • 프로세스는 준비 상태에서 실행 상태로 전이
타이머 런 아웃 (Timer Run Out)	• CPU를 할당받은 프로세스는 지정된 시간이 초과되면 스케줄러에 의해 PCB 저장, CPU 반납 후 다시 준비 상태로 전이됨 • 프로세스는 실행 상태에서 준비 상태로 전이 • 타임 슬라이스(Time Slice) 만료, 선점(Preemption) 시 타임아웃 발생
블록(Block)	• 실행 상태에 있는 프로세스가 지정된 할당시간을 초과하기 전에 입출력이나 기타 사건이 발생(Block)하면 CPU를 스스로 반납하고 입출력이 완료될 때까지 대기 상태로 전이됨 • 프로세스는 실행 상태에서 대기 상태로 전이 • 즉시 실행 불가능한 시스템 콜, I/O 작업 시작, 프로세스 간 통신 시 Block 발생
웨이크 업 (Wake-Up)	• 어느 순간에 입출력이 종료되면 대기 상태의 프로세스에게 입출력 종료 사실을 Wait & Signal 등에 의해 알려주고, 준비 상태로 전이됨 • 프로세스는 대기 상태에서 준비 상태로 전이
Swap-In	• 프로세스에게 다시 기억장치가 할당될 경우 • 지연 준비 상태나 지연 대기 상태에서 준비 상태나 대기 상태로 전이
Swap-Out	• 프로세스가 기억장치를 잃은 경우 • 준비 상태나 대기 상태에서 지연 준비 상태나 지연 대기 상태로 전이

잠깐! 알고가기

문맥 교환(Context Switching)
CPU가 현재 실행하고 있는 프로세스의 문맥 상태를 프로세스 제어블록(PCB)에 저장하고 다음 프로세스의 PCB로부터 문맥을 복원하는 작업을 문맥 교환이라고 한다.

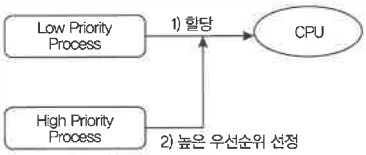
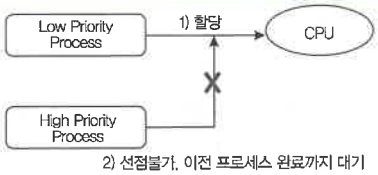
두음쌤 한마디

프로세스 상태 전이
「디타 블웨스」
디스패치 / 타이머 런 아웃 / 블록 / 웨이크 업 / Swap-in / Swap-out
→ 디스코 타임엔 블랙 웨딩 드레스

③ 프로세스 스케줄링 관리

- 프로세스 스케줄링은 CPU를 사용하려고 하는 프로세스들 사이의 우선순위를 관리하는 작업이다.
- 스케줄링은 처리율과 CPU 이용률을 증가시키고 오버헤드, 응답시간, 반환시간, 대기시간을 최소화시키기 위한 기법이다.
- 특정 프로세스가 적합하게 실행되도록 프로세스 스케줄링에 의해 프로세스 사이에서 CPU 교체가 일어난다.

▼ 프로세스 스케줄링의 유형

구분	선점형 스케줄링 (Preemptive Scheduling)	비선점형 스케줄링 (Non Preemptive Scheduling)
개념	하나의 프로세스가 CPU를 차지하고 있을 때, 우선순위가 높은 다른 프로세스가 현재 프로세스를 중단시키고 CPU를 점유하는 스케줄링 방식	한 프로세스가 CPU를 할당받으면 작업 종료 후 CPU 반환 시까지 다른 프로세스는 CPU 점유가 불가능한 스케줄링 방식
개념도	 ▲ 선점형 스케줄링	 ▲ 비선점형 스케줄링
장점	• 비교적 빠른 응답 • 대화식 시분할 시스템에 적합	• 응답시간 예상이 용이 • 모든 프로세스에 대한 요구를 공정하게 처리
단점	• 높은 우선순위 프로세스들이 들어오는 경우 오버헤드 초래	• 짧은 작업을 수행하는 프로세스가 긴 작업 종료 시까지 대기
알고리즘	• 라운드 로빈(Round Robin) • SRT(Shortest Remaining Time First) • 다단계 큐(Multi-Level Queue) • 다단계 피드백 큐(Multi-level Feedback Queue)	• 우선순위(Priority) • 기한부(Deadline) • FCFS • HRN(High Response Rate Next) • SJF(Shortest Job First)
활용	실시간 응답 환경, Deadline 응답 환경	처리시간 편차가 적은 특정 프로세스 환경

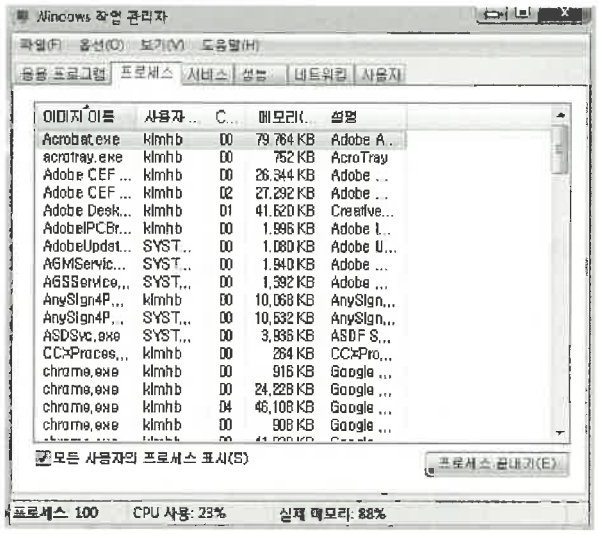
두음쌤 한마디

선점 스케줄링 알고리즘
「SMMR」
SRT / MLQ / MLFQ / Round Robin
→ Show Me the Money 다음 Round에 진출!

비선점 스케줄링 알고리즘
「우기 HFS」
우선순위 / 기한부 / HRN / FCFS / SJF
→ 우리 기업은 홈 패밀리 서비스(HFS)를 제공한다.

④ 윈도즈 프로세스 관리

- 윈도즈에서는 작업관리자의 프로세스 탭에서 다양한 프로세스를 조회할 수 있고 프로그램이 정상 동작하지 않을 때 [작업관리자]→[프로세스]의 '프로세스 끝내기'를 통해 프로세스를 중단시킨다.



▲ 프로세스 관리 창

⑤ 리눅스/유닉스 프로세스 관리

- top 명령어를 사용하여 현재 시스템의 CPU와 RAM 사용량을 모니터링한다.
- 특정 프로세스의 사용량이 많을 경우 해당하는 프로세스 ID(PID)를 확인하고 'kill -9 PID' 명령어를 사용해 종료한다.

(2) 가상화, 클라우드

1 가상화(Virtualization) 개념

- 가상화는 물리적인 리소스들을 사용자에게 하나로 보이게 하거나, 하나의 물리적인 리소스를 여러 개로 보이게 하는 기술이다.
- 대부분의 서버는 용량의 20% 정도만을 사용하는데, 가상화는 서버의 가동률을 60~70% 이상으로 올릴 수 있다.

① 가상화의 종류

▼ 가상화의 종류

종류	설명
플랫폼 가상화	• 하드웨어 플랫폼 위에서 실행되는 호스트 프로그램이 게스트 프로그램을 만들어 마치 독립된 환경을 만들어 낸 것처럼 보여주는 기법
리소스 가상화	• 게스트 소프트웨어 위에서 사용자는 독립된 하드웨어에서 소프트웨어가 실행되는 것처럼 활용하는 기법 • 메모리, 저장 장치, 네트워크 등을 결합하거나 나누기 때문에 사용자는 가상화된 물리적 장치들이 어떤 위치에 있는지 알기 어려움

핵인싸 퀴즈

9 DML 명령어 중에서 데이터를 삽입하는 명령어는 ( )이다.

10 DML 명령어 중에서 데이터를 삭제하는 명령어는 ( )이다.

11 INSERT 명령문은 다음과 같다.

INSERT INTO 테이블명(속성명1, ···)
( )(데이터1, ···)

정답 9. INSERT 10. DELETE 11. VALUES

② 가상화 기술요소

▼ 가상화 기술요소

기술요소	설명
컴퓨팅 가상화	• 물리적으로 컴퓨터 리소스를 가상화하여 논리적 단위로 리소스를 활용할 수 있도록 하는 기술 • 서버 가상화를 통해 하나의 시스템에서 1개 이상의 운영체제를 동시에 가동시킬 수 있으므로, 서버 이용률이 크게 향상 예 하이퍼바이저(Hypervisor)
스토리지 가상화	• 스토리지와 서버 사이에 소프트웨어/하드웨어 계층을 추가하여 스토리지를 논리적으로 제어 및 활용할 수 있도록 하는 기술 • 이기종 스토리지 시스템의 통합을 가능하게 하는 기술 예 분산 파일 시스템
I/O 가상화	• 서버와 I/O 디바이스 사이에 위치하는 미들웨어 계층으로, 서버의 I/O 자원을 물리적으로 분리하고 케이블과 스위치 구성을 단순화하여 효율적인 연결을 지원하는 기술 예 가상 네트워크 인터페이스 카드
컨테이너	• 컨테이너화된 애플리케이션들이 단일 운영체제상에서 실행되도록 해주는 기술 • 하이퍼바이저 없이 운영체제가 격리된 프로세스로 동작하기 때문에 오버헤드가 낮음 예 도커(Docker)
분산처리 기술	• 여러 대의 컴퓨터 계산 및 저장능력을 이용하여 커다란 계산문제나 대용량의 데이터를 처리하고 저장하는 기술
네트워크 가상화 기술	• 물리적으로 떨어져 있는 다양한 장비들을 연결하기 위한 수단으로 중계장치(라우터, 스위치 등)의 가상화를 통한 가상 네트워크(Virtual Network)를 지원하는 기술 예 SDN, NFV

2 클라우드 컴퓨팅(Cloud Computing)

- 클라우드 컴퓨팅은 인터넷을 통해 가상화된 컴퓨터 시스템 리소스(IT 리소스)를 제공하고, 정보를 자신의 컴퓨터가 아닌 클라우드(인터넷)에 연결된 다른 컴퓨터로 처리하는 기술이다.
- 구성 가능한 컴퓨팅 자원(예 컴퓨터 네트워크, 데이터베이스, 서버, 스토리지, 애플리케이션, 서비스)에 대해 어디서나 접근할 수 있다.

① 클라우드 컴퓨팅 분류

클라우드 컴퓨팅은 사설 클라우드, 공용 클라우드, 하이브리드 클라우드로 분류된다.

핵인싸 퀴즈

12 SELECT 명령문은 다음과 같다.

SELECT 컬럼명
( ) 테이블명
WHERE 조건;

13 SELECT 문에서 ( ) 절은 속성값을 그룹으로 분류하고자 할 때 사용하기 위해서 사용한다.

14 UPDATE 명령문은 다음과 같다.

UPDATE 테이블명
( ) 속성명 = 데이터
WHERE 조건;

정답 12. FROM 13. GROUP BY 14. SET

▼ 클라우드 컴퓨팅 분류

분류	주요 내용
사설 클라우드 (Private Cloud)	• 기업 또는 조직 내부에서 보유하고 있는 컴퓨팅 자원(IDC, 서버 등)을 사용하여 내부에 구축되어 운영되는 클라우드 • 자체 컴퓨팅 자원으로 모든 하드웨어, 소프트웨어, 데이터를 수용 • 직접적인 통제가 가능하며 보안성을 높일 수 있음
공용 클라우드 (Public Cloud)	• 클라우드 서비스 제공 업체에서 다중 사용자를 위한 컴퓨팅 자원 서비스를 제공하는 클라우드 • 일정한 비용을 지불하고 하드웨어, 소프트웨어 등을 사용 • 확장성, 유연성 등이 뛰어남
하이브리드 클라우드 (Hybrid Cloud)	• 기업 또는 조직 내부 자원을 이용한 사설 클라우드와 공용 클라우드를 모두 사용하는 클라우드 • 사설 클라우드의 약점인 구축비용 문제와 공용 클라우드의 약점인 보안성 확보 문제를 해결 • 사용 업무의 중요도, 보안성 확보의 중요도 등에 따라 이용 형태 변경가능

학습 Point

클라우드 컴퓨팅 분류는 클라우드 학습의 기초 개념이기 때문에 정확한 학습이 필요합니다.

두음쌤 한마디

클라우드 컴퓨팅 분류

「사공하」

사설 클라우드 / 공용 클라우드 / 하이브리드 클라우드

→ 사공이 하나도 없다.

② 클라우드 컴퓨팅 유형

클라우드 컴퓨팅 서비스는 IaaS, PaaS, SaaS로 구분된다.

▼ 클라우드 컴퓨팅 서비스의 유형

유형	설명
인프라형 서비스 (IaaS; Infrastructure as a Service)	• 서버, 스토리지 같은 시스템 자원을 클라우드로 제공하는 서비스 • 컴퓨팅 자원에 운영체제나 애플리케이션 등의 소프트웨어 탑재 및 실행 • 하위의 클라우드 인프라를 제어하거나 관리하지 않지만 스토리지, 애플리케이션에 대해서는 제어권을 가짐
플랫폼형 서비스 (PaaS; Platform as a Service)	• 인프라를 생성, 관리하는 복잡함 없이 애플리케이션을 개발, 실행, 관리할 수 있게 하는 플랫폼을 제공하는 서비스 • SaaS의 개념을 개발 플랫폼에도 확장한 방식으로 개발을 위한 플랫폼을 구축할 필요 없이, 필요한 개발 요소를 웹에서 빌려 쓸 수 있게 하는 모델 • OS, 애플리케이션과 애플리케이션 호스팅 환경 구성의 제어권을 가짐
소프트웨어형 서비스 (SaaS; Software as a Service)	• 소프트웨어 및 관련 데이터는 중앙에 호스팅되고 사용자는 웹 브라우저 등의 클라이언트를 통해 접속하여 소프트웨어를 서비스 형태로 이용하는 서비스 • 주문형 소프트웨어라고도 함

학습 Point

클라우드 서비스 유형은 시험에서 자주 출제되는 영역이기 때문에 정확한 이해와 암기가 필요합니다.

두음쌤 한마디

클라우드 서비스 유형

「인플소」

인프라형 서비스(IaaS) / 플랫폼형 서비스(PaaS) / 소프트웨어형 서비스(SaaS)

→ 인플루엔자 소식이 전해진다. 예방 접종 필수!

CHAPTER

02 데이터베이스 기초 활용

1 데이터베이스 종류 및 선정 ★★

(1) 데이터베이스(Database)의 개념

학습 Point
데이터베이스는 중요 개념이 다수 출제됩니다! 차근차근 보면서 익히시길 권장합니다.

- 다수의 인원, 시스템 또는 프로그램이 사용할 목적으로 통합하여 관리되는 데이터의 집합이다.
- DBMS(Database Management System)는 데이터베이스 관리의 복잡성을 해결하는 동시에 데이터 추가, 변경, 검색, 삭제 및 백업, 복구, 보안 등의 기능을 지원하는 소프트웨어이다.

(2) 데이터베이스의 종류

- 데이터베이스의 종류로는 파일 시스템, 계층형, 네트워크형, 관계형 DBMS, NoSQL이 존재한다.
- 현재 데이터베이스는 RDBMS와 NoSQL이 주로 사용된다.

▼ 데이터베이스의 종류

유형	설명	DBMS
파일 시스템	• 파일에 이름을 부여하고 논리적으로 어디에 위치시켜야 하는지 등을 정의 • 데이터베이스 전 단계의 데이터 관리 방식	• ISAM • VSAM
계층형 DBMS (HDBMS)	• 데이터를 상하 종속적인 관계로 계층화하여 관리하는 데이터베이스 • 데이터의 액세스 속도가 빠르나 상하 종속적인 관계로 구성	• IMS • System 2000
네트워크형 DBMS(NDBMS)	• 데이터의 구조를 네트워크의 노드 형태로 논리적으로 표현한 데이터 모델 • HDBMS의 상하 종속적인 관계 문제를 해결했으나 구성과 설계가 복잡 • 데이터의 종속성을 해결하지 못함	• IDS • IDMS
관계형 DBMS(RDBMS)	• 가장 보편화된 DBMS로 데이터를 저장하는 테이블의 일부를 다른 테이블과 상하 관계로 표시하여 상관관계를 정리	• Oracle • MS-SQL Server • MySQL

잠깐! 알고가기

ISAM(Indexed Sequential Access Method)
레코드에 순차적으로 접근하거나 또는 색인을 통해 선택적으로 접근하는 방식 둘 모두를 제공하는 파일 관리 시스템이다.

VSAM(Virtual Storage Access Method)
대형 운영체계에서 사용되는 파일 관리 시스템이다.

Oracle
미국 오라클에서 개발한 상용 RDBMS로 리눅스, 유닉스, 윈도즈 모두를 지원하며 대형 시스템에서 많이 사용한다.

MS-SQL Server
마이크로소프트에서 개발한 상용 RDBMS로서 윈도즈 서버에서 주로 구동되며, 개발 언어인 C# 등과 가장 잘 호환된다.

MySQL
썬 마이크로 시스템에서 소유했던 RDBMS로서 오라클에 인수되었다. 리눅스, 유닉스, 윈도즈에서 모두 사용이 가능하고 오픈소스 기반으로 개발되었다.

유형	설명	DBMS
NoSQL(Not only SQL)	• RDBMS의 주요 특성을 제공하지 않으나 뛰어난 확장성과 성능을 제공하는 비관계형 분산 데이터베이스	• Key Value DB • Document DB • Graph DB

(3) 데이터베이스 관리 시스템(DBMS)의 특징

DBMS의 특징으로 보안성, 일관성, 회복성, 무결성, 효율성 등이 있다.

▼ DBMS의 특징

특징	설명
보안성	불법적인 노출, 변경, 손실로부터 보호되어야 하는 성질
일관성	삽입, 삭제, 갱신, 생성 후에도 저장된 데이터가 변함없이 일정한 성질
회복성	장애가 발생하였을 시 특정 상태로 복구되어야 하는 성질
무결성	부적절한 자료가 입력되어 동일한 내용에 대하여 서로 다른 데이터가 저장되는 것을 허용하지 않는 성질
효율성	응답 시간, 저장 공간 활용 등이 최적화되어 사용자, 소프트웨어, 시스템 등의 요구 조건을 만족시켜야 하는 성질

두음쌤 한마디

DBMS의 특징

「보일회무효」

보안성 / 일관성 / 회복성 / 무결성 / 효율성

→ 보를 1회 써서 무효

(4) 데이터베이스 선정 절차

1 DBMS의 종류별 특징을 구분

- 계층형 DBMS, 네트워크형 DBMS, 관계형 DBMS, NoSQL 등의 구조를 참조한다.
- 오픈 소스 기반 DBMS를 사용할 경우 라이선스 정책을 확인해야 한다.

2 데이터베이스 특징에 따른 선택 기준 정의

RDBMS, NoSQL의 키-값 데이터베이스, 컬럼 패밀리 데이터베이스, 도큐먼트 데이터베이스, 그래프 데이터베이스 등의 특징에 따라 선택 기준을 정의한다.

▼ 데이터베이스 유형

구분	특징	제품
관계형 데이터베이스 (Relational Database)	• 테이블의 구조(스키마)를 정의하고 테이블 간의 관계를 정의하는 데이터베이스 • 가장 광범위하게 쓰이는 DBMS	• Oracle • MySQL • MS-SQL
키-값 데이터베이스 (Key Value Database)	• 간단한 키-값 메서드를 사용하여 데이터를 저장하는 비 관계형 데이터베이스 • 임베디드 시스템과 같은 간단한 시스템에 적합	• Redis • Memcached

핵인싸 퀴즈

15 ( )은/는 데이터베이스 관리자가 데이터 보안, 무결성 유지, 병행제어, 회복하기 위해 관리자(DBA)가 사용하는 제어용 언어이다.

16 GRANT 명령문은 다음과 같다.
GRANT 권한 ON 테이블 ( ) 사용자;

17 GRANT 명령문은 다음과 같다.

REVOKE 권한 ON 테이블 ( ) 사용자;

정답 15. 데이터 제어어(DCL) 16. TO 17. FROM

잠깐! 알고가기

JSON(JavaScript Object Notation)
인간이 읽을 수 있는 데이터 교환용으로 설계된 경량 텍스트 기반 개방형 표준 포맷이다.

두음쌤 한마디

NoSQL의 유형
「키컬도그」
키-값 / 컬럼 패밀리 / 도큐먼트 / 그래프 데이터베이스
→ 카키 컬러 옷을 입은 도그(개)

구분	특징	제품
컬럼 패밀리 데이터베이스 (Column Family Database)	• 하나의 Key에 여러 개의 컬럼이 달려 있는 형태의 비 관계형 데이터베이스 • 하나의 Row를 식별하기 위한 키를 갖고 있음	• Cassandra • Hbase • BigTable
도큐먼트 데이터베이스 (Document Database)	• JSON과 유사한 형식의 문서로 데이터를 저장 및 쿼리하도록 설계된 비 관계형 데이터베이스 • 일관된 구조가 필요 없고 컬럼은 하나 이상의 값을 가질 수 있음	• Mongo DB • Amazon Document DB • Couchbase
그래프 데이터베이스 (Graph Database)	• 그래프 구조를 사용하여 데이터를 표현하고 저장하는 비 관계형 데이터베이스 • 노드 간 관계를 구조화하여 저장	• Neo4j • OrientDB • ArangoDB

3 상용 DBMS 및 오픈 소스 DBMS 제품 파악 및 선정

- 상용화 데이터베이스 관리 시스템과 오픈 소스 기반 데이터베이스 관리 시스템의 주요 제품들을 파악한다.
- 주요 제품 중 2~3개의 DBMS를 선정하여 장점과 단점을 확인하여 적합한 DBMS를 선정한다.

▼ 상용화 및 오픈 소스 DBMS

Top 5 상용화 DBMS	Top 5 오픈 소스 기반 DBMS
Oracle	MySQL
MS SQL Server	PostgreSQL
DB2	Mongo DB
Microsoft Access	Redis
Teradata	Elasticsearch

2 관계형 데이터베이스★★★

(1) 개체-관계 다이어그램(ERD; E-R Diagram)

1 개체-관계 다이어그램의 개념

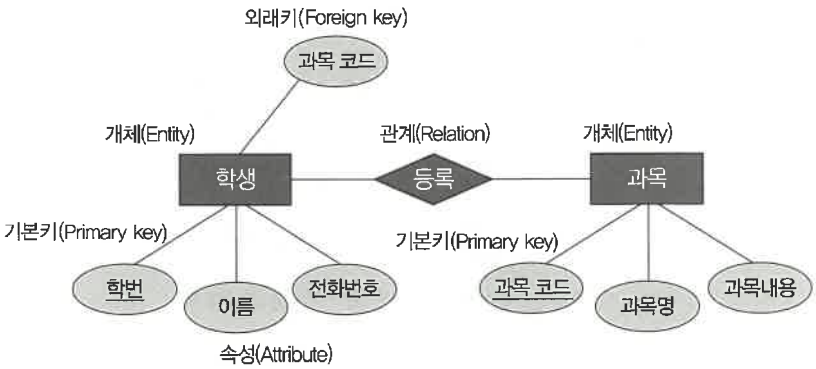
- 업무 분석 결과로 도출된 개체(Entity)와 개체 간의 관계(Relation)를 도식화한 표현이다.
- ERD로 요소 간 연관성을 도식화하여 데이터베이스 관리자, 개발자, 사용자 모두 데이터의 흐름과 연관성을 공통적으로 쉽게 확인할 수 있다.

2 개체-관계 모델(E-R Model)의 개념

- ERD의 구성요소인 개체, 관계, 속성을 추출하기 위해서는 업무나 시스템에 대한 명확한 정의가 있어야 한다.
- ERD로 도식화하기 전 각 개체를 사각형, 화살표, 마름모로 표기한 형태를 개체-관계 모델이라고 한다.

기 호	의 미
	개체
	속성
	기본키
	관계
	개체 타입과 속성을 연결
	개체 간의 관계 타입

▲ 개체-관계 모델 표준기호



▲ 개체-관계 모델 예시

▼ 업무로부터 개체, 관계, 속성 추출

업무	추출 결과
• 각각의 종업원은 한 매장에 소속 • 종업원에 관해서는 사번, 이름, 주소, 휴대폰 번호의 정보를 관리 • 매장에 대해서는 매장 코드, 매장명, 매장 전화번호, 매장 주소 정보가 유지	• 개체: 종업원, 매장 • 관계: 소속 • 속성: 사번, 이름, 주소, 휴대폰 번호, 매장 코드, 매장명, 매장 전화번호, 매장 주소

잠깐! 알고가기

외래키(Foreign Key)
한 테이블의 필드 중 다른 테이블의 행을 식별할 수 있는 키다.

기본키(Primary Key)
테이블 내 동일한 레코드가 입력되는 경우를 구분해 줄 수 있는 식별자이다.

핵인싸 퀴즈

18 ( )은/는 불법적인 데이터의 접근으로부터 데이터베이스를 보호하는 기법으로, 데이터베이스는 가장 내부에 위치하고, DBMS 자체는 강력한 보안 기능을 제공하기 때문에 접근 권한을 가진 사용자가 권한을 남용하여 유출하거나 변조가 가장 큰 위험이다.

19 신원 기반 접근제어 정책은 주체나 그들이 속해있는 그룹들의 신분에 근거하여 객체에 대한 접근을 제한하는 방법으로 ( ) (이)라고 불린다.

정답 18. 접근제어(Access Control)
19. DAC(Discretionary Access Control)

두음쌤 한마디

논리 데이터 모델링 속성
「개속관」
개체 / 속성 / 관계
→ 개속 관계를 유지하자

① 개체(Entity)

- 사물 또는 사건으로 정의되며 개체라고도 한다.
- ERD에서 개체는 사각형으로 나타내고 사격형 안에는 개체의 이름을 넣는다.

> - 가능한 한 대문자로 개체 이름을 써주며 단수형으로 명명한다.
> - 유일한 단어로 정한다.

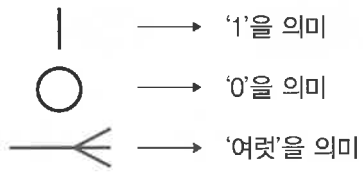
② 속성(Attribute)

- 개체가 가지고 있는 요소 또는 성질을 속성이라 부른다.
- 속성은 선으로 연결된 동그라미로 표기하거나 표 형식으로 표기하기도 한다.
- 관계형 데이터베이스 활용을 위해서는 까마귀발 모델(Crow's Foot Model)이 편리하다.

> - 속성 명은 단수형으로 명명한다.
> - 개체 명을 사용하지 않는다.
> - 속성이 필수 사항(Not Null)인지, 필수 사항이 아닌지(Null) 고려하여 작성한다.

③ 관계(Relationship)

- 두 개체 간의 관계를 정의한다.
- 개체는 사각형(ㅁ), 속성은 타원형(ㅇ)을 이용하여 표시한다.



▲ 관계 표시 기호(Crow's Foot Model)

(2) ERD 작성 절차

1 ERD 작성을 위한 요소 추출

업무의 흐름으로부터 개체, 관계, 속성을 추출하는 작업을 수행한다.

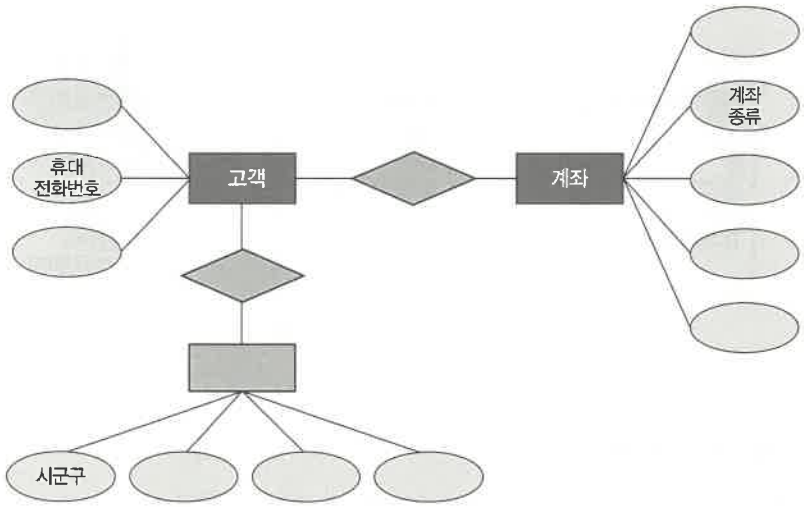
> **시나리오**
> (1) 보험사에서 사용하는 데이터베이스를 설계한다.
> (2) 고객이 있고 고객이 사용하는 계좌가 있다.
> (3) 고객에 대한 정보로는 이름, 주소, 휴대 전화번호, 주민 등록 번호가 있다.
> (4) 고객 계좌에 대한 정보로는 계좌 번호, 계좌 종류, 잔고, 개설 일자, 인출 한도가 있다.
> (5) 주소 정보에 대한 정보로는 시군구, 동, 상세 주소, 우편 번호가 있다.

▼ 업무로부터 개체, 관계, 속성 추출

구분	추출 요소
개체	• 고객, 계좌, 전화번호
속성	• 이름, 휴대전화번호, 주민등록번호, 계좌번호, 계좌종류, 잔고, 개설일자 • 인출 한도, 시군구, 동, 상세 주소, 우편 번호
관계	• 고객이 계좌번호를 소유 • 고객이 주소를 소유

2 ERD를 작성

추출된 요소를 가지고 ERD를 작성하고 일부 작성된 ERD에 추출 요소들을 입력한다.



▲ ERD 작성 예제

3 ERD 관계형 스키마 작성

작성된 ERD에 추가적인 제약조건을 고려하여 논리 관계형 스키마를 작성한다.

[제약조건]
(1) 고객은 다수의 집을 보유할 수 있다.
(2) 고객은 다수의 계좌를 보유할 수 있다.

① 테이블 형태로 표시

- 테이블의 가장 위 칸은 유일하게 필드를 구분할 수 있는 구분자(유일 키)가 되어야 한다.
- 유일 키를 만드는 방법은 여러 가지가 있으며, 일련번호를 부여하기도 한다.
- 유일 키 아래 칸에는 속성들을 기입한다.

핵인싸 퀴즈

20 규칙기반 접근제어 정책은 객체에 포함된 정보의 비밀성과 이러한 비밀성의 접근정보에 대하여 주체가 갖는 권한에 근거하여 객체에 대한 접근을 제한하는 방법으로 ( )(이)라고 불린다.

21 역할기반 접근제어 정책은 중앙관리자가 주체와 객체의 상호관계를 제어하며 조직 내에서 맡은 역할에 기초하여 자원에 대한 접근 허용 여부 결정하는 방법으로 ( )(이)라고 불린다.

22 ( )은/는 데이터를 빠르게 찾을 수 있는 수단으로서, 테이블에 대한 조회 속도를 높여 주는 자료구조이다.

23 ( )은/는 데이터가 정렬된 순서로 생성되는 인덱스로 B-Tree 알고리즘 활용하는 인덱스이다.

정답 20. MAC(Mandatory Access Control) 21. RBAC(Role Based Access Control) 22. 인덱스(Index) 23. 순서 인덱스(Ordered Index)

잠깐! 알고가기

릴레이션(Relation)
행(Row)와 열(Column)로 구성된 테이블이다.

튜플(Tuple)
릴레이션의 행(Row)에 해당하는 요소이다.

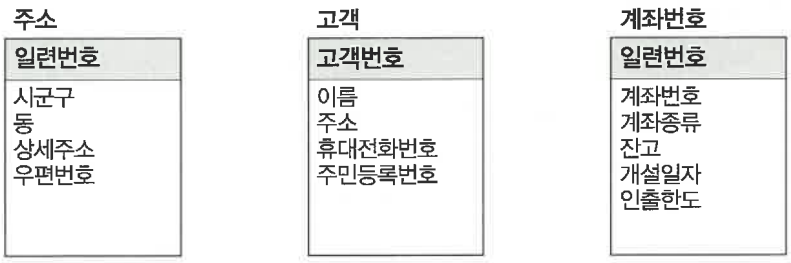
속성(Attribute)
릴레이션의 열(Column)에 해당하는 요소이다.

카디널리티(Cardinality)
튜플(Row)의 수이다.

차수(Degree)
애트리뷰트(Column)의 수이다.

스키마(Schema)
릴레이션이 어떻게 구성되는지, 어떤 정보를 담고 있는지에 대한 기본적인 구조이다.

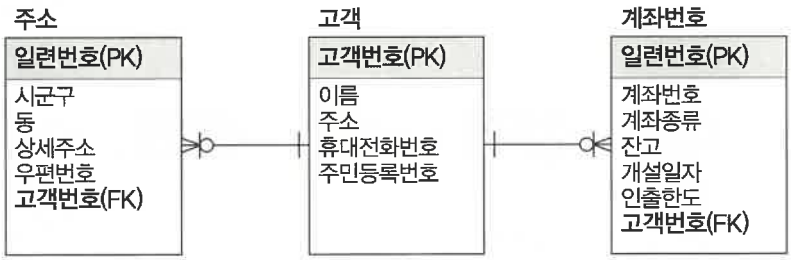
인스턴스(Instance)
정의된 스키마에 따라 생성된 테이블에 실제 저장된 데이터의 집합이다.



▲ ERD 작성을 위한 테이블 표시

② 테이블 간 관계 설정

- 테이블 간 관계를 표기할 때는 1:1, 1:N, N:1, N:N의 관계를 고려해야 한다.
- 테이블 간 연결할 수 있는 속성을 부여하여 하나의 테이블에서 다른 테이블의 값을 찾아갈 수 있어야 한다.



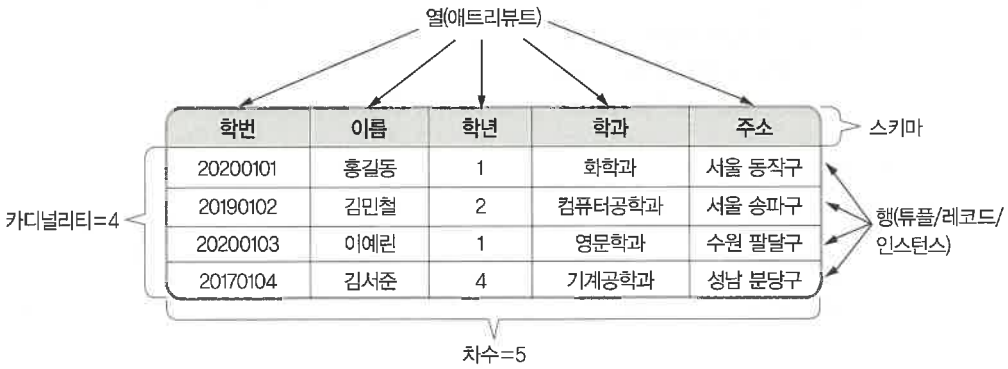
▲ ERD 관계 설정

(3) 관계 데이터 모델

학습 Point

관계 데이터 구조의 주요 용어, 특히 카디널리티와 차수를 묻는 문제가 출제됩니다. 계산하는 방법을 반드시 숙지하고 넘어가세요!

- 개체 집합에 대한 속성 관계를 표현하기 위해 개체를 테이블로 사용하고 개체 집합들 사이의 관계를 공통 속성으로 연결하는 데이터 모델이다.
- 관계 데이터 모델을 구성하는 용어는 릴레이션, 튜플, 속성, 카디널리티, 차수, 스키마, 인스턴스 등이 있다.
- 학생 릴레이션을 기준으로 관계 데이터 모델의 구조를 이해한다.



▲ 학생 릴레이션의 관계 데이터 구조

(4) 관계형 데이터베이스 테이블 생성

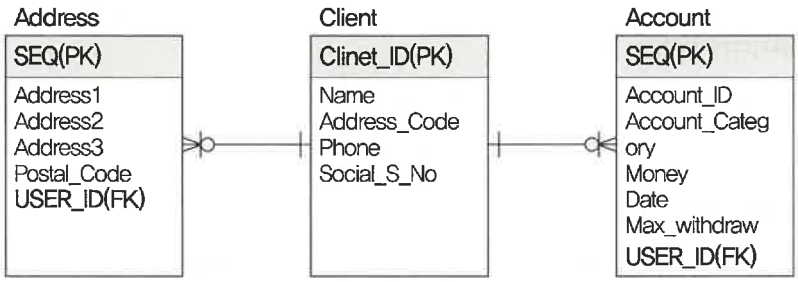
1 SQL(Structured Query Language)의 개념

학습 Point
SQL의 기본 개념과 명령어는 핵심입니다. 명령어 문제가 나올 가능성이 높으니 잘 봐두시길 당부드립니다.

- 관계형 데이터베이스 관리 시스템(RDBMS)의 데이터를 관리하기 위해 설계된 특수 목적의 프로그래밍 언어이다.
- 데이터베이스의 종류와 상관없이 명령어는 국제 표준으로 제정된 SQL을 사용한다.
- 미국 표준 연구소인 ANSI와 국제 표준화 기구인 ISO에서 SQL을 관계 데이터베이스의 표준 질의어로 채택하고 표준화하고 있다.

2 테이블 생성을 위한 물리 스키마 작성

- SQL 표준 언어는 한글 입력이 허용되지 않는다.
- 다음은 스키마와 매칭되는 물리 스키마이다.



▲ 물리 모델 작성

3 테이블 생성을 위한 SQL 명령어 작성

- 테이블을 관리하기 위한 명령어로 CREATE, ALTER, DELETE, TRUNCATE가 있다.

▼ 테이블 관리 명령어

구분	명령어	설명
생성	CREATE	데이터베이스 오브젝트 생성
수정	ALTER	데이터베이스 오브젝트 변경
삭제	DROP	데이터베이스 오브젝트 삭제
	TRUNCATE	데이터베이스 오브젝트 내용 삭제

두음쌤 한마디
테이블 관리 명령어
「크알드트」
CREATE / **A**LTER / **D**ROP / **T**RUNCATE

- 작성 시 필요한 데이터 타입은 숫자형(INT), 날짜형(DATE), 문자형(VARCHAR) 등으로 다양하다.

▼ 관계형 테이블 생성 예제

SQL 명령어	예제

CREATE TABLE 테이블명(
  컬럼 1 데이터 타입,
  컬럼 2 데이터 타입,
  ….
);
	CREATE TABLE ADDRESS(
	  ADDR1 VARCHAR(255),
	  ADDR2 VARCHAR(255),
	  ADDR3 VARCHAR(255),
	  POSTAL_CODE INT(6),
	  CLIENT_ID VARCHAR(20)
	);

3 데이터베이스 관리★★★

(1) 데이터베이스 기본 연산

데이터베이스가 가지는 기본적인 데이터 처리 기능인 Select(조회), Insert(삽입), Update(갱신), Delete(삭제)를 말한다.

▼ 데이터베이스 기본 연산

동작	유형	설명
데이터 조회	SELECT	해당 테이블을 구성하는 튜플들 중에서 전체 또는 조건을 만족하는 튜플을 검색하는 명령문
데이터 삽입	INSERT	해당 테이블에 새로운 튜플을 삽입할 때 사용하는 명령문
데이터 갱신	UPDATE	해당 테이블에 있는 튜플들 중에서 특정 튜플의 내용을 변경할 때 사용하는 명령문
데이터 삭제	DELETE	해당 테이블에 있는 튜플들 중에서 특정 튜플을 삭제할 때 사용하는 명령문

두음쌤 한마디

테이블 처리 명령어
「세인업데」
SELECT / INSERT /
UPDATE / DELETE
→ 내 친구 세인이 집에 없데

(2) 데이터베이스 기본 연산 수행절차

SQL 표준을 준수하여 저장 데이터에 대해 Select(조회), Insert(삽입), Update(갱신), Delete(삭제) 명령어를 작성한다.

▼ 기본연산 수행예시 테이블

SEQ	ADDR1	ADDR2	ADDR3	POSTAL_CODE	CLIENT_ID
1	서울시	강남구 삼성동	A아파트 101호	138021	321
2	경기도	분당구 정자동	B아파트 205호	127203	123
3	서울시	강남구 논현동	C아파트 1101호	138025	444
4	경기도	분당구 서현동	D아파트 402호	127288	555

1 데이터 조회 명령어 작성

학습 Point

명령어 예제를 보며 기본 SQL을 익혀두시면 단답형 문제 풀이에 큰 도움이 됩니다.

테이블에서 데이터를 읽기 위해서는 SELECT 명령어를 사용한다.
SELECT 다음 * 표기를 사용하면 모든 데이터를 읽어 오고 컬럼 명을 지정하면 특정 컬럼 만을 읽어 온다.

▼ 데이터 읽기 명령어 예제

	SQL 명령어	예제
전체조회	SELECT * FROM 테이블명;	SELECT * FROM ADDRESS;
지정조회	SELECT 컬럼 1, 컬럼 2, … FROM 테이블명;	SELECT SEQ, ADDR1 FROM ADDRESS;

2 데이터 삽입 명령어 작성

테이블에서 데이터를 삽입하기 위해서는 INSERT 명령어를 사용한다.

▼ 데이터 삽입 명령어 예제

SQL 명령어	예제
INSERT INTO 테이블명 VALUES (값 1, 값 2, …);	INSERT INTO ADDRESS VALUES ('서울시', '성동구 성수동', 'E아파트 101호', '109100', '660');

3 데이터 갱신 명령어 작성

테이블에서 데이터를 갱신하기 위해서는 UPDATE 명령어를 사용한다.
WHERE 조건절을 사용하여 업데이트할 데이터를 지정해야 한다.

▼ 데이터 갱신 명령어 예제

SQL 명령어	예제
UPDATE 테이블명 SET 컬럼 1 = 값 1, 컬럼 2 = 값 2, (…) WHERE 조건식;	UPDATE ADDRESS SET ADDR1 = '제주도', ADDR2 = '서귀포시 색달동', ADDR3 = 'F아파트 103동' WHERE SEQ = 321;

핵인싸 퀴즈

24 ( )은/는 각 컬럼에 적은 개수 값이 저장된 경우 선택하는 인덱스로, 수정 변경이 적을 때 유용한 인덱스이다.

25 ( )은/는 기본 테이블로부터 유도된 가상 테이블로 물리적으로 구현되어 있지 않다.

26 ( )은/는 두 릴레이션으로부터 관련된 튜플들을 결합하여 하나의 튜플로 만드는 가장 대표적인 데이터 연결 방법이다.

정답 24. 비트맵 인덱스(Bitmap Index) 25. 뷰(View) 26. 조인(Join)

4 데이터 삭제 명령어 작성

테이블에서 데이터를 삭제하기 위해서는 DELETE 명령어를 사용한다. WHERE 조건절을 사용하여 삭제할 데이터를 특정해야 한다.

▼ 데이터 삭제 명령어 예제

SQL 명령어	예제

DELETE
  FROM 테이블명
WHERE 조건식;
	DELETE
	  FROM ADDRESS
	WHERE ADDR1='서울특별시';

CHAPTER 03

네트워크 기초 활용

1 네트워크 계층 구조 ☆☆

(1) 네트워크 계층

학습 Point
네트워크는 핵심 단원 중 하나입니다. 전체적으로 눈여겨 봐두시길 추천합니다.

1 네트워크(Network)의 개념

- 원하는 정보를 원하는 수신자 또는 기기에 정확하게 전송하기 위한 기반 인프라를 네트워크라고 한다.
- 컴퓨터나 원거리 통신 장비 사이에서 메시지를 주고받는 양식과 규칙의 체계를 프로토콜(Protocol)이라 한다.

2 네트워크의 유형

네트워크의 유형은 WAN과 LAN으로 크게 분류된다.

▼ 거리에 따른 네트워크 분류

구분	설명
WAN(광대역 네트워크)	• 국가, 대륙과 같이 광범위한 지역을 연결하는 네트워크 • LAN에 비해 전송 거리가 넓음. 라우팅 알고리즘이 필요 • LAN 대비 에러율이 높고 전송 지연이 큼
LAN(근거리 네트워크)	• 한 건물 또는 작은 지역을 커버하는 네트워크

3 네트워크 데이터 교환 방식

네트워크의 데이터 교환 방식은 서킷 교환 방식과 패킷 교환 방식으로 분류된다.

▼ 네트워크 데이터 교환 방식

구분	설명
서킷 교환 방식	• 통신 사업자가 사전에 계약을 체결한 송신자와 수신자끼리만 데이터를 교환하는 방식 • 물리적 전용선을 활용하여 데이터 전달 경로가 정해진 후 동일 경로로만 전달 • 대역폭이 고정되고 안정적인 전송률을 확보
패킷 교환 방식	• 공중망을 활용하여 다수 사용자가 선로를 공유하는 방식 • 패킷이라는 단위를 사용하여 데이터를 송신하고 수신 • 컴퓨터 네트워크에서 주로 사용하는 방식

잠깐! 알고가기

대역폭(Bandwidth)
데이터를 동시에 전송할 수 있는 양을 의미한다.

패킷(Packet)
정보를 일정한 크기로 분할한 뒤 각각의 패킷에 송수신 주소 및 부가 정보를 입력한 전송 단위이다.

(2) OSI(Open System Interconnection) 7계층

1 OSI 7계층의 개념

- 국제 표준화 기구인 ISO(International Standardization Organization)에서 개발한 컴퓨터 네트워크 프로토콜 디자인과 통신을 계층으로 나누어 설명한 개방형 시스템 상호 연결 모델이다.
- 각 계층은 서로 독립적으로 구성되어 있고, 각 계층은 하위 계층의 기능을 이용하여 상위 계층에 기능을 제공한다.

2 OSI 7계층의 구조

1계층인 물리 계층부터 7계층인 애플리케이션 계층으로 정의되어 있다.

▼ OSI 7계층

계층	설명	프로토콜	전송 단위
응용 계층 (Application Layer)	• 사용자 친화 환경 제공(이메일, 웹 등)	• HTTP • SMTP • DNS • telnet • FTP • DHCP • SNMP	데이터(Data)
표현 계층 (Presentation Layer)	• 데이터 형식 설정과 부호교환, 암/복호화		
세션 계층 (Session Layer)	• 송 · 수신 간 연결 접속 및 동기제어		
전송 계층 (Transport Layer)	• 송 · 수신 간 신뢰성 있는 통신을 보장	• TCP • UDP	세그먼트 (Segment)
네트워크 계층 (Network Layer)	• 단말 간 데이터 전송을 위한 최적화된 경로 제공	• IP • ICMP • ARP • RARP	패킷(Packet)
데이터 링크 계층 (Data Link Layer)	• 오류, 흐름을 제어하여 신뢰성 있는 데이터 전송	• 이더넷 • MAC	프레임(Frame)
물리 계층 (Physical Layer)	• 실제 장비들을 연결하기 위한 연결 장치 • 0과 1의 비트 정보를 회선에 보내기 위한 전기적 신호 변환	• RS-232C	비트(Bit)

학습 Point

OSI 7계층의 계층별 주요 사항에 관해서는 확인하고 넘어가시길 당부드립니다!

두음쌤 한마디

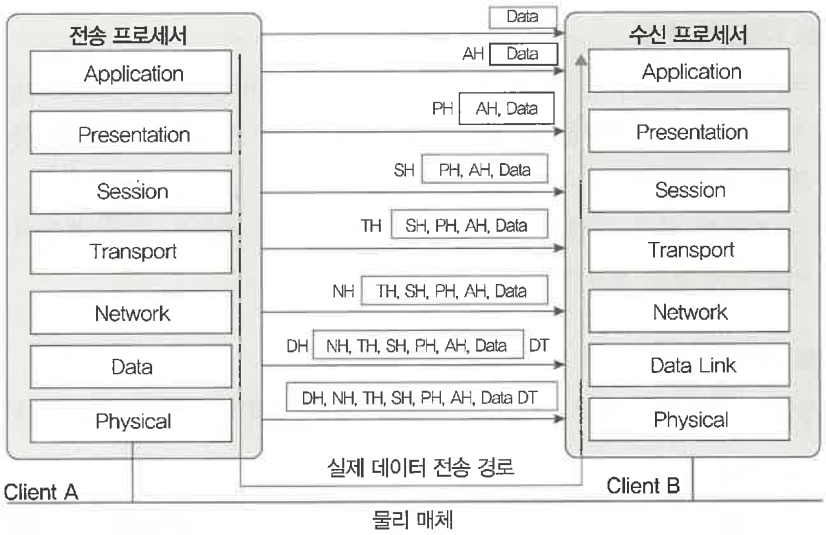
OSI 7계층

「아파서 티내다, 피나다」

Application(7) / Presentation(6) / Session(5) / Transport(4) / Network(3) / Data Link(2) / Physical(1)

→ 아파서 사람들에게 티냈는데, 피까지 났다.

3 OSI 7계층별 전달 정보



▲ OSI 7계층별 전달 정보

- 계층을 지날 때마다 헤더가 붙는데, 해당 계층의 기능과 관련된 제어 정보가 포함되어 있다.
- 제어 정보들은 모두 운영체제가 제공하는 프로토콜에 의해 송신 측에서는 계층을 지날 때마다 덧붙여서 추가되고, 수신 측에서는 계층을 지날 때마다 제거된다.

(3) 네트워크 주요장비

학습 Point

네트워크의 주요장비는 단골 문제로 출제될 가능성이 높습니다. 두음쌤의 도움을 받아 학습하세요!

두음쌤 한마디

네트워크 주요 장비

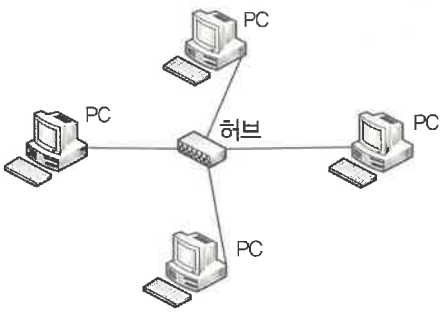
「허리브스라」

허브 / 리피터 / 브리지 / 스위치 / 라우터

→ 허라를 브스브릴라(부셔버릴라)

1 허브(Hub) – 1계층 장비

허브는 여러 대의 컴퓨터를 연결하여 네트워크로 보내거나 하나의 네트워크로 수신된 정보를 여러 대의 컴퓨터로 송신하기 위한 장비이다.

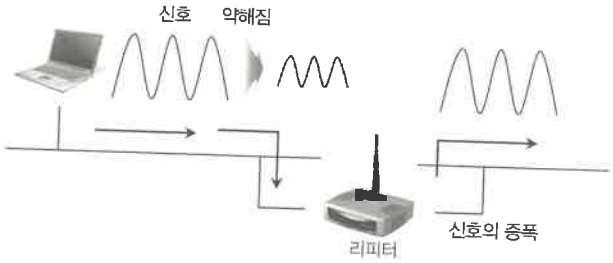


"연결 기기 간 통신 가능"

▲ 허브 개념도

2 리피터(Repeater) – 1계층 장비

리피터는 디지털 신호를 증폭시켜주는 역할을 하여 신호가 약해지지 않고 컴퓨터로 수신되도록 한다.



"디지털 신호를 증폭"

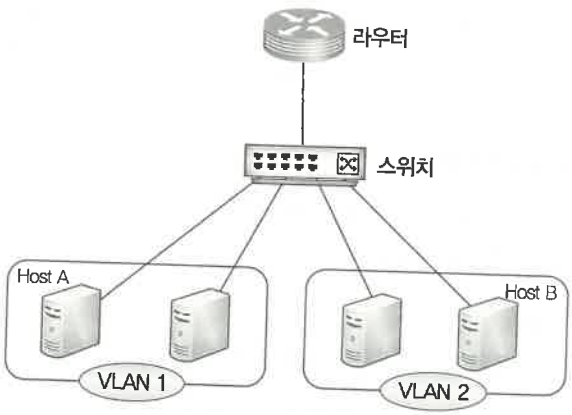
▲ 리피터 개념도

3 브리지(Bridge) – 2계층 장비

- 브리지와 스위치는 두 시스템을 연결하는 네트워킹 장치이며 두 개의 LAN을 연결하여 훨씬 더 큰 LAN을 만들어 준다.
- 브리지는 소프트웨어 방식으로 처리하기 때문에 속도가 느리고 포트들이 같은 속도를 지원한다.

4 스위치(Switch) – 2계층 장비

- 전송 중 패킷 충돌이 일어나지 않도록 MAC 주소 테이블을 이용해 목적지 MAC 주소를 가진 장비 측 포트로만 프레임을 전송하는 역할을 한다.
- 제공하는 포트 수가 수십, 수백 개로 2~3개의 포트를 제공하는 브리지보다 많다.
- 브리지는 Store and Forwarding 방식만을 사용하나, 스위치는 Cut Through 방식과 Fragment Free 방식을 같이 사용한다.



"패킷 전송 간 충돌방지"

▲ 스위치 개념도

잠깐! 알고가기

MAC(Media Access Control)
데이터 링크 계층에서 통신을 위해 네트워크 인터페이스에 할당된 고유 식별자이다. MAC 주소 확인 명령어는 'ipconfig/all'(윈도즈), 'ifconfig'(리눅스/유닉스)이다.

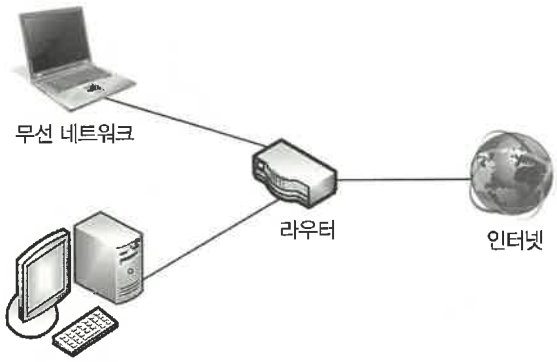
Store and Forwarding 방식
데이터를 전부 받은 후 다음 처리를 하는 방식이다.

Cut Through 방식
데이터의 목적지 주소만 확인 후 바로 전송 처리하는 방식이다.

Fragment Free 방식
프레임의 앞 64바이트만을 읽어 에러를 처리하고 목적지 포트로 전송하는 방식이다.

5 라우터(Router) – 3계층 장비

- 패킷의 위치를 추출하여, 그 위치에 대한 최적의 경로를 지정하며, 이 경로를 따라 데이터 패킷을 다음 장치로 전향시키는 장치이다.
- 라우팅 프로토콜은 최적의 경로 설정을 하여 원하는 목적지까지 지정된 데이터가 안전하게 전달되도록 한다.



▲ 라우터 개념도

2 네트워크 프로토콜 파악★★

(1) 네트워크 프로토콜

1 네트워크 프로토콜(Network Protocol)의 개념

- 네트워크 프로토콜은 컴퓨터나 원거리 통신 장비 사이에서 메시지를 주고받는 양식과 규칙의 체계이다.
- 통신 규약 또는 규칙에는 전달 방식, 통신 방식, 자료의 형식, 오류 검증 방식, 코드 변환 규칙, 전송 속도 등을 정하게 된다.

2 네트워크 프로토콜의 특징

네트워크 프로토콜의 특징으로 단편화, 재조립, 캡슐화, 연결 제어, 오류 제어, 동기화, 다중화 등이 있다.

▼ 네트워크 프로토콜의 특징

구분	설명
단편화	전송이 가능한 작은 블록으로 나누어지는 기능
재조립	단편화되어 온 조각들을 원래 데이터로 복원하는 기능

핵인싸 퀴즈

27 ( )은/는 조인의 대상 범위가 넓을 때 발생하는 임의 접근을 줄이기 위한 경우나 연결고리에 마땅한 인덱스가 존재하지 않을 경우 해결하기 위한 조인 방법이다.

28 ( )은/는 해싱 함수 기법을 활용하여 조인을 수행하는 방식이다.

29 ( )은/는 SQL문 안에 포함된 또다른 SQL문이다.

정답 27. 정렬 합병 조인(Sort-Merge Join) 28. 해시 조인(Hash Join) 개념 29. 서브 쿼리(Sub-Query)

구분	설명
캡슐화	상위 계층의 데이터에 각종 정보를 추가하여 하위 계층으로 보내는 기능
연결 제어	데이터의 전송량이나 속도를 제어하는 기능
오류 제어	전송 중 잃어버리는 데이터나 오류가 발생한 데이터를 검증하는 기능
동기화	송신과 수신 측의 시점을 맞추는 기능
다중화	하나의 통신 회선에 여러 기기들이 접속할 수 있는 기능

3 IP(Internet Protocol)

- 대표적인 프로토콜인 IP(Internet Protocol) 주소는 전 세계 컴퓨터에 부여되는 유일한 식별자이다.
- IPv4는 인터넷 초기부터 현재까지 쓰고 있는 주소체계이며 000.000.000.000과 같이 12자리로 표시하며 약 43억 개를 부여할 수 있다.
- 최근에는 디바이스의 증가로 IPv4가 가진 주소의 양이 부족할 수 있어 IPv6를 공표하였다.

▼ IPv4와 IPv6 차이 비교

구분	IPv4	IPv6
주소 길이	32bit	128bit
표시 방법	8비트씩 4부분으로 나뉜 10진수 (192.168.10.1)	16비트씩 8부분으로 나뉜 16진수 (2001:9e76:..:e11c)
주소 개수	약 43억 개	약 31조 개
주소 할당	A, B, C, D 등 클래스 단위로 비순차적 할당(비효율적)	네트워크 규모 및 단말기 수에 따른 순차적 할당(효율적)
품질 제어	품질보장 곤란	등급별, 서비스별로 패킷 구분 가능해 품질보장 용이
헤더 크기	가변	고정
전송 방식	유니캐스트, 멀티캐스트, 브로드캐스트	유니캐스트, 멀티캐스트, 애니캐스트

(2) TCP/IP 프로토콜

1 TCP/IP 프로토콜의 개념

- TCP/IP란 TCP와 IP 프로토콜만을 지칭하는 것이 아니라 UDP, ICMP 등 관련된 프로토콜을 통칭한다.
- TCP와 UDP로 구분되는 프로토콜은 전송 계층에서 응용 계층과 인터넷 계층 사이의 통신을 담당한다.

잠깐! 알고가기

유니캐스트(Unicast)
고유 주소로 식별된 하나의 네트워크 목적지에 1:1로 (one-to-one) 트래픽 또는 메시지를 전송하는 전송 기술이다.

멀티캐스트(Multicast)
하나 이상의 송신자들이 특정한 하나 이상의 수신자들에게 데이터를 전송하는 방식으로 인터넷 화상 회의 등의 응용에서 사용한다.

브로드캐스트(Broadcast)
하나의 송신자가 같은 서브 네트워크 상의 모든 수신자에게 데이터를 전송하는 전송 기술이다.

애니캐스트(Anycast)
단일 송신자로부터의 데이터그램들을 토폴로지 상의 잠재적인 수신자 그룹 안에서 가장 가까운 노드로 연결시키는 전송 기술이다.

두음쌤 한마디

IPv4 전송방식
「유멀브」
유니캐스트, 멀티캐스트, 브로드캐스트

IPv6 전송방식
「유멀애」
유니캐스트, 멀티캐스트, 애니캐스트

잠깐! 알고가기

ICMP(Internet Control Message Protocol)
인터넷 통신 환경에서 오류에 관한 경고 또는 알림과 관련된 메시지를 전달하는 목적의 프로토콜을 말한다.

Internet 5 layer

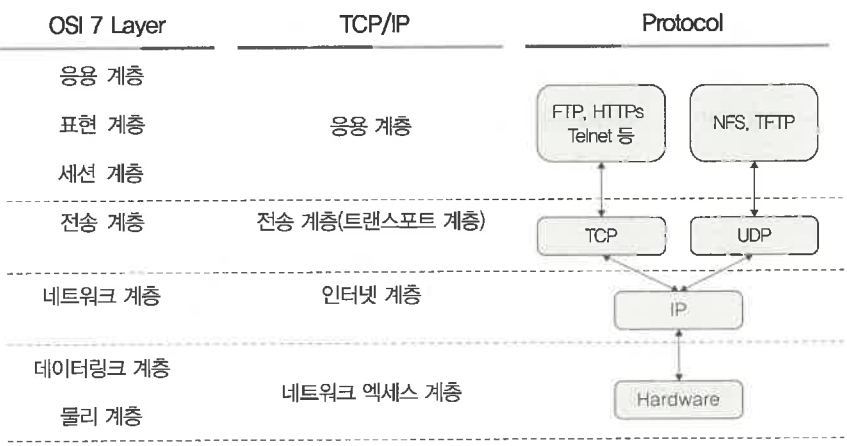
App

Transport

Network

Data link

Physic



▲ TCP IP 계층과 프로토콜

2 TCP(Transmission Control Protocol)

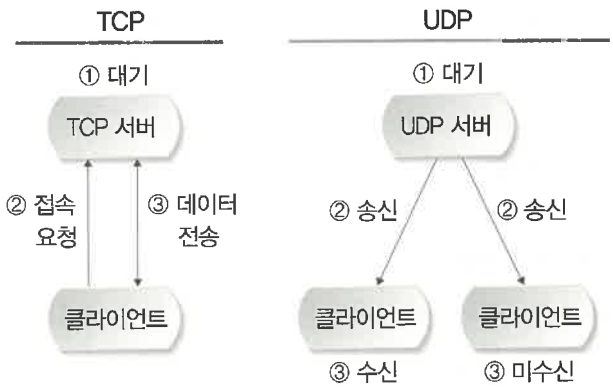
- 전송 계층에 위치하면서 근거리 통신망이나 인트라넷, 인터넷에 연결된 컴퓨터에서 전송되는 데이터를 안정적이고 에러 없이 교환할 수 있게 해주는 프로토콜이다.
- TCP는 IP의 핵심 프로토콜 중 하나로, IP와 함께 TCP/IP라는 명칭으로 사용된다.

3 UDP(User Datagram Protocol)

비 연결형 서비스를 지원하는 전송 계층 프로토콜로서, TCP와 달리 정보를 주고받을 때 보내거나 받는다는 신호 절차를 거치지 않고 보내는 쪽에서 데이터그램 단위로 데이터를 일방적으로 전달하는 통신 프로토콜이다.

잠깐! 알고가기

데이터 그램(Datagram)
송신지에서 수신지로 배달되는 충분한 정보를 갖는 독립적인 패킷이다.

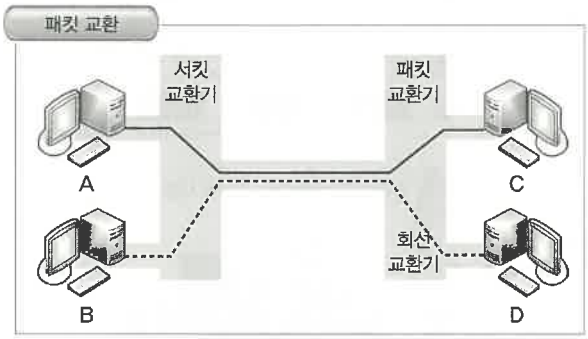


▲ TCP/UDP 간 전송방식 차이 확인

3 네트워크 핵심 알고리즘★★

(1) 패킷 교환(Packet Switching)

- 네트워크 통신 방식 중 가장 많이 사용하는 통신 방식으로, 작은 블록의 패킷으로 데이터를 전송하며 데이터를 전송하는 동안만 자원을 사용하도록 하는 기법이다.
- 패킷은 동일한 목적지에 도달하기 위해 많은 서로 다른 동적 경로를 사용한다.
- X.25, 프레임 릴레이 및 ATM과 같은 다양한 기술들을 사용한다.



▲ 패킷 교환 개념

1 X.25 프로토콜

- 전기 통신 국제기구인 ITU-T에서 관리 감독하는 프로토콜이다.
- X.25는 패킷이라고 불리는 데이터 블록을 사용하여 대용량의 데이터를 다수의 패킷으로 분리하여 송신하며, 수신 측에서는 다수의 패킷을 결합하여 원래의 데이터로 복원한다.
- X.25는 OSI 7계층 중 1~3계층을 담당하고 있다.
- 초기에 에러 제어나 흐름 제어를 위한 복잡한 기능으로 인해 성능이 좋지 않아 현재는 프레임 릴레이나 ISDN, ATM 등 고속망으로 대체되었다.

2 프레임 릴레이(Frame Relay) 프로토콜

- 데이터 링크 계층에서 동작하며 전용선보다 저렴하고 적은 복잡성과 간단한 구현 방법을 가진 대표적인 WAN 프로토콜이다.
- 프레임 릴레이 망에서는 가상회선을 사용하여 여러 개의 물리적 회선을 하나의 회선으로 대체해서 사용이 가능하다.
- 프레임 릴레이는 사용자의 요청에 따라 유연한 대역폭을 할당한다.
- 프레임 릴레이는 1~2계층만을 담당한다.

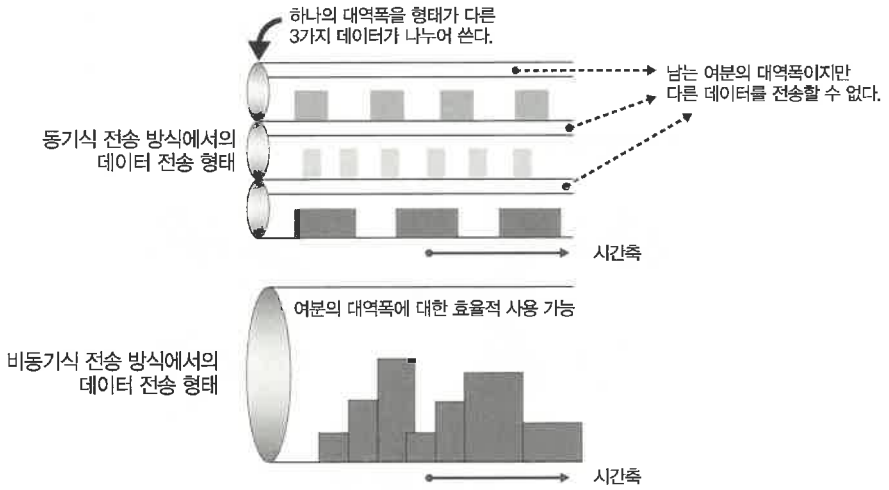
잠깐! 알고가기

가상회선(Virtual Circuit)
가상회선은 패킷 교환망에서 통신 경로가 논리적으로 고정되어 물리적으로 고정된 전송 경로상에서 통신하는 것과 같은 효과를 주는 방식이다.

대역폭(Bandwidth)
대역폭은 주어진 데이터를 전송하는 데 필요한 주파수 폭이다. 따라서 전송해야 할 데이터의 양이나 시스템의 성능에 의해 대역폭이 결정되는데, 동일한 시간에 많은 정보를 보내기 위해서는 대역폭이 넓어야 한다.

3 ATM(Asynchronous Transfer Mode) 프로토콜

- 광대역 전송에 쓰이는 비동기 전송 방식이며, 동기화를 맞추지 않아 보낼 데이터가 없는 사용자의 슬롯은 다른 사람이 사용할 수 있도록 하여 네트워크 상의 효율성을 높인 기법이다.
- ATM 망은 연결형 회선이기 때문에 하나의 패킷을 보내 연결을 설정하게 되고 이후 데이터 전송이 이루어진다.
- ATM은 OSI 7계층과는 다른 고유한 참조 모델을 가지고 있다.



▲ ATM 방식 개념도

회선 교환 방식

(2) 서킷 교환(Circuit Switching)

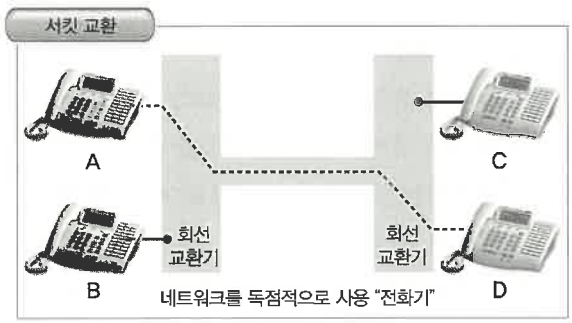
학습 Point

패킷 교환과 서킷 교환은 실기 문제로 출제될 가능성이 높은 내용입니다. 차이점을 잘 이해하고 넘어가야 합니다!

- 패킷 교환과 달리 네트워크 리소스를 특정 채널이 독점하도록 하는 것을 서킷 교환 방식이라고 한다.
- 네트워크를 독점적으로 사용하기 때문에 전송이 보장된다는 특징이 있다.
- 서킷 교환은 서킷을 확보하기 위한 작업을 진행하고 데이터를 전송하며 서킷을 닫는 프로세스로 진행되며 작업 기간 동안 다른 기기들은 해당 경로를 사용할 수 없다.

학습 Point

서킷 교환(Circuit Switching)은 회선 교환이라고도 불립니다.



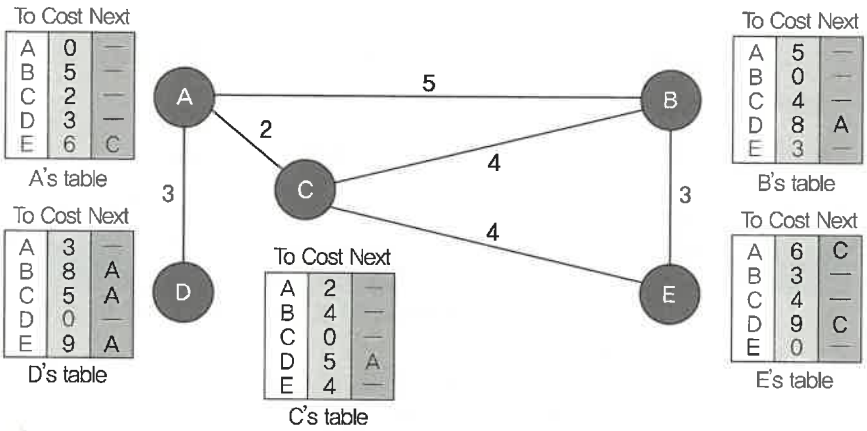
▲ 서킷 방식 개념

(3) 라우팅 알고리즘

송신 측으로부터 수신 측까지 데이터를 전달하는 과정에서 최적의 경로를 산출하기 위한 로직이 라우팅 알고리즘이다.

1 거리 벡터 알고리즘(Distance Vector Algorithm)

- 라우터와 라우터 간의 최단 경로를 찾고 그 최적 경로를 이용할 수 없을 경우에 다른 경로를 찾는 알고리즘이다.
- 각 라우터가 업데이트될 경우마다 전체 라우팅 테이블을 보내라고 요청하지만 수신된 경로 비용 정보는 이웃 라우터에게만 보내진다. 링크 상태 라우팅 알고리즘보다 계산 면에서 단순하다.



▲ 거리 벡터 알고리즘

- 각 라우터는 모든 목적지 라우터에 대한 최소 경로 비용 테이블(Vector)를 유지한다.
- 초기 라우터의 경로 비용 테이블은 자신이 직접 연결된 인접 라우터에 대한 비용 정보만 유지한다.
- 벨만-포드 알고리즘(Bellman-Ford Algorithm)을 이용한다.

2 링크 상태 알고리즘(Link State Algorithm)

- 라우터와 라우터 간의 모든 경로를 파악한 뒤 대체 경로를 사전에 마련해 두는 알고리즘이다.
- 링크 상태 알고리즘을 사용하면 네트워크를 일관성 있게 파악할 수 있으나 거리 벡터 알고리즘에 비하여 계산이 더 복잡하고 트래픽을 광범위한 범위까지 전달해야 한다.

핵인싸 퀴즈

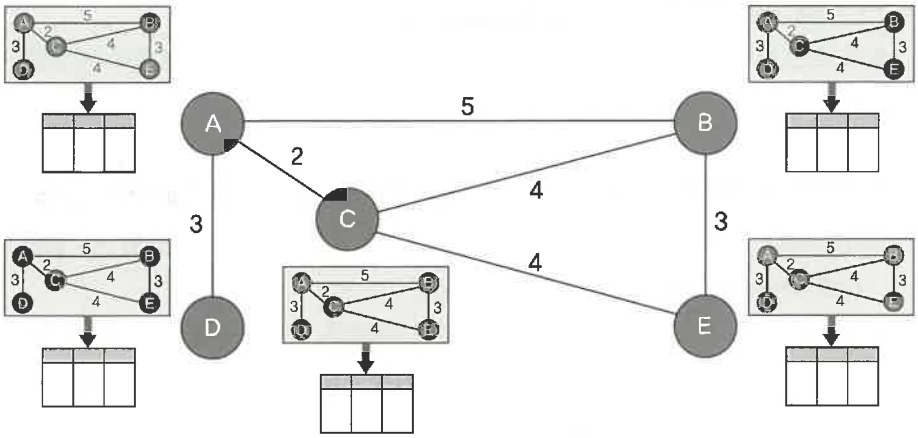
30 ( )은 서브 쿼리의 종류 중 하나로 서브 쿼리가 메인 쿼리의 컬럼을 가지고 있지 않은 형태이며, 메인 쿼리에 서브 쿼리에서 실행된 결과 제공하는 용도로 사용하는 쿼리이다.

31 ( )은/는 테이블을 집합 개념으로 보고, 두 테이블 연산에 집합 연산자를 사용하는 방식이다.

정답 30. 비연관 서브 쿼리(Un-Correlated Sub-Query) 31. 집합 연산자(Set Operator)

두음쌤 한마디

라우팅 프로토콜(3계층)
「ROBI(로비)」
RIP / OSPF / BGP / IGRP
→ 호텔 로비



▲ 링크 상태 알고리즘

- 각 라우터가 전체 네트워크의 구성과 링크 상태정보를 가지고 모든 목적지까지 최소 비용 경로를 계산한다.
- 다익스트라 알고리즘(Dijkstra Algorithm)을 이용한다.

잠깐! 알고가기

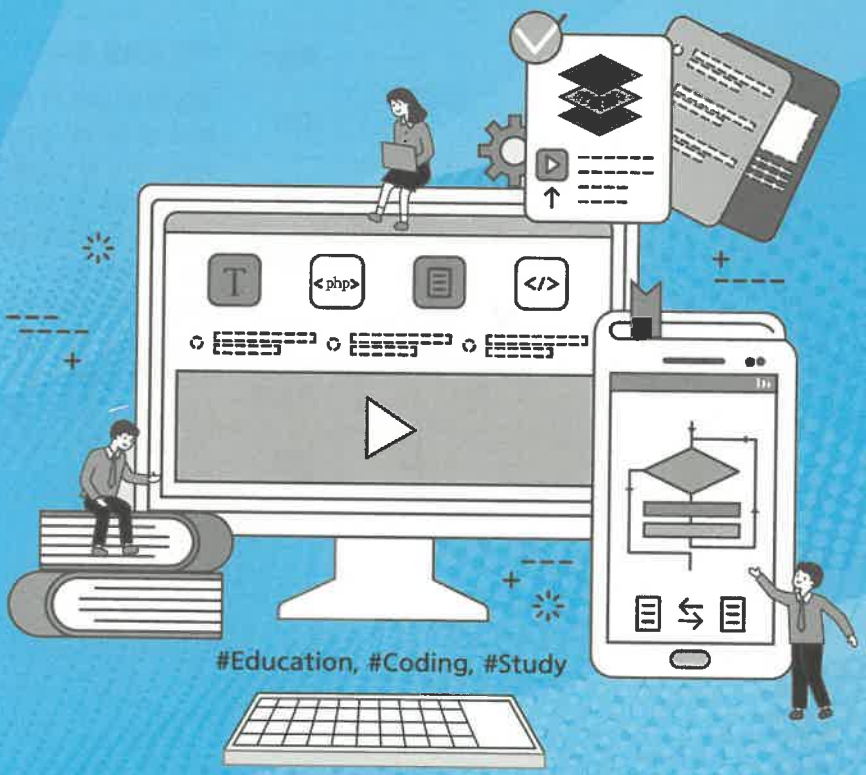
홉 카운트(Hop Count)
데이터가 출발지와 목적지 사이에서 통과해야 하는 중간 장치들의 개수를 가리킨다.

▼ 라우팅 프로토콜의 종류

프로토콜	설명
RIP	• 홉 카운트(Hop Count)를 Metric으로 설정해 이용하여 최적의 경로를 설정하는 소규모용 프로토콜 • 거리 벡터 알고리즘 사용, 30초 주기로 정보갱신 • Routing Information Protocol
OSPF	• RIP의 단점을 개선하기 위해 링크 상태 알고리즘 기반으로 최단 경로를 찾는 프로토콜 • 변경 정보가 전체 라우터에 동일하게 유지 • Open Shortest Path First
BGP	• 규모가 큰 네트워크의 상호 연결을 위해 사용하는 프로토콜 • 초기 연결 시 전체 경로를 나타내는 라우팅 테이블을 교환하고 이후에는 변화된 정보만을 교환 • Border Gateway Protocol
IGRP	• RIP와 동일한 거리 벡터 기반의 내부용 프로토콜 • Interior Gateway Routing Protocol

Ⅱ

프로그래밍 언어 활용



CHAPTER

01 알고리즘 구현

1 알고리즘☆☆

학습 Point
알고리즘은 개정 전 정보처리기능사 시험 단골 문제입니다. 주의 깊게 차근차근 봐주시길 당부드립니다.

(1) 알고리즘(Algorithm) 개념

알고리즘이란 문제를 해결하기 위한 일련의 절차나 방법이다.

(2) 순서도

1 순서도(Flow Chart) 개념

순서도는 도형과 화살표를 이용하여 알고리즘을 도식적으로 표현하는 방법이다.

√	단말	순서도의 시작과 끝
	흐름선	작업 흐름을 명시
	준비	작업 단계 시작 전 준비 (변수 및 초기치 선언 등)
√	처리	처리하여야 할 작업을 명시 (변수에 계산 값 입력 등)
	입출력	일반적인 데이터의 입력 또는 결과의 출력
√	판단	조건에 따라 흐름선을 선택 (일반적으로 참, 거짓 구분)
	프린트	프린터를 이용한 출력 (서류 등의 지면의 출력)
	결합	기본 흐름선에 따른 흐름선 합류

▲ 순서도 기호

2 순서도 변수

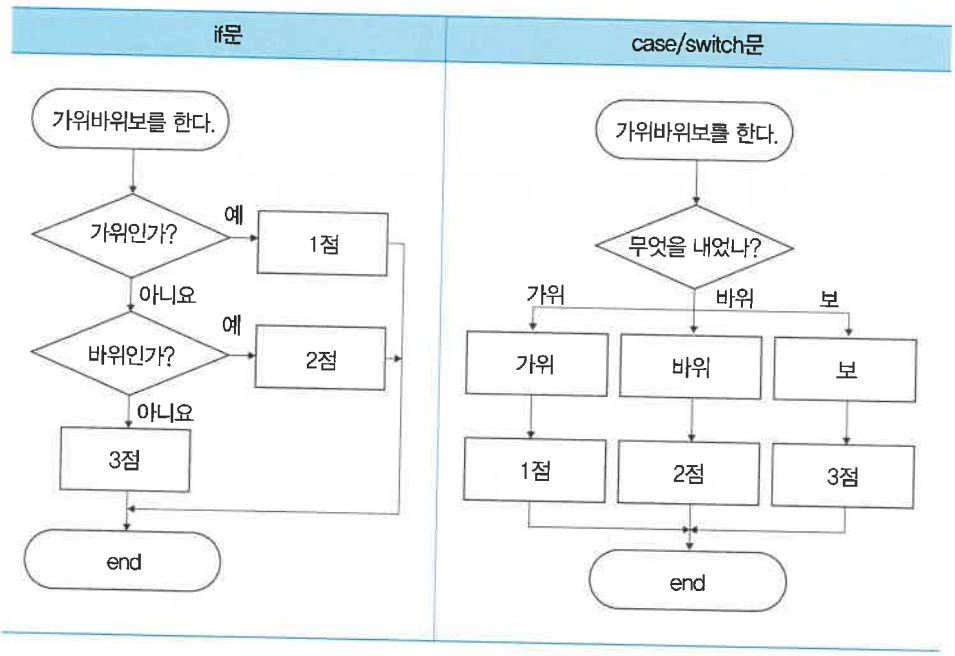
- 변수는 어떤 값을 주기억장치에 기억하기 위해서 사용하는 공간이다.
- 순서도 변수는 준비 기호를 이용하여 선언하고 초깃값을 설정한다.
- 변수에는 일반 변수와 배열 변수가 있으며 아래는 사용 예제이다.

▼ 변수 사용 예

구분	예제
일반 변수	단일 메모리 공간에 할당하고 접근이 가능한 변수 예 EVENCNT
배열 변수	연속된 메모리 공간을 할당하고 접근이 가능한 변수 예 A(10)

3 순서도 조건문

- 순서도를 이용한 조건문에는 if문과 case/switch문이 있다.
- if문은 판단 기호를 이용하여 '예'와 '아니오'로 이동한다.
- case/switch문은 판단 기호의 결괏값에 따라 이동한다.



▲ 순서도 if문 case/switch문

핵인싸 퀴즈

1 (　　)은/는 사용자로 하여금 컴퓨터의 하드웨어를 보다 쉽게 사용할 수 있도록 인터페이스를 제공해 주는 소프트웨어이다.

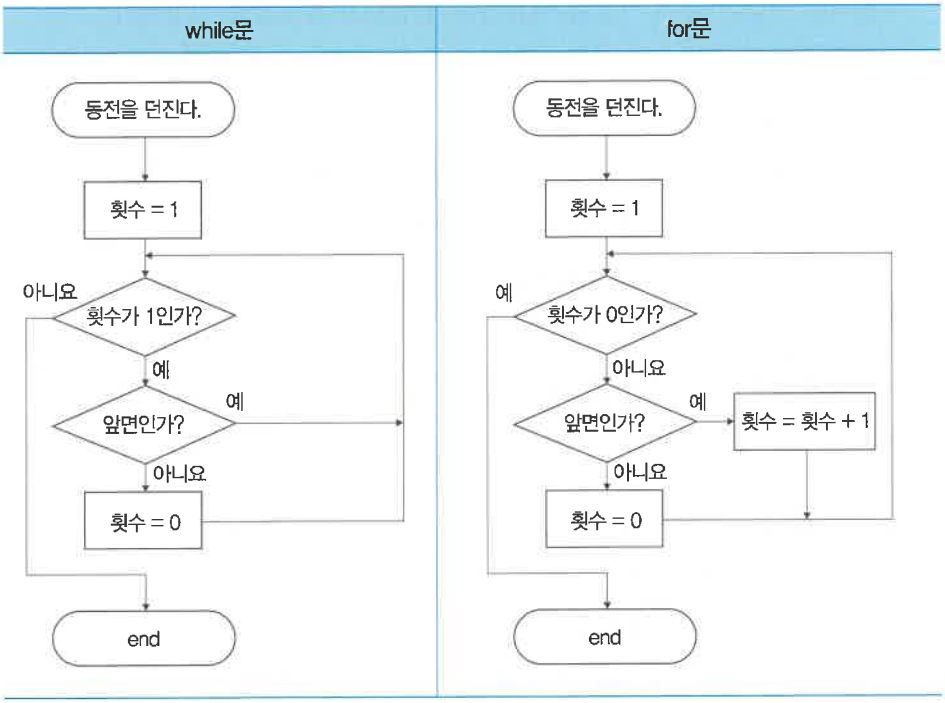
2 (　　)은/는 MS-DOS의 멀티태스킹 기능과 GUI 환경을 제공하는 응용 프로그램으로서, 마이크로소프트(Microsoft)가 개발한 운영체제이다.

3 (　　) 운영체제는 애플이 매킨토시용으로 개발한 그래픽 사용자 인터페이스(GUI) 운영체제이다.

정답 1. 운영체제 2. 윈도즈 3. 맥(Mac)

4 순서도 반복문

for문, while문을 순서도를 이용하여 표현한다.



▲ 순서도 반복문

2 절차형 프로그래밍 언어로 알고리즘 처리★★★

학습 Point

절차형 프로그래밍 언어로 알고리즘 처리를 위해서는 C언어 변수, 조건문, 반복문, 연산자, 배열에 대한 이해가 선행되어야 합니다.

- 절차형 프로그래밍 언어란 컴퓨터에 저장된 명령어들이 순차적으로 실행되는 프로그래밍 언어이다.
- 절차형 프로그래밍 언어에는 대표적으로 C언어가 있다.

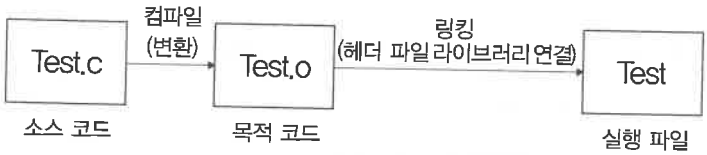
(1) C언어를 이용한 알고리즘 처리

1 C언어 개념

1972년에 벨 연구소의 데니스 리치가 개발한 프로그래밍 언어로, UNIX 운영체제 구현에 사용되는 언어이다.

2 C언어 구조

- C언어는 헤더 파일과 소스 코드 파일로 구성되어 있다.
- 소스 코드 파일을 컴파일과 링킹을 거쳐 실행 파일을 생성한다.



▲ C언어 컴파일 및 링킹 과정

- C언어는 입력값, 함수 내 연산, 결괏값 출력의 함수 형태로 되어 있으며, 모든 문장의 끝에는 세미콜론을 붙여야 한다.
- 문장을 작성할 때는 가독성을 위해 들여쓰기를 한다.

▼ C언어 소스 코드 구조

```
#include <stdio.h>

void main()
{
    문장;
    return;
}
```

구분	내용
#include	• '#include' 구문으로 헤더 파일을 포함시킴 • stdio.h는 표준입출력(STanDard Input Output)의 약자이고 헤더 파일의 확장자는 h임 예 #include <stdlib.h> 표준 라이브러리(Standard Library) 헤더 파일을 포함시킴.

void main()
{
 	
}
	• void는 반환 타입, main은 함수 이름
	• main은 프로그램에서 유일하게 사용되며 사용자 정의 함수도 추가 가능
	• '('와 ')' 사이에는 명령문 인수를 전달받으며 main 함수는 생략이 가능
	• 중괄호 '{'와 중괄호 '}'를 코드 블록이라고 하며, 코드 블록 안에 코드를 작성

return;	• 함수명 앞에 있는 반환 타입이 void인 경우는 반환 값이 없음을 의미하며, 'return;'이라고 작성 • void가 아닌 경우에는 반환 값이 있음을 의미하며 'return 값;'이라고 작성

반환 값이 없는 경우 예	반환 값이 있는 경우 예

`void func() {`
`  return;`
`}`
	`int func() {`
	`  return 0;`
	`}`

잠깐! 알고가기

헤더 파일(Header File)
컴파일러에 의해 소스 코드에 자동으로 포함되는 파일이다.

소스 코드(Source Code)
사용자가 작성하는 파일로 확장자가 .c인 파일이다.

컴파일(Compile)
사람이 작성한 소스 코드 파일을 컴퓨터가 이해할 수 있는 목적 코드로 변환하는 과정이다.

링킹(Linking)
변환된 목적 코드와 헤더 파일, 라이브러리 등을 연결하는 과정이다.

들여쓰기
프로그램을 작성하는 경우 가독성을 위해 들여쓰기를 한다. 예를들어 a라는 정수형 변수를 선언하는 경우 탭 또는 스페이스로 들여쓰기 후 선언한다.

들여쓰기 전	들여쓰기 후
`void main() {`	`void main() {`
`int a;`	`  int a;`
`}`	`}`

잠깐! 알고가기

라이브러리(Library)
자주 사용되는 함수에 대해 미리 컴파일된 파일이다.

핵인싸 퀴즈

4 ( )은/는 썬 마이크로시스템즈의 제임스 고슬링을 중심으로 개발한 객체 지향적 프로그래밍 언어이다.

5 ( )은/는 응용 프로그램에서 사용할 수 있도록, 운영체제나 프로그래밍 언어가 제공하는 기능을 제어할 수 있게 만든 인터페이스이다.

6 ( )은/는 속도가 빠른 장치와 느린 장치 사이에서 속도 차에 따른 병목 현상을 줄이기 위한 범용 메모리이다.

정답 4. 자바(Java) 5. API 6. 캐시 메모리

3 C언어 주석

- 주석은 도움말로 주로 소스 코드의 이해를 돕기 위해 사용된다.
- 소스 코드가 컴파일되는 과정에서 제거된다.

▼ C언어 주석

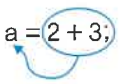
주석	내용

//	한 줄 단위의 주석
	예 세미콜론 뒤에 주석 작성
	`a = 5;  // 5를 a에 대입`
	예 문장 시작 부분에 주석 작성
	`// 5를 a에 대입`
	`a = 5;`

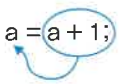
/* */	여러줄 단위의 주석
	예
	`/*`
	`  5를 a에 대입`
	`  3을 b에 대입`
	`*/`
	`a = 5;`
	`b = 3;`

4 C언어 변수

- 변수는 저장하고자 하는 어떠한 값이 있을 때, 그 값을 주기억장치에 기억하기 위한 공간을 의미한다.



2와 3을 더한 결괏값 5를 변수 a에 대입함



오른쪽 a와 1을 더한 결괏값을 왼쪽 a에 대입함

▲ 변수 계산 결과의 대입

• 변수를 사용하기 위해서는 아래와 같이 변수를 선언 후 사용한다.

▼ C언어 변수 예제 소스 코드

```
#include <stdio.h>

void main()
{
  int a;
  a = 2;
  a = 2 + 3;
  a = a + 1;
}
```

- 정수형 변수 a를 선언
- 2를 a에 대입
- 2와 3을 더한 결괏값을 a에 대입
- 왼쪽 a와 1을 더한 결괏값을 오른쪽 a에 대입

5 C언어 조건문

• 조건문으로 if문이 있으며 if문은 조건이 참일 경우 문장이 실행되고, 조건이 거짓일 경우에는 문장이 실행되지 않는다.

• else if문과 else 구문을 이용하면 2개 이상의 조건문을 사용할 수 있다.

▼ C언어 if문 예제 소스 코드

```
#include <stdio.h>

void main()
{
  int a = 1;
  int b = 2;

  if( a > b ) {
    printf("a가 b보다 크다");
  } else if ( a < b ) {
    printf("a가 b보다 작다");
  } else {
    printf("a와 b가 같다");
  }
}
```

- 정수형 변수 a를 선언하고 1로 초기화
- 정수형 변수 b를 선언하고 2로 초기화
- a가 b보다 크면
- "a가 b보다 크다"를 화면 출력
- a가 b보다 작으면
- "a가 b보다 작다" 화면 출력
- 그 외일 경우
- "a와 b가 같다" 화면 출력

출력 : a가 b보다 작다

학습 Point

C언어에서 알고리즘을 구현하기 위해서는 명령문(조건문, 반복문)과 연산자(산술, 관계, 논리)가 반드시 사용되며 2. 프로그래밍 언어 활용에서 상세하게 다룹니다.

잠깐! 알고가기

데이터 타입(Data Type)
정수, 실수 같은 여러 종류의 데이터를 식별하는 형태이다.

6 C언어 반복문

• 반복문에는 while문과 for문이 있다.

▼ C언어 반복문

while문	for문

while (조건식) {
  문장;
}
	for (초깃값 ; 조건식 ; 증감 값) {
	  문장;
	}

조건식이 참일 경우 문장을 반복 실행한다.	• 초깃값이 실행되고 조건식이 참일 경우 문장을 실행하고, 증감 값이 증가 또는 감소한다. • 증가 또는 감소된 값이 조건식을 참으로 만족하면 문장이 반복 실행된다.

7 C언어 연산자

• 연산자는 데이터 처리를 위해 연산을 표현하는 기호이다.

• 연산자에는 대표적으로 산술 연산자, 관계 연산자, 논리 연산자가 있다.

① 산술 연산자

• 산술 연산자에는 대표적으로 '+', '−', '*', '/' 연산자가 있다.

```
#include <stdio.h>

void main()
{
  int a = 2;
  int b = 2;

  int r1 = a + b;
  int r2 = a - b;
  int r3 = a * b;
  int r4 = a / b;
}
```

• 정수형 변수 a를 선언하고 2로 초기화
• 정수형 변수 b를 선언하고 2로 초기화

• a와 b를 더한 결과 4를 정수형 변수 r1에 대입함
• a와 b를 뺀 결과 0을 정수형 변수 r2에 대입함
• a와 b를 곱한 결과 4를 정수형 변수 r3에 대입함
• a와 b를 나눈 결과 1을 정수형 변수 r4에 대입함

② 관계 연산자

• 관계 연산자는 두 피연산자 사이의 크기를 비교하는 연산자이다.

• 관계 연산자에는 '<', '<=', '>', '>=', '==', '!=' 가 있다.

▼ C언어 관계 연산자 예제 코드

```
#include <stdio.h>

void main()
{
  int a = 1;
  int b = 2;

  if( a > b ) {
    printf("a는 b보다 크다.");
  } else if ( a < b ) {
    printf("a는 b보다 작다.");
  }
}
```

- 정수형 변수 a를 선언하고 1로 초기화
- 정수형 변수 b를 선언하고 2로 초기화
- a가 b보다 크면
- "a는 b보다 크다."를 출력
- a가 b보다 작으면
- "a는 b보다 작다."를 출력

출력 결과 : a는 b보다 작다.

③ 논리 연산자

- 논리 연산자는 두 조건식의 참과 거짓에 따라 전체 조건식의 참과 거짓을 결정하는 연산자이다.
- 논리 연산자에는 '&&', '||'가 있다.

▼ C언어 논리 연산자 예제 코드

```
#include <stdio.h>

void main()
{
  int a = 1;
  int b = 2;

  if( a == 1 && b == 2 ) {
    printf("a는 1이고 b는 2이다.");
  }
}
```

- 정수형 변수 a에 1 대입
- 정수형 변수 b에 2 대입
- a가 1과 같고 b가 2와 같으면
- "a는 1이고 b는 2이다."를 출력

출력 결과 : a는 1이고 b는 2이다.

잠깐! 알고가기

printf() 함수

- 화면에 출력을 위한 함수로 출력 형식을 지정할 수 있다.
- '('와 ')' 사이에 % 기호를 사용한 서식 지정자와 변수를 지정하여 출력 서식을 지정한다.

서식 지정자	내용
%d	10진 정수(decimal) 출력
%s	문자열(string) 출력
%f	부동소수점(float, double) 출력

핵인싸 퀴즈

7 ( )은/는 현재 디렉터리 이름을 보여주거나 변경하는 윈도즈 CLI 명령어이다.

8 ( )은/는 하나 이상의 파일을 다른 위치로 복사 변경하는 윈도즈 CLI 명령어이다.

9 ( )은/는 CMD 프로그램을 종료하는 윈도즈 CLI 명령어이다.

정답 7. CD 8. COPY 9. EXIT

8 C언어 배열

• 동일한 타입의 변수를 여러 개 만드는 경우 배열을 사용한다.

– 메모리에 연속적으로 할당이 됨
– 데이터는 정수 1, 2, 3, 4임
– 배열은 '배열명 + [+ 인덱스 번호 +]' 형태로 접근 가능
– 인덱스 번호는 0부터 시작
– a[0]에서 'a'는 배열명, 0은 인덱스 번호임
– a[0]은 1, a[1]은 2, a[2]는 3, a[3]은 4임

▲ C언어 배열

• 동일한 정수형 변수 타입을 연속으로 4개 선언 및 할당을 배열로 한다.

▼ C언어 배열 예제 코드

```
#include <stdio.h>

void main()
{
  int arr[4] = {1,2,3,4};

  for( i=0; i<4;i++) {
    printf("%d", arr[i]);
  }
}
```

• 배열 변수 arr을 선언하고 1, 2, 3, 4로 초기화

• for문의 인덱스 i가 0부터 5까지 증가
• 배열 화면에 출력한다.

출력 결과 : 1234

(2) C언어를 이용한 알고리즘 처리 예제

1 짝수 홀수 판별 알고리즘

• 다음은 1부터 10까지 홀수, 짝수의 합을 계산하는 알고리즘이다.

▼ C언어 짝수 홀수 판별 알고리즘

```
#include <stdio.h>
void main()
{
  int i;
  int odd_sum=0;
  int even_sum=0;

  for(i=1;i<=10;i++) {
    if( i%2 == 0 ) {
      even_sum += i;
    } else {
      odd_sum += i;
    }
  }

  printf("%d %d",
     odd_sum, even_sum);
}
```

- for문에서 사용할 변수 i 선언
- 홀수 합을 저장할 변수 odd_sum 선언 및 0으로 초기화
- 짝수 합을 저장할 변수 even_sum 선언 및 0으로 초기화
- 1부터 10까지 1씩 순차적 증가하며 반복
- i와 2를 나머지 연산하여 나머지가 0이면
- even_sum에 i 값을 누적
- i와 2를 나머지 연산하여 나머지가 0이 아니면
- even_sum에 i 값을 누적
- 홀수의 합과 짝수의 합을 화면에 출력

출력 결과 : 30 25

2 스위칭 변수를 이용한 합계 알고리즘

- 다음은 1부터 10까지 '+' 연산자와 '−' 연산자를 번갈아 가면서 "1−2+3−4+5−6+7−8+9−10"을 계산하는 알고리즘이다.

▼ C언어 스위칭 변수를 이용한 합계 알고리즘

```
#include <stdio.h>

void main()
{
  int i;
  int sw=0;
  int sum=0;

  for(i=1;i<=10;i++)
  {
    if(sw == 0) {
      sum += i;
      sw = 1;
    } else {
      sum -= i;
      sw = 0;
    }
  }
  printf("%d", sum);
}
```

- for문에서 사용할 변수 i 선언
- 스위칭 변수 sw 선언 및 0으로 초기화
- 누적 합을 저장하는 변수 sum 선언 및 0으로 초기화
- 1부터 10까지 1씩 순차적 증가하며 반복
- 스위칭 변수 sw가 0과 같으면
- sum 변수와 i값을 더한 결과를 sum에 대입
- 스위칭 변수 sw에 1을 대입
- 스위칭 변수 sw가 0과 같지 않으면
- sum 변수와 i값을 뺀 결과를 sum에 대입
- 스위칭 변수 sw에 0을 대입
- 화면에 결과를 출력

출력 결과 : −5

핵인싸 퀴즈

10 ( )은/는 현재 작업 중인 디렉터리 정보를 출력하는 리눅스/유닉스 CLI 명령어이다.

11 ( )은/는 현재 파일 혹은 디렉터리를 복사하는 리눅스/유닉스 CLI 명령어이다.

12 ( )은/는 현재 파일이나 디렉터리를 삭제하는 리눅스/유닉스 CLI 명령어이다.

정답 10. pwd 11. cp 12. rm

3 객체지향형 프로그래밍 언어로 알고리즘 처리★★★

• 객체지향형 프로그래밍 언어에는 대표적으로 자바(Java)가 있다.

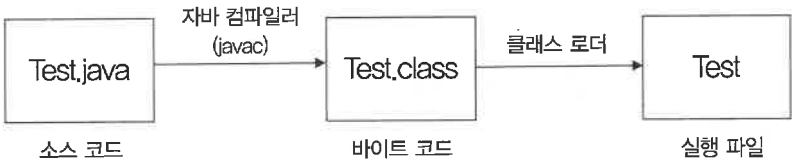
(1) 자바를 이용한 알고리즘 처리

1 자바 개념

• 썬 마이크로 시스템즈에서 1995년에 개발한 객체지향 프로그래밍 언어이다.

2 자바 프로그램 구조

• 자바 프로그램은 소스 코드와 라이브러리 파일로 구성되어 있다.
• 자바 프로그램은 C언어와 달리 확장자가 java인 소스 코드를 컴파일러가 바이트 코드인 class 파일로 변경 후 클래스 로더에 의해 실행된다.



▲ Java 프로그램 컴파일 과정

• 자바 프로그램은 클래스 형태로 구성되어 있다.
• 클래스 내부에 main 함수를 포함하고 있다.

▼ 자바 프로그램 구조

```
public class Test1 {
  public static void main(String[] args) {
    문장;
  }
}
```

구분	내용

public class Test1 {
 	
}
	• 접근제어 지시자 4가지 유형

구분	내용
private	• 클래스 내부에서만 접근을 허용
default	• 클래스 내부와 동일 패키지에서만 접근을 허용 • 아무런 선언을 하지 않으면 default임
protected	• 클래스 내부와 동일 패키지, 상속받은 클래스에서만 접근을 허용
public	• 클래스 내부, 동일 패키지, 상속, 외부 접근 가능 • 접근을 제한하지 않음

잠깐! 알고가기

static
정적이라는 의미로 static 변수와 static 메서드 형태가 있다.

static 변수
객체들이 다 같이 공유하는 데이터를 의미한다.

static 메서드
객체들의 데이터와 관계없이 완벽하게 공통적인 로직을 정의할 때 사용한다.

두음쌤 한마디

접근제어 지시자 유형
「프디피」
private / default / protected / public

구분	내용
	• 접근제어 지시자는 변수, 메서드의 접근 범위를 제한하는 용도이며 private, default, protected, public순으로 접근 허용 범위가 커짐 • class는 클래스를 선언하기 위한 예약어 • 클래스 이름은 Test1이고, 클래스 이름은 소스 코드 파일 명과 동일하게 작성이 되어야 함 • 소스 코드 파일 명은 Test1.java

`public static void main(String[] args) {`
 	
`}`
	• main 함수는 프로그램의 시작점
	• static은 모든 객체에서 공통으로 사용함을 의미
	• void는 반환 타입
	• main은 함수 이름이며 프로그램에서 유일하게 사용
	• String[] args는 커맨드라인 매개변수

• 자바 프로그래밍은 객체, 클래스, 메시지를 이용하여 개발하는 방식으로, 각 구성요소에 대한 내용은 다음과 같다.

▼ 자바 프로그래밍의 구성요소

구성요소	설명
객체(Object)	• 현실 세계에서 개체를 데이터 속성과 메서드를 결합한 형태로 표현한 것을 의미하며 개체, 속성, 메서드로 구성 • 절차 지향에서의 모듈은 객체지향의 객체에 대응
클래스(Class)	• 객체지향 프로그래밍에서 객체를 표현하는 추상 데이터 타입으로 객체를 생성하는 틀 • 클래스를 통해 추상화된 자료형을 제공
메시지(Message)	• 객체 간의 통신

구성요소	설명
개체(Entity)	현실 세계에 보이는 본질을 의미
속성(Attribute)	자료 저장소 역할을 하며, 절차 지향 프로그래밍의 변수와 대응
메서드(Method)	호출 단위를 의미하며, 절차 지향 프로그래밍의 함수와 대응

두음쌤 한마디

객체지향 프로그래밍의 구성요소

「객클메」

객체 / 클래스 / 메시지

3 자바 변수

• 자바도 C언어와 유사하게 변수를 선언하고 사용할 수 있다.

▼ 자바 변수 예제 소스 코드

```
public class Test1 {
  public static void main(String[] args) {
    int a;
    a = 0;
    a = 2 + 3;
    a = a + 1;
  }
}
```

• 정수형 변수 a를 선언
• a에 0을 대입하여 0으로 초기화
• 2와 3을 더한 값을 a에 대입함
• 오른쪽 a는 5이며, 1을 더한 6을 왼쪽 a에 대입함

4 자바 조건문

- if문은 조건문이 참일 경우 문장이 실행되고 거짓일 경우에는 실행되지 않는다.
- 다음은 if문 예제 소스 코드로 C언어와 유사하며, C언어와의 차이점으로 출력을 위해 System.out.println() 메서드가 사용된다.

잠깐! 알고가기

메서드(Method)
메서드는 클래스에서 정의된 여러 종류의 변수들을 사용하여 정해진 기능들을 실행할 수 있도록, 코드들을 선언한 것으로 다른 곳에서 인자를 주어 호출할 수도 있고, 정해진 자료형을 반환할 수 있다.

▼ 자바 if문 예제 소스 코드

`public class Test1 {`
`  public static void main(String[] args) {`
`  int a = 1;`
`  int b = 2;`
 	
`  if( a > b ) {`
`  System.out.println("a가 b보다 크다");`
`  } else if ( a < b ) {`
`  System.out.println("a가 b보다 작다");`
`  } else {`
`  System.out.println("a와 b가 같다");`
`  }`
`  }`
`}`
	• 정수형 변수 a를 선언하고 1로 초기화
	• 정수형 변수 b를 선언하고 2로 초기화
	
	• a가 b보다 크면
	• "a가 b보다 크다" 출력
	• a가 b보다 작으면
	• "a가 b보다 작다" 출력
	• 그 외일 경우
	• "a와 b가 같다" 출력

출력 결과 : a가 b보다 작다	

5 자바 반복문

- 자바는 C언어와 유사하게 반복문으로 while문과 for문이 있다.

▼ 자바 반복문

while문	for문

`while (조건식) {`
`  문장;`
`}`
	`for (초깃값;조건식;증감 값) {`
	`  문장;`
	`}`

조건식이 참일 경우 문장을 반복 실행한다.	• 초깃값이 실행되고 조건식이 참일 경우 문장을 실행하고, 증감 값이 증가 또는 감소한다. • 증가 또는 감소된 값이 조건식을 참으로 만족하면 문장이 반복 실행된다.

6 자바 연산자

• 연산자도 C언어와 유사하며 산술 연산자, 관계 연산자, 논리 연산자 등이 있다.

① 산술 연산자

▼ 자바 산술 연산자 예제 코드

```
public class Test1 {
   public static void main(String[]
args) {
    int a = 1;
    int b = 2;

    int r1 = a + b;
    int r2 = a - b;
    int r3 = a * b;
    int r4 = a / b;
  }
}
```

- a를 선언하고 1로 초기화
- b를 선언하고 2로 초기화

- r1를 선언하고 a와 b를 더한 결과를 r1에 대입함
- r2를 선언하고 a와 b를 뺀 결과를 r2에 대입함
- r3를 선언하고 a와 b를 곱한 결과를 r3에 대입함
- r4를 선언하고 a와 b를 나눈 결과를 r4에 대입함

② 관계 연산자

• 관계 연산자는 두 피연산자 사이의 크기를 비교하는 연산자이다.

• 관계 연산자에는 '<', '<=', '>', '>=', '==', '!=' 가 있다.

▼ 자바 관계연산자 예제 코드

```
public class Test1 {
  public static void main(String[] args) {
    int a = 1;
    int b = 2;

    if ( a < b ) {
      System.out.println("a가 b보다 작다.");
    }
  }
}
```

- 정수형 변수 a를 선언하고 1로 초기화
- 정수형 변수 b를 선언하고 2로 초기화

- a가 b보다 작으면
- "a가 b보다 작다."출력

출력 결과 : a가 b보다 작다.

핵인싸 퀴즈

13 ( )의 특징으로 보안성, 일관성, 회복성, 무결성, 효율성 등이 있다.

14 ( ) 데이터베이스는 ( ) 구조를 사용하여 데이터를 표현하고 저장하는 비 관계형 데이터베이스이다.

15 ( )은/는 JSON과 유사한 형식의 문서로 데이터를 저장 및 쿼리 하도록 설계된 비 관계형 데이터베이스이다.

정답 13. DBMS 14. 그래프 15. 문서 또는 도큐먼트

핵인싸 퀴즈

16 ( )은/는 업무 분석 결과로 도출된 개체(Entity)와 개체 간의 관계(Relation)를 도식화한 표현이다.

17 ( )은/는 개체가 가지고 있는 요소 또는 성질이다.

18 ( )은/는 관계형 데이터베이스 관리 시스템(RDBMS)의 데이터를 관리하기 위해 설계된 특수 목적의 프로그래밍 언어이다.

정답 16. ERD 또는 개체-관계 다이어그램 17. 속성(Attribute) 18. SQL

③ 논리 연산자

- 논리 연산자는 두 조건식의 참과 거짓에 따라 전체 조건식의 참과 거짓을 결정하는 연산자이다.
- 논리 연산자에는 '&&', '||'가 있다.

▼ 자바 논리 연산자 예제 코드

```
public class Test1 {
  public static void main(String[] args) {
    int a = 1;
    int b = 2;

    if( a == 1 && b == 2 ) {
      printf("a는 10이고 b는 20이다.");
    }
  }
}
```

- 정수형 변수 a에 1 대입
- 정수형 변수 b에 2 대입
- a가 1과 같고 b가 2와 같으면
- "a는 10이고 b는 20이다."를 출력

출력 결과 : a는 10이고 b는 20이다.

7 자바 배열

- 동일한 타입의 변수를 여러 개 만드는 경우 배열을 사용한다.
- 자바는 배열의 길이는 length라는 속성을 이용할 수 있다.

▼ 자바 배열 예제 코드

```
public class Test1 {
  public static void main(String[] args) {
    int[] arr = {1,2,3,4};

    for( i=0; i<arr.length;i++) {
      System.out.print(arr[i]);
    }
  }
}
```

- 배열 arr 선언 및 초기화
- for문 인덱스 i가 0부터 length까지 1씩 증가
- 배열의 각 요소 출력

출력 결과 : 1234

(2) 자바를 이용한 알고리즘 처리 예제

1 짝수 홀수 판별 알고리즘

- 다음은 1부터 10까지 홀수, 짝수의 합을 계산하는 알고리즘이다.

▼ 짝수 홀수 판별 알고리즘

```
public class Test1 {
  public static void main(String[]
args) {
    int i;
    int odd_sum=0;
    int even_sum=0;
    int[] data = {1, 2, 3, 4, 5,
6, 7, 8, 9, 10};

    for(i=0;i<10;i++) {
      if( i%2 == 0 ) {
        even_sum += data[i];
      } else {
        odd_sum += data[i];
      }
    }
    System.out.print(odd_sum);
    System.out.print(', ');
    System.out.print(even_sum);
  }
}
```

- for문에서 사용할 변수 i 선언
- 홀수 합을 저장할 변수 odd_sum 선언 및 0으로 초기화
- 짝수 합을 저장할 변수 even_sum 선언 및 0으로 초기화
- data 배열 선언 및 초기화

- 0부터 9까지 1씩 순차적 증가하며 반복
- i와 2를 나머지 연산하여 나머지가 0이면
- 짝수 합 변수인 even_sum에 i값을 누적
- i와 2를 나머지 연산하여 나머지가 이면
- 홀수 합 변수인 even_sum에 i값을 누적

- 홀수의 합을 화면에 출력
- ','를 출력
- 짝수의 합을 화면에 출력

출력 결과 : 30, 25

2 스위칭 변수를 이용한 합계 알고리즘

- 아래는 1부터 10까지 '+' 연산자와 '−' 연산자를 번갈아 가면서 "1−2+3−4+5−6+7−8+9−10"을 계산하는 알고리즘이다.

▼ 스위칭 변수를 이용한 합계 알고리즘

```
public class Test1 {
  public static void main(String[]
args) {
    int i;
    int sw=0;
    int sum=0;

    for(i=1;i<=10;i++)
    {
      if(sw == 0) {
        sum += i;
        sw = 1;
      } else {
        sum -= i;
        sw = 0;
      }
    }

    System.out.print(sum);
  }
}
```

- for문에서 사용할 변수 i 선언
- 스위칭 변수 sw 선언 및 0으로 초기화
- 누적 합을 저장하는 변수 sum 선언 및 0으로 초기화

- 1부터 10까지 1씩 순차적 증가하며 반복

- 스위칭 변수 sw가 0과 같으면
- sum 변수에 i값을 더한 결과를 sum에 대입
- 스위칭 변수 sw에 1을 대입
- 스위칭 변수 sw가 0과 같지 않으면
- sum 변수에 i값을 뺀 결과를 sum에 대입
- 스위칭 변수 sw에 0을 대입

- 화면에 결과를 출력

출력 결과 : −5

핵인싸 퀴즈

19 ( )은/는 데이터베이스 오브젝트 생성 명령어이다.

20 ( )은/는 데이터베이스 오브젝트 수정 명령어이다.

21 ( )은/는 데이터베이스 오브젝트 삭제 명령어이다.

정답 19. CREATE 20. ALTER 21. DROP

CHAPTER

02 프로그래밍 언어 활용

1 프로그래밍 언어의 기본문법 이해 및 언어 특성의 활용☆☆

프로그래밍을 위한 대표적인 용어로 변수, 바인딩, 데이터 타입 등이 있다.

▼ 프로그래밍 기본 용어

용어	설명
변수	• 어떤 값을 주기억장치에 기억하기 위해서 사용하는 공간
식별자	• 프로그램의 구성요소를 구별하기 위한 기준 예 변수명, 함수명
바인딩	• 변수와 변수에 관련된 속성을 연결하는 과정 • 정적 바인딩과 동적 바인딩으로 구분 정적 바인딩 : 프로그램 실행 시간 전에 속성을 연결하는 방식 동적 바인딩 : 프로그램 실행 시간에 속성을 연결하는 방식
선언	• 변수에 이름, 데이터 타입 등의 속성을 부여하는 작업 • 명시적 선언과 묵시적 선언으로 구분 명시적 선언 : 선언문을 이용하여 변수 이름을 나열하고 속성을 부여하는 방식 묵시적 선언 : 별도의 선언문 없이 디폴트 규칙에 의해 속성이 부여되는 방식
영역	• 이름이 사용되는 범위를 의미 • 정적 영역과 동적 영역으로 구분 정적 영역 : 변수를 찾을 때 구조에 기반하는 방식 동적 영역 : 변수를 찾을 때 구조보다는 순서에 기반하는 방식
할당	• 변수에 메모리 공간을 바인딩하는 작업
데이터 타입	• 변수가 가질 수 있는 속성 및 속성값의 길이
연산자	• 데이터 처리를 위해 연산을 표현하는 기호 예 +, −, *, /
명령문	• 프로그램을 구성하는 문장으로, 지시사항을 처리하는 단위

학습 Point

프로그래밍 언어 활용에서는 기본문법과 개념을 중심으로 출제 가능성이 높습니다. 눈여겨 봐주세요!

(1) 변수

1 데이터 타입(Data Type)의 개념

- 프로그래밍 언어에서 실수치, 정수 자료형과 같은 여러 종류의 데이터를 식별하는 형태이다.
- 메모리 공간을 효율적으로 사용하고 2진수 데이터를 다양한 형태로 사용하기 위해 존재한다.

2 데이터 타입의 유형

- 프로그래밍 언어에 따라 데이터 타입의 유형은 다를 수 있다.
- 일반적으로 불린 타입, 문자 타입, 문자열 타입, 정수 타입, 부동 소수점 타입, 배열 타입의 유형을 가진다.

▼ 데이터 타입의 유형

유형	설명

불린 타입
(Boolean type)
	• 조건이 참인지 거짓인지 판단하고자 할 때 사용
	• C언어에서는 미지원
	• C++, 자바에서는 참일 경우 true로 거짓일 경우 false로 표현
	• 파이썬에서는 참일 경우 True, 거짓일 경우 False로 표현
	예 bool a = true; // C++
	boolean a = true; // 자바

문자 타입 (Char type)	문자 하나를 저장하고자 할 때 사용 예 char a = 'A';

문자열 타입
(String type)
	나열된 여러 개의 문자를 저장하고자 할 때 사용
	C언어에서는 지원하지 않는 타입
	예 string a = "Hello"; // C++
	String a = "Hello"; // 자바

정수 타입 (Int type)	정숫값을 저장하고자 할 때 사용 예 int a = 5;
부동 소수점 타입 (Float type)	소수점을 포함하는 실숫값을 저장하고자 할 때 사용 예 float a = 4.5; double b = 4.5;

배열 타입
(Array type)
	여러 데이터를 하나로 묶어서 저장하고자 할 때 사용
	예 a라는 이름의 정수형 변수를 5개 선언
	int a[5] = {1, 2, 3, 4, 5}; // C, C++
	int[] a = {1, 2, 3, 4, 5}; // 자바

학습 Point

파이썬에서는 별도로 타입을 지정하지 않아도 됩니다. 반면 C, C++, 자바에서는 변수 사용을 위해 별도로 타입을 지정해야 합니다.

두음쌤 한마디

데이터 타입의 유형

「불문열 정부배」

불린 / 문자 / 문자열 / 정수 / 부동 소수점 / 배열 타입

→ 불문율(열)로 정부에서 배수입 금지

잠깐! 알고가기

문자와 문자열 따옴표 작성 방법

- 문자는 작은 따옴표(' ')를 붙인다.
- 문자열은 큰 따옴표(" ")를 붙인다.

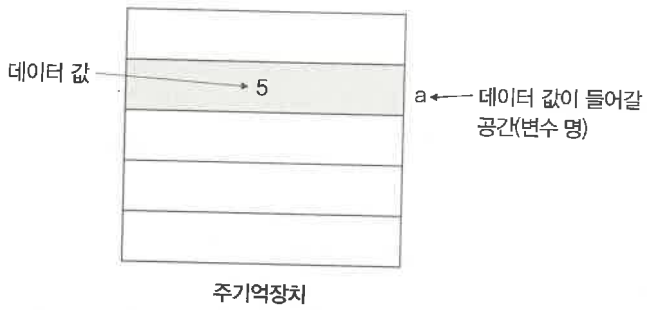
예 문자 : 'a'
예 문자열 : "a"

float와 double 차이

float형과 double형 모두 실숫값을 표현하지만, float는 4바이트로 보통 소수점 6자리까지 표현 가능하지만, double형은 8바이트 소수점 15자리까지 표현 가능하다.

3 변수(Variable)의 개념

• 변수는 저장하고자 하는 어떠한 값이 있을 때, 그 값을 주기억장치에 기억하기 위한 공간을 의미한다.



변수명 a라는 주기억장치의 공간에 5라는 데이터 값이 들어 있음

▲ 변수의 개념도

4 변수 설정 규칙

• 변수는 프로그램 소스 코드의 공유, 유지 관리, 표준화 등을 위해 일정한 규칙에 따라 작성된다.

▼ 변수 설정 규칙

구분	규칙
사용 가능 문자	• 영문 대문자/소문자, 숫자, 밑줄('_')의 사용이 가능 예 a, A, a1, _hello
변수 사용 규칙	• 첫 자리에는 숫자를 사용할 수 없음 예 _1, a1, a100
	• 변수 이름의 중간에는 공백을 사용할 수 없음 예 my_student
변수 의미 부여	• 데이터값의 의미나 역할을 표현 예 age, student
	• 이미 사용되고 있는 예약어의 경우에는 변수로 사용할 수 없음 • 예약어와 동일 또는 유사하게 변수로 사용하게 되면 혼돈을 줄 수 있으므로 가급적 예약어와 동일, 유사하게 사용하지 말아야 함 예 for, if, while

학습 Point

C, C++, 자바, 파이썬 모두 같은 변수 설정 규칙을 가지고 있습니다. 한번에 알아두세요!

잠깐! 알고가기

예약어(Reserved Word)
컴퓨터 프로그래밍 언어에서 이미 문법적인 용도로 사용되고 있기 때문에 식별자로 사용할 수 없는 단어이다. 데이터 타입(int, float, …), 조건문(if, switch, case), 반복문(while, for, do), 루프 제어 명령문(break, continue), 함수 반환 값(return) 등이 이에 해당된다.

5 변수 선언

- 자료형과 변수명을 작성하여 변수를 생성하는 과정이다.
- C, C++, 자바에서는 변수 선언을 하고, 파이썬에서는 변수 선언을 하지 않는다.

① 일반 변수 선언

▼ 일반 변수 선언

선언	설명
데이터타입 변수명 = 초깃값;	불린, 문자, 정수, 실수 등을 선언할 때 사용 예 정수형 변수 a 선언 int a; 예 초깃값 0인 실수형 변수 b 선언 float b = 0.0;

② 배열 변수 선언

- 불린, 문자, 정수, 실수 등을 배열로 선언할 때 사용한다.
- 배열 인덱스는 0부터 시작한다.

▼ 배열 변수 선언

언어	선언 및 초기화
C, C++	데이터 타입 배열 변수명[배열 사이즈] = {초깃값_들}; 예 정수형 배열 a를 3개 선언 int a[3]; 예 초깃값 0인 실수형 변수 b를 3개 선언 float b[3] = {0.0, 0.0, 0.0};
자바	데이터 타입[배열사이즈] 배열 변수명 = {초깃값_들}; 예 정수형 배열 arr를 3개 선언 int[3] arr; 예 정수형 배열 arr를 3개 선언 및 초기화 int[] arr = {1, 2, 3};

두음쌤 한마디

연산자의 유형

「산시관논 비대증」

산술 연산자 / 시프트 연산자 / 관계 연산자 / 논리 연산자 / 비트 연산자 / 대입 연산자 및 복합대입 연산자 / 증감 연산자

학습 Point

연산자의 유형이 출제될 가능성이 높습니다. 두음쌤의 도움을 받아 학습하시기 바랍니다!

(2) 연산자와 명령문 활용

1 연산자(Operator)의 개념

- 연산자는 프로그램 실행을 위해 연산을 표현하는 기호이다.
- 연산자에는 산술 연산자, 시프트 연산자, 관계 연산자, 논리 연산자 등이 있으며, 각 연산자는 다음과 같은 종류를 가진다.

① 산술 연산자(Arithmetic Operator)

산술 연산자는 +, −와 같이 가장 일반적으로 사용되는 연산자이다.

▼ 산술 연산자의 종류

연산자	내용
+	양쪽의 값을 더하는 연산자
−	왼쪽 값에서 오른쪽 값을 빼는 연산자
*	두 개의 값을 곱하는 연산자
/	왼쪽 값을 오른쪽 값으로 나누는 연산자
%	왼쪽 값을 오른쪽 값으로 나눈 나머지를 계산하는 연산자

② 시프트 연산자(Shift Operator)

시프트 연산자는 비트를 이동시키는 연산자이다.

▼ 시프트 연산자의 종류

연산자	내용
《	왼쪽 값을 오른쪽 값만큼 비트를 왼쪽으로 이동시키는 연산자 예 2《1 왼쪽 2를 오른쪽 1만큼 비트를 왼쪽으로 이동시키므로 4가 됨
》	왼쪽 값에 오른쪽 값만큼의 비트를 채우면서 오른쪽으로 이동시키는 연산자 예 2》1 왼쪽 2를 오른쪽 1만큼 비트를 오른쪽으로 이동시키므로 1이 됨

③ 관계 연산자(Relation Operator)

관계 연산자는 두 피연산자 사이의 크기를 비교하는 연산자이다.

▼ 관계 연산자의 종류

연산자	내용
〉	왼쪽에 있는 값이 오른쪽에 있는 값보다 크면 참을 반환하고, 그렇지 않으면 거짓을 반환하는 연산자
〈	왼쪽에 있는 값이 오른쪽에 있는 값보다 작으면 참을 반환하고, 그렇지 않으면 거짓을 반환하는 연산자
〉=	왼쪽에 있는 값이 오른쪽에 있는 값보다 크거나 같으면 참을 반환하고, 그렇지 않으면 거짓을 반환하는 연산자
〈=	왼쪽에 있는 값이 오른쪽에 있는 값보다 작거나 같으면 참을 반환하고, 그렇지 않으면 거짓을 반환하는 연산자
==	왼쪽에 있는 값이 오른쪽에 있는 값과 같으면 참을 반환하고, 그렇지 않으면 거짓을 반환하는 연산자
!=	왼쪽에 있는 값이 오른쪽에 있는 값과 다르면 참을 반환하고, 그렇지 않으면 거짓을 반환하는 연산자

핵인싸 퀴즈

22 ( )은/는 데이터베이스 데이터 조회 명령어이다.

23 ( )은/는 데이터베이스 데이터 삽입 명령어이다.

24 ( )은/는 데이터베이스 데이터 갱신 명령어이다.

정답 22. SELECT 23. INSERT 24. UPDATE

잠깐! 알고가기

C언어 연산자 우선순위

식 〉 단항 〉 곱하기 〉 더하기 〉 시프트 〉 관계 〉 비트 연산 〉 논리 연산 〉 복합 연산

④ 비트 연산자(Bit Operator)

비트 연산자는 0과 1의 각 자리에 대한 연산을 수행하며, 0 또는 1의 결괏값을 갖는 연산자이다.

▼ 비트 연산자의 종류

연산자	내용
&	두 값을 비트로 연산하여 모두 참이면 참을 반환하고, 그렇지 않으면 거짓을 반환하는 연산자
|	두 값을 비트로 연산하여 하나가 참이면 참을 반환하고, 그렇지 않으면 거짓을 반환하는 연산자
^	두 값을 비트로 연산하여 서로 다르면 참을 반환하고, 그렇지 않으면 거짓을 반환하는 연산자

⑤ 논리 연산자(Logic Operator)

논리 연산자는 두 피연산자 사이의 논리적인 관계를 정의하는 연산자이다.

▼ 논리 연산자의 종류

연산자	내용
&&	두 개의 논릿값이 모두 참이면 참을 반환하고, 그렇지 않으면 거짓을 반환하는 연산자
||	두 개의 논릿값 중 하나가 참이면 참을 반환하고, 그렇지 않으면 거짓을 반환하는 연산자

⑥ 대입 연산자(Assignment Operator) 및 복합대입 연산자

- 변수에 값을 대입할 때 사용하는 연산자이며, 오른쪽에 있는 값을 이용해 왼쪽의 변수에 값을 대입한다.
- 복합대입 연산자는 연산과 대입이 합쳐진 연산이다.

▼ 대입 연산자의 종류

연산자	내용
=	• 왼쪽의 변수에 오른쪽의 값을 대입하는 연산자
+=	• 왼쪽의 변수에 오른쪽의 값을 더한 후, 그 결괏값을 왼쪽의 변수에 대입하는 연산자 • a += b는 a = a + b와 동일
-=	• 왼쪽의 변수에 오른쪽의 값을 뺀 후, 그 결괏값을 왼쪽의 변수에 대입하는 연산자 • a -= b는 a = a - b와 동일
*=	• 왼쪽의 변수에 오른쪽의 값을 곱한 후, 그 결괏값을 왼쪽의 변수에 대입하는 연산자 • a *= b는 a = a * b와 동일
/=	• 왼쪽의 변수를 오른쪽의 값으로 나눈 후, 그 결괏값을 왼쪽의 변수에 대입하는 연산자 • a /= b는 a = a / b와 동일
%=	• 왼쪽의 변수를 오른쪽의 값으로 나눈 후, 그 나머지를 왼쪽의 변수에 대입하는 연산자 • a %= b는 a = a % b와 동일

⑦ 증감 연산자(Increment & Decrement Operator)

증감 연산자는 피연산자를 1씩 증가 혹은 1씩 감소시킬 때 사용하는 연산자이다.

▼ 증감 연산자의 종류

연산자	내용
++x	변수의 값을 1 증가시킨 후에 해당 변수를 사용하는 연산자
x++	변수를 사용한 후에 변수의 값을 1 증가시키는 연산자
--x	변수의 값을 1 감소시킨 후에 해당 변수를 사용하는 연산자
x--	변수를 사용한 후에 변수의 값을 1 감소시키는 연산자

예 x의 값이 3이라고 가정했을 때 자바에서 출력값

```
System.out.print(x++); // 3이 출력된 이후에 x의 값이 1 증가
System.out.print(x);   // 4가 출력됨
```

예 x의 값이 3이라고 가정했을 때 자바에서 출력값

```
System.out.print(++x); // x의 값을 1 증가시킨 후에 x를 출력(4가 출력됨)
System.out.print(x);   // 4가 출력됨
```

잠깐! 알고가기

파이썬 증감 연산자

파이썬에는 증감연산자가 없으므로 변수에 1을 더한다.

예 i = i + 1;

2 명령문

- 명령문은 프로그램을 구성하는 문장으로, 지시 사항을 처리하는 단위이다.
- 기본적인 문법들의 종류가 매우 많은 것은 아니며, 각 언어마다 유사한 문법 체계를 사용한다.

① 조건문

- 조건문은 조건의 참, 거짓 여부에 따라 실행 경로를 달리하는 if 문과 여러 경로 중에 하나를 선택하는 switch 문으로 구분한다.
- if 문은 산술 또는 논리적으로 비교가 가능하나 switch는 조건이 동일한지의 여부만 확인한다.

㉮ C, C++, 자바 언어에서의 if 문

- if 문의 경우 대다수의 프로그래밍 언어에서 기본 명령문으로 사용하게 되었다.

학습 Point

if문은 프로그래밍의 핵심 문법 중 하나입니다. 예시를 최대한 쉽게 작성했으니 차근차근 읽고 이해합시다!

▼ if문 종류

<table>
<tr><th>종류</th><th colspan="2">설명</th></tr>
<tr><td>if문</td><td colspan="2">어떤 문장을 수행할지 여부를 결정할 때 사용
<pre>if (조건문) {
  명령문;
}</pre></td></tr>
<tr><td>else if문</td><td colspan="2">조건이 여러 개일 경우 사용</td></tr>
<tr><td></td><td><pre>if (조건문1) {
  명령문1;
} else if (조건문2) {
  명령문2;
} else if (조건문3) {
  명령문3;
}</pre></td><td>• if문이 반드시 먼저 나오고 else if문이 나옴
• else if문은 1번 이상 사용 가능</td></tr>
<tr><td>else문</td><td colspan="2">조건식이 모두 거짓일 때 사용</td></tr>
<tr><td></td><td><pre>if (조건문1) {
  명령문1;
} else {
  명령문2;
}</pre></td><td>if문의 조건문 1이 거짓일 경우 else문의 명령문 2가 실행됨</td></tr>
<tr><td></td><td><pre>if (조건문1) {
  명령문1;
} else if (조건문2) {
  명령문2;
} else if (조건문3) {
  명령문3;
} else {
  명령문4;
}</pre></td><td>모든 조건문이 거짓일 경우 else문에 있는 명령문이 실행됨</td></tr>
</table>

다음은 점수에 따라서 수우미양가를 알려주는 예제이다. (score 변수에는 점수가 입력되어 있다고 가정)

▼ C, C++, 자바 언어에서의 if 문 예제

C언어	C++언어	자바 언어

if(score >= 90){
  printf("수");
} else if(score >= 80){
  printf("우");
} else if(score >= 70){
  printf("미");
} else if(score >= 60){
  printf("양");
} else {
  printf("가");
}
	if(score >= 90){
	  std::cout << "수";
	} else if(score >= 80){
	  std::cout << "우";
	} else if(score >= 70){
	  std::cout << "미";
	} else if(score >= 60){
	  std::cout << "양";
	} else {
	  std::cout << "가";
	}
		if(score >= 90){
		  System.out.print("수");
		} else if(score >= 80){
		  System.out.print("우");
		} else if(score >= 70){
		  System.out.print("미");
		} else if(score >= 60){
		  System.out.print("양");
		} else {
		  System.out.print("가");
		}

④ 파이썬 if문

if문 작성 시 조건문 다음에 콜론(:)을 반드시 작성해야 하며, 들여쓰기를 해야한다.

▼ 파이썬에서 사용하는 if문

```
if 조건문1:
  명령문1;
elif 조건문2:
  명령문2;
else:
  명령문3;
```

다음은 점수에 따라서 수우미양가를 알려주는 예제이다. (score 변수에는 점수가 입력되어 있다고 가정)

▼ 파이썬에서의 if문 예제

```
if score >= 90:
  print("수")
elif score >= 80:
  print("우")
elif score >= 70:
  print("미")
elif score >= 60:
  print("양")
else:
  print("가")
```

잠깐! 알고가기

파이썬 들여쓰기

파이썬으로 프로그램을 작성하는 경우 코드 블록('{', '}')을 명시하지 않고 들여쓰기를 한다. 들여쓰기를 하지 않으면 실행 시 오류가 발생한다.

들여쓰기 전	들여쓰기 후
if a > 0: b = c;	if a > 0: b = c;
// 오류 발생	//오류 발생 안 함

학습 Point

switch문 역시 중요 문법 중 하나입니다. 언어별로 표현만 다르지 내용이 일맥상통합니다. 어렵지 않아요. 힘내봅시다!

㉰ switch문

- switch문에서는 조건에 해당하는 case로 이동하고 break가 있으면 switch문을 빠져나온다.
- break가 존재하지 않을 경우 break를 만날 때까지 switch문에 있는 다음 case 문장을 실행한다.

▼ C, C++, 자바에서 사용하는 switch문

```
switch (조건문){
case 조건값:
  명령문;
  break;
default:
}
```

▼ C, C++, 자바 언어에서의 switch문 예제(1)

C언어	C++ 언어	자바 언어

switch(score/10){
case 10:
printf("수");
break;
case 9:
printf("수");
break;
case 8:
printf("우");
break;
case 7:
printf("미");
break;
case 6:
printf("양");
break;
default:
printf("가");
}
	switch(score/10){
	case 10:
	std::cout << "수";
	break;
	case 9:
	std::cout << "수";
	break;
	case 8:
	std::cout << "우";
	break;
	case 7:
	std::cout << "미";
	break;
	case 6:
	std::cout << "양";
	break;
	default:
	std::cout << "가";
	}
		switch(score/10){
		case 10:
		System.out.print("수");
		break;
		case 9:
		System.out.print("수");
		break;
		case 8:
		System.out.print("우");
		break;
		case 7:
		System.out.print("미");
		break;
		case 6:
		System.out.print("양");
		break;
		default:
		System.out.print("가");
		}

break를 만나기 전까지 계속 실행하기 때문에 아래와 같은 코드를 작성해도 동일하게 작동한다.

▼ C언어에서의 switch문 예제(2)

```
switch(score/10){
case 10: // case 10에 해당되더라도 break가 없으므로 case 9의 문장 실행
case 9:
  printf("수"); break;
case 8:
  printf("우"); break;
case 7:
  printf("미"); break;
case 6:
  printf("양"); break;
default:
  printf("가");
}
```

② 반복문

반복문은 특정 부분을 조건이 만족할 때까지 실행하도록 하는 명령문이다.

㉮ C, C++, 자바 while문과 for문

조건문이 참인 동안 반복해서 명령을 수행한다.

▼ C, C++, 자바 while문

```
while (조건문) {
  명령문;
}
```

▼ C, C++, 자바 for문

```
for (초깃값; 조건문; 증감 값) {
  명령문;
}
```

㉯ 파이썬 while문과 for문

while문과 for문 작성 시 문장 끝에 콜론(:)을 반드시 작성해야 하며, 그 다음 줄부터는 반드시 들여쓰기를 해야 한다.

▼ 파이썬 while문

```
while 조건문:
    명령문;
```

핵인싸 퀴즈

25 ( )은/는 원하는 정보를 원하는 수신자 또는 기기에 정확하게 전송하기 위한 기반 인프라이다.

26 ( )은/는 컴퓨터나 원거리 통신 장비 사이에서 메시지를 주고받는 양식과 규칙의 체계이다.

27 ( )은/는 한 건물 또는 작은 지역을 커버하는 네트워크이다.

정답 25. 네트워크 26. 프로토콜 27. LAN 또는 근거리 네트워크

▼ 파이썬 for문

`for 변수 in range (시작값, 끝값+1) :` `  명령문;`	• range에 (시작값)과 (끝값+1)로 정의할 경우 (끝값) – (시작값)만큼 반복 • 변수는 (시작값)부터 (끝값)까지 1씩 증가
`for 변수 in range (반복 횟수) :` `  명령문;`	• range에 (반복 횟수)를 정의한 경우 0부터 (반복 횟수 – 1)까지 변수가 1씩 증가

다음은 1부터 100까지의 숫자를 모두 더하는 예제이다. (합한 값은 sum, 인덱스는 i라는 변수를 사용하고, sum의 초깃값은 0이라고 가정)

▼ 파이썬 for문 예제

`for i in range (0, 100) :` `  sum += i;`	i라는 변수는 0부터 99까지 1씩 증가하여 100번 반복
`for i in range (100) :` `  sum += i;`	i라는 변수는 0부터 99까지 1씩 증가하여 100번 반복

㉯ 반복문 제어 명령어

▼ 반복문 제어 명령어

명령어	내용
break	반복문을 강제로 멈추기 위해 사용하는 명령어
continue	반복문에서 다음 반복으로 넘어갈 수 있도록 하는 명령어

다음 예시는 i가 7일 때 반복문을 강제로 멈추는 예시이다.

▼ C++에서 break문 사용 예시

`int i=0;`
`while ( i < 10 ) {`
`  if ( i == 7 ) break;`
`  i++;`
`  std::cout << i;`
`}`
	• 초기 i값은 0
	• 반복문은 i가 10 미만일 경우 수행
	• i가 7일 경우 break로 반복문을 강제로 멈춤
	• i가 1 증가
	• i 값을 출력

출력: 123456	

다음 예시는 i가 7일 때 다음 반복문으로 넘어갈 수 있도록 하는 예시이다.

▼ C++에서 continue문 사용 예시

`i=0;`
`while ( i < 10 ) {`
`  if ( i == 7 )`
`  continue;`
`  i++;`
`  std::cout << i;`
`}`
	• 초기 i 값은 0
	• 반복문은 i가 100 미만일 경우 수행
	• i가 7일 경우 수행
	• continue를 만나면 다음 반복 수행
	• i가 1 증가
	• i 값을 출력

출력: 12345689	

학습 Point

C, C++, 자바, 파이썬 모두 break, continue 명령을 사용하고 기능도 동일합니다. 한번에 알아두세요!

(3) 사용자 정의 자료형 활용

1 사용자 정의 자료형 개념

- C/C++, 자바와 같은 프로그래밍 언어에서는 사용자가 직접 자료형을 만드는 것이 가능하다. 이렇게 직접 만드는 자료형을 사용자 정의 자료형이라고 한다.
- C++의 경우 열거체, 구조체, 공용체로 구분하여 사용자 정의 자료형의 작성이 가능하다.

① 열거체(Enumeration)

- 열거체는 서로 연관된 정수형 상수들의 집합이다.
- 정수형 상수에 이름을 붙여서 코드를 이해하기 쉽게 해준다.
- 멤버에 초깃값을 설정하지 않으면 0부터 차례대로 할당된다.

▼ C, C++에서 사용하는 enum

```
enum 열거체명{
  멤버1,
  멤버2,
  ...
};
```

- 멤버1은 0이 되며 멤버 2부터는 1씩 증가된다.
- 다음은 열거체 예제 코드이다.

▼ C, C++에서 enum 예제 코드

```
#include <stdio.h>

enum Week {
    Sunday,
    Monday,
    Tuesday = 5,
    Wednesday
};

void main()
{
    enum Week week;
    week = Sunday;
}
```

- Sunday는 초깃값이 설정되지 않아 0으로 설정
- Monday는 1로 설정
- Tuesday는 5를 대입하여 5로 설정
- Wednesday는 6으로 설정

- enum Week라는 타입의 week 변수 선언
- week 변수에 Sunday의 값인 0을 대입

핵인싸 퀴즈

28 ( )은/는 통신 사업자가 사전에 계약을 체결한 송신자와 수신자끼리만 데이터를 교환하는 방식이다.

29 ( )은/는 공중망을 활용하여 다수 사용자가 선로를 공유하는 방식이다.

30 ( )은/는 다른 응용 프로그램, 사용자, 데이터 흐름 등에 우선순위를 정하여, 데이터 전송에 특정 수준의 성능을 보장하기 위한 능력을 말한다.

정답 28. 서킷 교환방식 29. 패킷 교환 방식 30. QoS(Quality of Service)

학습 Point

자바, 파이썬에서는 구조체를 지원하지 않습니다. 대신 클래스를 이용해 기능을 구현할 수 있습니다. 참고하세요!

② 구조체

구조체는 사용자가 기본 타입을 가지고 새롭게 정의할 수 있는 사용자 정의 자료형이다.

▼ C, C++에서 사용하는 struct

```
struct 구조체명{
  자료형 변수명1;
  자료형 변수명2;
};
```

▼ C, C++에서 struct 사용 예시

```
struct Student {
  char gender;
  int age;
  int num;
};

void main()
{
  struct Student s;
  s.gender = 'M';
  s.age = 24;
  s.num = 2020;
}
```

- Student라는 이름의 구조체 타입을 정의
- 성별을 나타내는 char형 변수
- 나이를 나타내는 int형 변수
- 학번을 나타내는 int형 변수
- struct Student라는 타입의 s라는 이름의 변수 선언
- 구조체 변수 s에 포함된 gender 변수에 값 대입
- 구조체 변수 s에 포함된 age 변수에 값 대입
- 구조체 변수 s에 포함된 school_name 변수에 값 대입

③ 공용체 개념

- 공용체는 모든 멤버 변수가 하나의 메모리 공간을 공유하는 사용자 정의 자료형이다.
- 문법은 구조체와 거의 비슷하지만, 메모리 구조 측면에서 구조체와 공용체가 다르다.

▼ C, C++ 공용체

```
union 공용체명{
  자료형 변수명1;
  자료형 변수명2;
};
```

▼ C, C++ 공용체 사용 예시

`union A {`
`  char c;`
`  int i;`
`};`
	• A라는 이름의 공용체 타입을 정의
	• c라는 이름의 char형 변수(1바이트라고 가정)
	• i라는 이름의 int형 변수(4바이트라고 가정)

`void main()`
`{`
`  union A a;`
`  a.c = 'M';`
`  a.i = 24;`
`}`
	• union A라는 타입의 a라는 이름의 변수 선언
	• 공용체 변수 a에 포함된 c 변수에 값 대입
	• 공용체 변수 a에 포함된 i 변수에 값 대입

잠깐! 알고가기

구조체 메모리 할당

- 구조체에서는 4바이트 단위로 메모리 공간을 사용한다.
- char 타입은 1바이트이지만 구조체에서 char로 선언을 하면 4바이트만큼 할당되게 된다.

④ 추상화와 상속

사용자 정의 자료형은 추상화와 상속의 개념을 이용한다.

▼ 추상화와 상속

구분	설명
추상화	• 복잡한 문제의 본질을 이해하기 위해 세부 사항은 배제하고 중요한 부분을 중심으로 간략화하는 기법 • 기능 추상화, 자료 추상화, 제어 추상화로 구분

종류	설명
기능 추상화	입력 자료를 출력 자료로 변환하는 과정을 추상화하는 방법
자료 추상화	자료와 자료에 적용할 수 있는 연산을 함께 정의하는 방법
제어 추상화	외부 이벤트에 대한 반응을 추상화하는 방법

상속	• 상위 수준 그룹의 모든 특성을 하위 수준 그룹이 이어받아 재사용 또는 확장하는 특성을 의미 • 단일 상속과 다중 상속이 있으며, 상위 수준의 그룹이 하나만 존재할 때 이를 단일 속성이라고 부름
구체화	• 하위 수준 그룹이 상위 수준 그룹의 추상적인 부분을 구체화시키는 것을 의미

(4) 프로그래밍 언어의 언어별 특성 활용

1 개발 편의성에 따른 분류

▼ 개발 편의성에 따른 분류

종류	설명
저급 언어	기계가 이해할 수 있도록 만들어진 언어 예 기계어, 어셈블리어
고급 언어	개발자가 소스 코드를 작성할 때 쉽게 이해할 수 있도록 작성된 언어 예 C, C++, 자바, 파이썬

2 실행하는 방식에 따른 분류

▼ 실행하는 방식에 따른 언어 분류

종류	설명
명령형 언어	컴퓨터에 저장된 명령어들이 순차적으로 실행되는 프로그래밍 방식으로 절차형 언어라고도 불림 예 C언어, 베이직
객체지향 언어	객체 간의 메시지 통신을 이용하여 프로그래밍하는 방식 예 C++, Java
함수형 언어	수학적 수식과 같은 함수들로 프로그램을 구성하여 호출하는 방식 예 LISP, 하스켈

3 구현 기법에 따른 분류

▼ 구현 기법에 따른 분류

종류	설명
컴파일 방식의 언어	• 고급 언어를 기계어로 번역하는 방식의 언어 • 컴파일러에 의해 실행에 필요한 정보가 컴파일 시간에 계산되어 실행 속도가 높음 예 C, C++
인터프리터 방식의 언어	• 고급 언어 명령문을 하나씩 번역하고 실행하는 방식의 언어 • 프로그램 실행과 동시에 동작 예 파이썬, Javascript
혼합형 방식의 언어	• 고급 언어를 컴파일하여 중간 언어로 변환한 후, 인터프리터에 의해 번역을 실행하는 방식의 언어 예 자바

인터프리터 방식의 언어에는 대표적으로 파이썬과 자바스크립트가 있다.

▼ 스크립트 언어 종류

종류	설명
파이썬	• 다양한 플랫폼에서 쓸 수 있고, 라이브러리(모듈)가 풍부 • 유니코드 문자열을 지원하여 다양한 언어의 문자 처리 • 들여쓰기를 사용하여 블록을 구분하는 문법 채용 • 다른 언어로 쓰인 모듈들을 연결하는 언어
자바스크립트	• 객체 기반의 스크립트 프로그래밍 언어 • 웹 브라우저 내에서 주로 사용하며, 다른 응용 프로그램의 내장 객체에도 접근할 수 있는 기능이 존재 • 자바스크립트는 브라우저마다 지원되는 버전이 상이

핵인싸 퀴즈

31 ( )은/는 3계층 프로토콜로 ( ) 주소는 전 세계 컴퓨터에 부여되는 유일한 식별자이다.

32 ( )은/는 비 연결형 서비스를 지원하는 데이터 그램 단위 기반의 전송 계층 프로토콜이다.

33 ( )은/는 광대역 전송에 쓰이는 비동기 전송방식이며, 동기화를 맞추지 않아 보낼 데이터가 없는 사용자의 슬롯은 다른 사람이 사용할 수 있도록 하여 네트워크상의 효율성을 높인 기법이다.

정답 31. IP(Internet Protocol)
32. UDP(User Datagram Protocol)
33. ATM(Asynchronous Transfer Mode)

2 라이브러리 활용의 활용☆

(1) 라이브러리(Library) 개념

- 효율적인 프로그램 개발을 위해 필요한 프로그램을 모아 놓은 집합체로서, 프로그래밍 언어에 따라 일반적으로 도움말, 설치 파일, 샘플 코드 등을 제공한다.
- 라이브러리란 영어로 도서관을 의미하며, 필요할 때 찾아서 쓸 수 있도록 모듈화되어 제공되는 프로그램을 말한다.

(2) 라이브러리의 구성

라이브러리는 도움말, 설치 파일, 샘플 코드 등으로 구성된다.

▼ 라이브러리의 구성

구성	설명
도움말	라이브러리를 사용할 수 있도록 하는 도움말 문서
설치 파일	라이브러리를 적용하기 위해 제공되는 설치 파일
샘플 코드	라이브러리를 이해하고 손쉽게 적용하기 위해 제공되는 샘플 소스 코드

두음쌤 한마디

라이브러리의 구성

「도설샘」

도움말 / 설치 파일 / 샘플 코드

→ 도를 설파하는 선생님

(3) 라이브러리 종류

▼ 라이브러리의 종류

종류	설명
표준 라이브러리	• 프로그래밍 언어가 기본적으로 가지고 있는 라이브러리를 의미 • 각 프로그래밍 언어의 표준 라이브러리는 여러 종류의 모듈과 패키지를 가지며, 표준 라이브러리를 이용하면 별도의 파일 설치 없이 날짜와 시간 등의 기능을 이용할 수 있음
외부 라이브러리	• 표준 라이브러리와 달리 별도의 파일을 설치 • 외부 라이브러리는 누구나 개발하여 설치할 수 있으며, 인터넷 등을 이용하여 공유할 수도 있음

(4) 모듈(Module)과 패키지(Package)

라이브러리는 모듈과 패키지를 총칭하며, 모듈이 개별 파일이라면 패키지는 파일들을 모아 놓은 폴더라고 볼 수 있다.

핵인싸 퀴즈

34 ( ) 알고리즘은 송신 측으로부터 수신 측까지 데이터를 전달하는 과정에서 최적의 경로를 산출하기 위한 로직이다.

35 ( ) 알고리즘은 라우터와 라우터 간의 모든 경로를 파악한 뒤 대체 경로를 사전에 마련해 두는 알고리즘이다.

36 ( ) 알고리즘은 라우터와 라우터 간의 최단 경로를 찾고 그 최적 경로를 이용할 수 없을 경우에 다른 경로를 찾는 알고리즘이다.

정답 34. 라우팅 35. 링크 상태 (Link State) 36. 거리 벡터 (Distance Vector)

학습 Point

난수를 생성할 때 C 언어에서 rand() 함수를 사용하는 것이 출제되고 있습니다. 반드시 확인하고 넘어가세요!

잠깐! 알고가기

- rand()
- rand는 별도의 파라미터가 필요하지 않기 때문에 rand()로 사용한다.
- rand를 그냥 사용할 경우 32767까지 값이 나오기 때문에, 0에서 99 사이의 숫자를 생성하고 싶은 경우 '% 연산'을 이용하여 rand()%100으로 사용한다.

▼ 모듈과 패키지

구분	설명
모듈(Module)	• 모듈은 함수나 변수, 클래스가 저장된 파일 • 한 개의 파일에서 기능을 제공함 import (모듈명)
패키지(Package)	• 여러 개의 모듈을 한 개의 폴더에 묶어서 기능을 제공 • 패키지명과 모듈명을 import하여 불러올 수 있음 import (패키지명).(모듈명)

(5) 라이브러리 활용하기

1 표준 라이브러리 상세

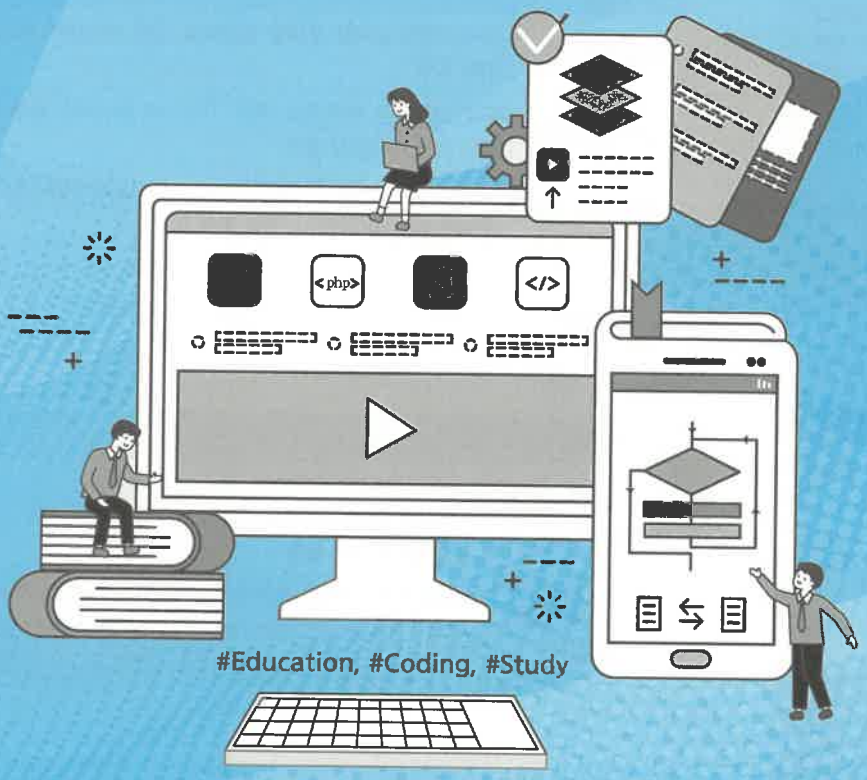
- 입출력, 문자열 등 일반적으로 많이 사용하는 라이브러리를 표준 라이브러리 형태로 제공한다.
- 표준 라이브러리의 함수들을 조합하여 새로운 함수 및 라이브러리를 만들 수 있다.

▼ 표준 라이브러리

구분			설명
문자열 연산	• 일반적인 문자열의 조작 기능을 제공		
	C언어	strcpy()	• 문자열을 복사하는 함수
		strcat()	• 문자열을 연결하는 함수
문자열 탐색 · 치환	• 문자열의 패턴을 정의하여 탐색, 치환 등의 기능을 제공		
	C언어	strstr()	• 문자열이 포함되어 있는지 알려주는 함수
	자바	String.contains	• 문자열이 포함되어 있는지 알려주는 메소드
수치 계산	• 기본적인 수치 계산 기능을 제공		
난수 생성	• 난수를 다루는 기능을 제공		
	C언어	rand()	• 임의의 난수를 생성하는 함수 • 0~32767 사이의 정수 생성
	자바	Math.random	• 임의의 난수를 생성하는 메소드 • 0~1 사이의 실수 생성
파일 경로 조작	• 파일의 경로 관련 기능을 제공		
디렉터리 생성	• 디렉터리를 생성하는 기능을 제공		
날짜 조작	• 날짜와 시간의 일자와 관련된 기능을 제공		
로그 출력	• 로그 출력 기능을 제공		

III

애플리케이션 테스트 수행



CHAPTER

01 애플리케이션 테스트 수행

1 테스트 수행★★★

(1) 테스트 개요

학습 Point

테스트 개념 및 필요성은 이 단원에서 가장 기본이 됩니다. 꼭 알고 가세요!

잠깐! 알고가기

동료 검토(Peer Review)
2~3명이 진행하는 리뷰의 형태로서 요구사항 명세서 작성자가 요구사항 명세서를 설명하고, 이해관계자들이 설명을 들으면서 결함을 발견하는 형태로 진행하는 검토 기법이다.

워크 스루(Walk Through)
검토 자료를 회의 전에 배포해서 사전검토한 후 짧은 시간 동안 회의를 진행하는 형태로 리뷰를 통해 오류를 검출하고 문서화하는 기법이다.

인스펙션(Inspection)
소프트웨어 요구, 설계, 원시 코드 등을 저작자 외의 다른 전문가 또는 팀이 검사하여 오류를 찾아내는 공식적 검토 방법이다.

두음쌤 한마디

오류 예방 관점의 정적 분석 유형
「동워인」
동료 검토 / 워크스루 / 인스펙션
→ 동쪽에서 워(War) 일으킨 사람(인)

1 테스트(Test) 개념

테스트란 개발된 응용 애플리케이션이나 시스템이 사용자가 요구하는 기능과 성능, 사용성, 안정성 등을 만족하는지 확인하고, 노출되지 않은 숨어있는 결함을 찾아내는 활동이다.

2 테스트 필요성

테스트는 오류 발견 관점, 오류 예방 관점, 품질 향상 관점에서 필요하다.

▼ 테스트 필요성

구분	설명
오류 발견 관점	프로그램에 잠재된 오류를 발견하고 이를 수정하여 올바른 프로그램을 개발하기 위해 필요
오류 예방 관점	프로그램 실행 전에 동료 검토, 워크스루, 인스펙션 등을 통해 오류를 사전에 발견하는 예방 차원에서 필요
품질 향상 관점	사용자의 요구사항 및 기대 수준을 만족하도록 반복적인 테스트를 거쳐 제품의 신뢰도를 향상하는 품질 보증을 위해 필요

3 테스트 원리

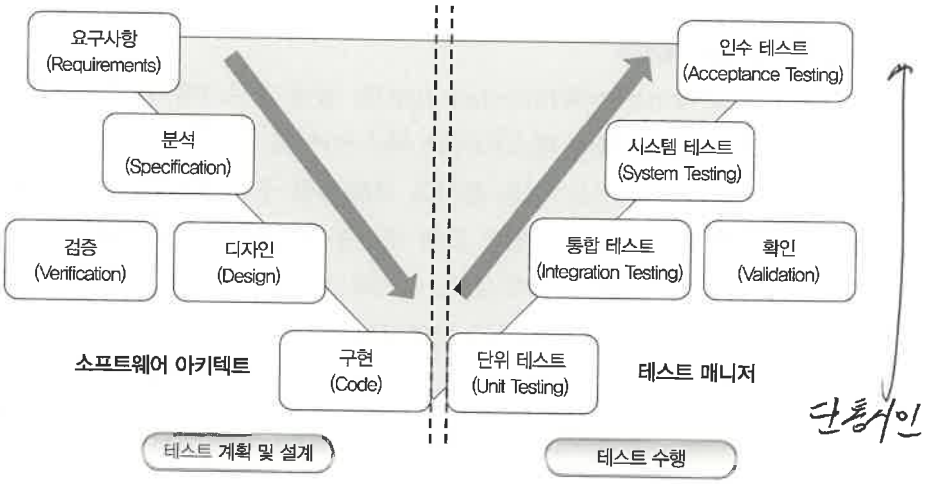
▼ 테스트 원리

원리	설명
테스팅은 결함이 존재함을 밝히는 것	• 결함이 존재함을 밝히는 활동 • 결함이 없다는 것을 증명할 수는 없음 • 결함을 줄이는 활동
완벽한 테스팅은 불가능	• 완벽하게 테스팅하려는 시도는 불필요한 시간과 자원 낭비 • 무한 경로(한 프로그램 내의 내부 조건은 무수히 많을 수 있음), 무한 입력값(입력이 가질 수 있는 모든 값의 조합이 무수히 많음)으로 인한 테스트 어려움

원리	설명
개발 초기에 테스팅 시작	• 조기 테스트 설계 시 장점은 테스팅 결과를 단시간에 알 수 있고, 테스팅 기간 단축, 재작업을 줄여 개발 기간 단축 및 결함 예방
결함 집중	• 적은 수의 모듈에서 대다수 결함이 발견 • 20%의 모듈에서 80%의 결함이 발견
살충제 패러독스	• 동일한 테스트 케이스에 의한 반복적 테스트는 새로운 버그를 찾지 못함 • 테스트 케이스의 정기적 리뷰와 개선 및 다른 시각에서의 접근이 필요
테스팅은 정황에 의존적	• 소프트웨어의 성격에 맞게 테스트 실시 • 정황과 비즈니스 영역에 따라 테스트를 다르게 수행
오류-부재의 궤변	• 요구사항을 충족시켜주지 못한다면, 결함이 없다고 해도 품질이 높다고 볼 수 없음

4 테스트 모델

소프트웨어 생명 주기의 V-모델은 소프트웨어 생명 주기의 단계별로 개발자 관점에서의 공정 과정상 검증과 사용자 관점에서의 최종 산출물에 관한 확인을 지원하기 위한 테스트 모델이다.



▲ 소프트웨어 생명 주기의 V-모델

- 테스트 과정에 필요한 역할은 소프트웨어 아키텍트와 테스트 매니저가 있다.
- V 모델에서 각각 좌측과 우측의 핵심 역할을 담당하고 서로 보완 관계에 있다.
- 소프트웨어 생명 주기는 요구사항, 분석, 디자인, 구현 또는 개발 순으로 진행되며, 프로젝트의 특성과 방법론에 따라 반복적으로 수행하는 경우도 있다.
- 테스트는 단위 테스트, 통합 테스트, 시스템 테스트, 인수 테스트의 순으로 진행된다.

학습 Point

테스트의 기본 원리에 관한 내용 설명이 나오면 원리를 답할 수 있도록 준비해 두세요!

잠깐! 알고가기

모듈(Module)
프로그램을 구성하는 구성요소의 일부로서 관련된 함수나 변수 또는 클래스를 모아 놓은 파일이다. 모듈은 다른 프로그램에서 불러와서 사용할 수 있다.

두음쌤 한마디

소프트웨어 테스트의 원리
「결완초집 살정오」
결함이 존재 / 완벽한 테스팅 불가능 / 초기에 테스팅 시작 / 결함집중 / 살충제 패러독스 / 정황에 의존 / 오류-부재의 궤변
→ 결국, 완성된 초가집에 살기 위해서 정오에 이사 옴

학습 Point

소프트웨어 생명 주기 V 모델은 그림 기반으로 이해해주세요!

잠깐! 알고가기

소프트웨어 아키텍트(Software Architect)
시스템의 아키텍처를 설계하고, 설계 결과를 프로그래밍을 포함한 다양한 방식으로 검증하고, SW 설계자들의 아키텍처 준수 여부를 모니터링하는 역할을 가진 고급 소프트웨어 엔지니어이다.

학습 Point

프로젝트 수행 단계에 따른 테스트 분류는 실제 대부분의 SW 개발 현장에서 적용하고 있습니다. 두음으로 단계를 암기하시면 도움이 됩니다.

두음쌤 한마디

프로젝트 수행 단계에 따른 테스트 분류

「단통시인」

단위 테스트 / 통합 테스트 / 시스템 테스트 / 인수 테스트

→ 단단한 통에 물을 가지고 다니는 시인

잠깐! 알고가기

인터페이스(Interface)

서로 다른 두 개의 시스템, 장치 사이에서 정보나 신호를 주고받는 경우의 접점 또는 시스템이다.

프로세스(Process)

일반적으로 CPU에 의해 처리되는 사용자 프로그램, 즉 실행 중인 프로그램이다.

학습 Point

테스트의 출발점인 단위 테스트는 개념 및 방법 위주로 학습하세요!

5 프로젝트 수행 단계에 따른 테스트 분류

▼ 프로젝트 수행 단계에 따른 테스트 분류

분류	설명
단위 테스트	• 작은 소프트웨어 단위(컴포넌트 또는 모듈)를 테스트하는 것으로서, 일반적으로 개발자가 수행하는 테스트
통합 테스트	• 단위 테스트를 통과한 모듈 사이의 인터페이스, 통합된 컴포넌트 간의 상호작용을 검증하는 테스트 • 하나의 프로세스가 완성된 경우 부분적으로 통합 테스트를 수행하기도 함
시스템 테스트	• 통합된 단위 시스템의 기능이 시스템에서 정상적으로 수행되는지를 검증하는 테스트 • 성능 및 장애 테스트가 시스템 테스트에 포함
인수 테스트	• 최종 사용자와 업무의 이해관계자 등이 테스트를 수행함으로써 개발된 제품에 대해 운영 여부를 결정하는 테스트 • 실제 업무 적용 전에 수행

6 테스트 기법에 따른 분류

① 화이트 박스 테스트

화이트 박스 테스트는(White-box test)는 응용 프로그램의 내부 구조와 동작을 검사하는 소프트웨어 테스트(구조 테스트)이다.

- 화이트 박스 테스트는 코드 분석과 프로그램 구조에 대한 지식을 바탕으로 문제가 발생할 가능성이 있는 모듈 내부를 테스트하는 방법이다.
- 화이트 박스 테스트는 내부 소스 코드의 동작을 개발자가 추적할 수 있으므로, 동작의 유효성뿐만 아니라 실행되는 과정을 확인할 수 있기 때문이다.
- 화이트 박스 테스트를 통해서 코드가 어떤 경로로 실행되며, 불필요한 코드 또는 테스트되지 못한 부분을 확인할 수 있다.

▼ 화이트 박스 테스트 유형

유형	설명
기초 경로 테스트	활동 다이어그램 또는 프로그램 코드에서 프로그램 수행 경로 도출 후 각 경로 테스트 케이스 작성 후 테스트 결과와 예상 결과를 비교하는 테스트 기법
제어 흐름 테스트	프로그램의 제어구조를 그래프 형태로 나타내고, 그래프 구성요소는 블록과 분기로 나타내어 내부 로직을 테스트하는 기법
조건 검사	프로그램의 조건문에 초점을 맞추어 검사
루프 검사	프로그램의 반복 구조에 초점을 맞추어 검사
데이터 흐름 테스트	제어 흐름 그래프에 데이터 사용현황을 추가한 그래프를 통해 테스트하는 기법
분기(Branch) 테스트	분기 문을 실행하도록 테스트 케이스를 설계하여 내부 구조를 테스트하는 기법

- 화이트 박스 테스트는 구조 테스트라고도 하며 구문 커버리지, 결정 커버리지, 조건 커버리지, 조건/결정 커버리지, 변경 조건/결정 커버리지, 다중 조건 커버리지 테스트도 포함한다.
- 화이트 박스 테스트의 검증 기준은 문장 검증 기준, 분기 검증 기준, 개별조건 검증 기준, 경로 검증 기준이 있다.

② 블랙박스 테스트

- 프로그램 외부 사용자의 요구사항 명세를 보면서 수행하는 테스트(기능 테스트)이다.
- 블랙박스 테스트는 소프트웨어의 특징, 요구사항, 설계 명세서 등에 초점을 맞춰 테스트가 이루어진다.
- 블랙박스 테스트는 기능 및 동작 위주의 테스트를 진행하기 때문에 내부 구조나 작동 원리를 알지 못해도 가능하다.

▼ 블랙박스 테스트 유형

유형	설명
균등 분할	프로그램의 입력 데이터를 여러 분류로 나누어 검사
한곗값 테스트	한곗값 테스트는 '최솟값 바로 위, 최대치 바로 아래' 등 입력값의 극한 한계를 테스트하는 기법
원인-효과 그래프 검사	그래프를 활용하여 입력 데이터 간의 관계 및 출력에 미치는 영향 분석을 체계적으로 테스트하는 기법
비교 검사	여러 버전의 프로그램에 같은 입력값을 넣어서 결과 데이터를 비교해 보는 테스트 기법

- 명세 기반 테스트인 동등분할 테스트, 경곗값 분석 테스트, 결정 테이블 테스트, 상태 전이 테스트, 유스케이스 테스트, 분류 트리 테스트, 페어 와이즈 테스트 등이 포함된다.

(2) 프로젝트 수행 단계에 따른 테스트 접근 방법

1 단위 테스트

- 단위 테스트는 테스트 가능한 단위로 작게 분리된 소프트웨어(컴포넌트 또는 모듈) 내에서 결함을 찾고 기능을 검증하는 테스트 활동이다.
- 단위 테스트는 구조 기반과 명세 기반 테스트로 분류된다.

▼ 단위 테스트

유형	설명
구조 기반 테스트	• 프로그램 내부 구조 및 복잡도를 검증하는 화이트 박스(White Box) 테스트 • 유형에는 제어구조 시험, 루프 테스트, 구문 기반, 결정 기반, 조건 기반, 조건/결정 기반, 변경 조건/결정 기반, 다중 조건 기반 커버리지 테스트 등이 있음
명세 기반 테스트	• 목적 및 실행 코드 기반으로 동작을 실행함으로써 검증하는 블랙박스(Black Box) 테스트 • 동등분할 테스트, 경곗값 분석 테스트, 결정 테이블 테스트, 상태 전이 테스트, 유스케이스 테스트, 분류 트리 테스트, 페어 와이즈 테스트 등이 있음

2 통합 테스트

- 통합 테스트는 컴포넌트 간 인터페이스 관련 오류 및 결함을 찾아내기 위한 체계적인 테스트 기법이다.
- 운영체제(OS), 파일 시스템, 하드웨어 또는 시스템 간 인터페이스와 같은 각각 다른 부분과 상호 연동이 정상적으로 작동하는지를 테스트한다.

▼ 통합 테스트 수행 방법

테스트 방안	빅뱅 방식	상향식	하향식
테스트 수행 방법	• 모든 모듈을 동시에 통합 후 테스트 수행	• 최하위 모듈부터 점진적으로 상위 모듈과 함께 테스트	• 최상위 모듈부터 하위 모듈을 통합하면서 테스트
드라이버/스텁	• 드라이버/스텁 없이 실제 모듈로 테스트	• 테스트 드라이버 필요	• 테스트 스텁 필요
장점	• 단시간 테스트 가능 • 작은 시스템에 유리	• 장애 위치 파악 쉬움 • 모든 모듈을 개발하는 시간 낭비가 필요 없음	• 장애 위치 파악 쉬움 • 이른 프로토 타입 가능 • 중요 모듈의 선 테스트 가능
단점	• 장애 위치 파악이 어려움 • 모든 모듈이 개발되어야 가능	• 중요 모듈들이 마지막 테스트 가능성 높음 • 이른 프로토타입 어려움	• 많은 스텁이 필요 • 하위 모듈들의 불충분한 테스트 수행

학습 Point

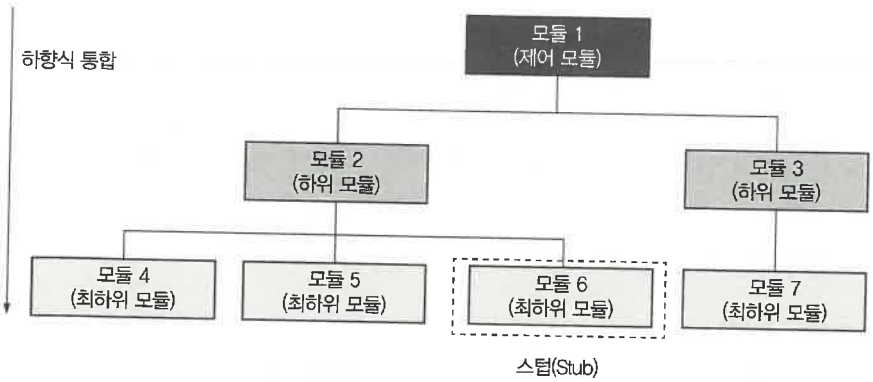
통합 테스트 중 빅뱅 방식, 상향식, 하향식에 대해서는 개념 및 테스트 수행 방식에 대해서 꼭 확인하세요!

잠깐! 알고가기

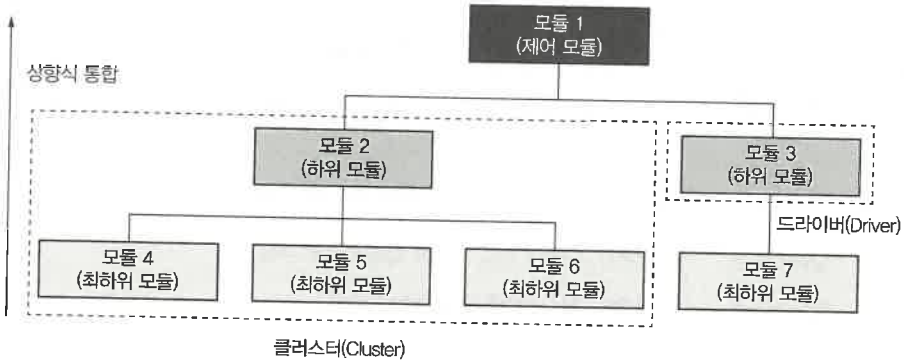
드라이버(Driver)
하위 모듈은 있지만, 상위 모듈이 없는 경우 상위의 모듈에서 데이터의 입력과 출력을 확인하기 위한 더미 모듈로 상향식 통합 테스트에 사용된다.

스텁(Stub)
상위 모듈을 테스트하기 위해 사용되는 하위 모듈 및 모든 하위 컴포넌트를 대신하는 더미 모듈로 하향식 통합 테스트에 사용된다.

프로토타입(Prototype)
정보시스템의 미완성 버전 또는 중요한 기능들이 포함된 시스템의 초기 모델이다.



▲ 하향식 통합 테스트



▲ 상향식의 통합 테스트

3 시스템 테스트

통합된 단위 시스템의 기능이 시스템에서 정상적으로 수행되는지를 검증하는 테스트로 컴퓨터 시스템을 완벽하게 검사하기 위한 목적 또는 성능 목표를 가지고 테스트한다.

▼ 시스템 테스트

유형	설명
기능적 요구사항 테스트	요구사항 명세서, 비즈니스 절차, 유스케이스 등 명세서 기반의 블랙박스(Black Box) 테스트
비기능적 요구사항 테스트	성능 테스트, 회복 테스트, 보안 테스트, 내부 시스템의 메뉴 구조, 웹 페이지의 내비게이션 등의 구조적 요소에 대한 화이트 박스(White Box) 테스트

두음쌤 한마디

통합 테스트 유형

「빅상하」

빅뱅 방식 / 상향 방식 / 하향 방식

→ 빅(큰) 상하의를 샀다.

상향식 및 하향식 통합 수행 방식

「하스 상드」

하향식(스텁) / 상향식(드라이버)

→ 하얀 스타킹을 상으로 드렸다.

학습 Point

시스템 테스트의 경우 개념 위주로 알아두세요!

잠깐! 알고가기

유스케이스(Usecase)

시스템이 액터에게 제공해야 하는 기능으로 시스템 요구사항이자, 사용자 관점에서 바라본 시스템의 기능이다.

4 인수 테스트

- 최종 사용자와 업무의 이해관계자 등이 테스트를 수행함으로써 개발된 제품에 대해 운영 여부를 결정하는 테스트이다.
- 시스템의 일부 또는 특정한 비기능적인 특성에 대해 인수 테스트를 통해 확인한다.
- 인수 테스트의 종류는 아래와 같다.

▼ 인수 테스트

종류	테스트 내용
사용자 인수 테스트	비즈니스 사용자가 시스템 사용의 적절성 여부 확인
운영상의 인수 테스트	시스템 관리자가 시스템 인수 시 수행하는 테스트 활동으로 백업/복원 시스템, 재해 복구, 사용자 관리, 정기 점검 등을 확인
계약 인수 테스트	계약상의 인수/검수 조건을 준수하는지 여부 확인
규정 인수 테스트	정부 지침, 법규, 규정 등이 규정에 맞게 개발되었는지 확인
알파 테스트	개발하는 조직 내 잠재 고객에 의해 테스트 수행
베타 테스트	실제 환경에서 고객에 의해 테스트 수행

잠깐! 알고가기

백업(Backup)
임시 보관을 일컫는 말로, 데이터 백업이라고 하며, 데이터를 미리 임시로 복제하여 문제가 일어나도 데이터를 복구할 수 있도록 준비해 두는 것을 말한다.

(3) 테스트 기반(Test Based)에 따른 테스트의 종류

테스트의 종류는 명세 기반, 구조 기반, 경험 기반 테스트로 나누어진다.

▼ 테스트 기반에 따른 분류

종류	설명	기법
명세 기반 테스트	사용자의 요구사항 분석서의 명세를 기반으로 테스트 케이스를 선정하여 테스트하는 기법	동등분할, 경곗값 분석, 결정 테이블, 상태 전이, 유스케이스, 분류 트리, 페어 와이즈 테스트 등이 있음
구조 기반 테스트	소프트웨어 내부 논리 흐름에 따라 테스트 케이스를 작성하고 결함을 발견하는 테스트 기법	구문, 결정, 조건, 조건/결정, 변경 조건/결정, 다중 조건(멀티 조건) 커버리지 테스트 등이 있음
경험 기반 테스트	유사 소프트웨어나 유사 기술 평가에서 테스터의 경험을 토대로 한, 직관과 기술 능력을 기반으로 수행하는 테스트 기법	탐색적, 오류추정, 체크 리스트, 특성 테스트 등이 있음

두음쌤 한마디

인수 테스트 유형
「사운 계규 알베」
사용자 인수 테스트 / 운영상의 인수 테스트 / 계약 인수 테스트 / 규정 인수 테스트 / 알파 테스트 / 베타 테스트

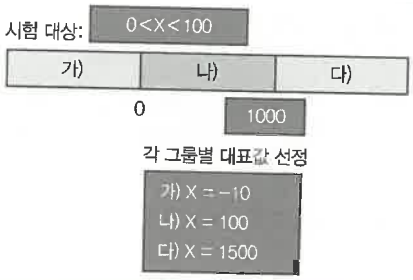
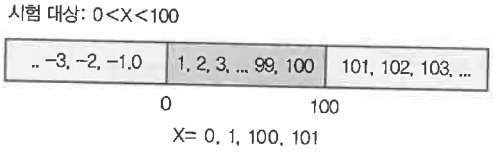
학습 Point

테스트 학습의 기본인 테스트 기반에 따른 분류는 정확하게 알아두세요! 실기 시험에 출제될 확률이 매우 높습니다.

1 명세 기반 테스트

사용자의 요구사항 분석서의 명세를 기반으로 테스트 케이스를 선정하여 테스트하는 기법이다.

▼ 명세 기반 테스트 유형

<table>
<tr><th>유형</th><th>사례</th><th>설명</th></tr>
<tr><td>동등분할 테스트</td><td> 
</td><td>입력 데이터의 영역을 유사한 도메인별로 유효 값/무효 값을 그룹핑하여 대표값 테스트 케이스를 도출하여 테스트하는 기법</td></tr>
<tr><td>경곗값 분석 테스트</td><td> 
</td><td>동등분할 후 경곗값 부분에서 오류 발생 확률이 높기에 경곗값을 포함하여 테스트 케이스를 설계하여 테스트하는 기법</td></tr>
<tr><td>결정 테이블 테스트</td><td>
<table>
<tr><th colspan="2">테스트 케이스 번호</th><th>1</th><th>2</th><th>3</th><th>4</th><th>5</th></tr>
<tr><td rowspan="3">의사결정</td><td>현금 주문</td><td>Y</td><td>Y</td><td>N</td><td>N</td><td>N</td></tr>
<tr><td>신용 카드</td><td>–</td><td>–</td><td>Y</td><td>Y</td><td>N</td></tr>
<tr><td>우수 고객</td><td>Y</td><td>N</td><td>Y</td><td>N</td><td>–</td></tr>
<tr><td rowspan="4">액션</td><td>주문 처리</td><td>√</td><td>√</td><td>√</td><td>√</td><td></td></tr>
<tr><td>주문 거부</td><td></td><td></td><td></td><td></td><td>√</td></tr>
<tr><td>10% 할인</td><td>√</td><td></td><td>√</td><td></td><td></td></tr>
<tr><td>정상 가격</td><td></td><td>√</td><td></td><td>√</td><td></td></tr>
</table>
테스트 케이스:
1) 명령은 현금 지급 또는 신용 인증일 때만 수행
2) 우수 고객은 10% 할인이 가능하도록 수행</td><td>요구사항의 논리와 발생 조건을 테이블 형태로 나열하여, 조건과 행위를 모두 조합한 테스트 기법</td></tr>
</table>

두음쌤 한마디

테스트 기반에 따른 분류

「명구경」

명세 기반 테스트 / 구조 기반 테스트 / 경험 기반 테스트

→ 명나라 구경하러 가기

학습 Point

명세 기반 테스트 유형은 두음 기반으로 유형을 암기해 주세요!

두음쌤 한마디

명세 기반 테스트 유형

「동경결상 유분폐」

동등분할 테스트 / 경곗값 분석 테스트 / 결정 테이블 테스트 / 상태 전이 테스트 / 유스케이스 테스트 / 분류 트리 테스트 / 페어 와이즈 테스트

핵인싸 퀴즈

1 ( )은/는 문제를 해결하기 위한 일련의 절차나 방법이다.

2 ( )은/는 양의 정수 n이 있을 때, 1에서부터 n까지의 자연수를 모두 곱한 값을 구하는 알고리즘이다.

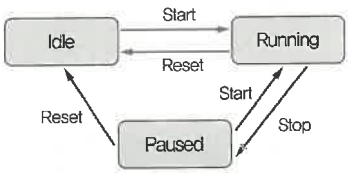
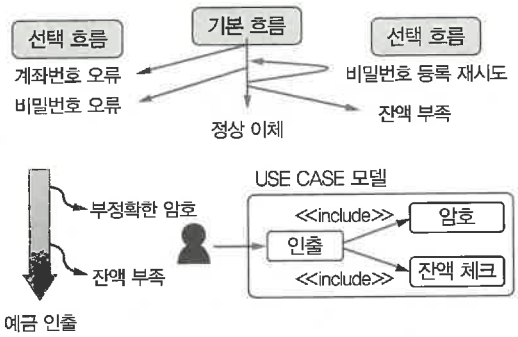
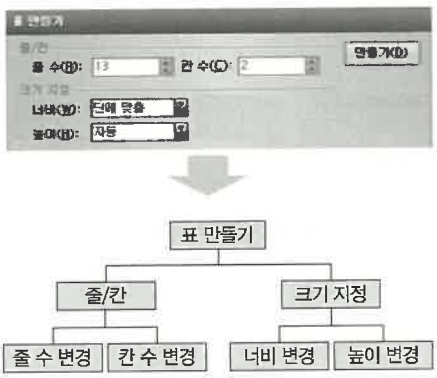
3 ( )은/는 첫 번째 항과 두 번째 항을 더해서 세 번째 항을 만들고 두 번째 항과 세 번째 항을 더해서 네 번째 항을 만드는 방법으로 다음 항을 만들어가는 수열이다.

4 ( )은/는 하나의 함수에서 자신을 다시 호출하여 작업을 수행하여 문제를 푸는 방식이다.

5 ( )은/는 주어진 배열에서 서로 인접한 두 원소를 검사하여 가장 큰 자료를 맨 뒤로 이동 시키면서 정렬하는 방식

6 ( )은/는 주어진 배열에서 가운데를 기준으로 작으면 왼쪽, 크면 오른쪽을 검색하는 방식

정답 1. 알고리즘 2. 팩토리얼 3. 피보나치 수열 4. 재귀호출 5. 버블 정렬 6. 이분 검색

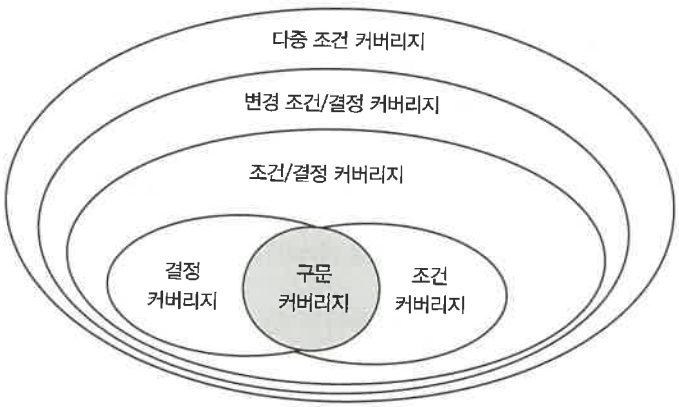
유형	사례	설명
상태 전이 테스트	 1) Test Case 1: Idle〉Running〉Idle 2) Test Case 2: Idle〉Running〉Paused〉Running	테스트 대상/시스템이나 객체의 상태를 구분하고, 이벤트에 의해 어느 한 상태에서 다른 상태로 전이되는 경우의 수를 테스트하는 기법
유스케이스 테스트		시스템이 실제 사용되는 유즈케이스로 모델링되어 있을 때 프로세스 흐름을 기반으로 테스트 케이스를 명세화하여 테스트를 수행하는 기법
분류 트리 테스트		소프트웨어의 일부 또는 전체를 트리 구조로 분석 및 표현하여 테스트 케이스를 설계하여 테스트하는 기법

<table>
<tr><th>유형</th><th>사례</th><th>설명</th></tr>
<tr><td>페어와이즈 테스트</td><td>
<table>
<tr><th>재생</th><th>볼륨</th><th>이퀄라이저</th></tr>
<tr><td>play</td><td>크게</td><td>off</td></tr>
<tr><td>play</td><td>크게</td><td>on</td></tr>
<tr><td>play</td><td>작게</td><td>off</td></tr>
<tr><td>play</td><td>작게</td><td>on</td></tr>
<tr><td>stop</td><td>크게</td><td>off</td></tr>
<tr><td>stop</td><td>크게</td><td>on</td></tr>
<tr><td>stop</td><td>작게</td><td>off</td></tr>
<tr><td>stop</td><td>작게</td><td>on</td></tr>
</table>

<table>
<tr><th>재생</th><th>볼륨</th><th>이퀄라이저</th></tr>
<tr><td>play</td><td>크게</td><td>off</td></tr>
<tr><td>play</td><td>작게</td><td>on</td></tr>
<tr><td>stop</td><td>크게</td><td>on</td></tr>
<tr><td>stop</td><td>작게</td><td>off</td></tr>
</table>
</td><td>테스트 데이터값 간에 최소한 한 번씩을 조합하는 방식이며, 이는 커버해야 할 기능적 범위를 모두 조합하는 방법에 비해 상대적으로 적은 양의 테스트 세트를 구성하여 테스트하는 기법</td></tr>
</table>

2 구조 기반 테스트

소프트웨어 내부 논리 흐름에 따라 테스트 케이스를 작성하고 결함을 발견하는 테스트 기법이다.



▲ 구조 기반 테스트 유형

▼ 구조 기반 테스트 유형

유형	설명
구문 커버리지 테스트	• 프로그램 내의 모든 명령문을 적어도 한 번 수행하는 커버리지 테스트 • 조건문 결과와 관계없이 구문 실행 개수로 계산
결정 커버리지 테스트	• 프로그램 내의 전체 결정문이 적어도 한 번은 참과 거짓의 결과를 수행하는 커버리지 테스트

학습 Point

구조 기반 테스트 유형은 유형별 개념, 동작 방식에 대해서 꼭 알고 가세요!

두음쌤 한마디

구조 기반 테스트 유형

「구결조 조변다」

구문 커버리지 테스트 / 결정 커버리지 테스트 / 조건 커버리지 테스트 / 조건/결정 커버리지 테스트 / 변경 조건/결정 커버리지 테스트 / 다중조건 커버리지 테스트

잠깐! 알고가기

커버리지(Coverage)
소프트웨어의 테스트를 수행할 때 얼마나 테스트가 충분한가를 나타내는 지표이다.

유형	설명
조건 커버리지 테스트	• 결정 명령문 내의 각 조건이 적어도 한 번은 참과 거짓의 결과가 되도록 수행하는 커버리지 테스트
조건/결정 커버리지 테스트	• 전체 조건식뿐만 아니라 개별 조건식도 참 한 번, 거짓 한 번 결과가 되도록 수행하는 커버리지 테스트
변경 조건/결정 커버리지 테스트	• 개별 조건식이 다른 개별 조건식에 영향을 받지 않고 전체 조건식의 독립적으로 영향을 주도록 함으로써 조건/결정 커버리지를 향상시킨 커버리지 테스트
다중 조건 커버리지 테스트	• 결정 조건 내 모든 개발 조건식의 모든 가능한 조합을 100% 보장하는 커버리지 테스트

① 구문 커버리지 테스트

```
if (A > 0 and B > 0) or (C > D) then
value = value + add;-----------------------------------㉠
value = value + subtract;------------------------------㉡
```

구문 커버리지 테스트는 ㉠, ㉡의 구문이 실행된다.

② 결정 커버리지 테스트

조건식 / (X,Y)	(1, 6)	(2,1)
X>=-2	T	T
Y<4	F	T
X>=-2 and Y<4	F	T

결정 커버리지

▲ 결정 커버리지 테스트

- 프로그램 내의 전체 결정문이 적어도 한 번은 참과 거짓의 결과를 수행하는 커버리지 테스트이다.

③ 조건 커버리지 테스트

- 결정 명령문 내의 각 조건이 적어도 한 번은 참과 거짓의 결과가 되도록 수행하는 커버리지 테스트이다.

핵인싸 퀴즈

7 ( )은/는 1972년에 벨 연구소의 데니스 리치가 개발한 프로그래밍 언어로, UNIX 운영체제 구현에 사용되는 언어

8 ( )은/는 소스 코드를 목적 코드로 변환하는 과정이다.

9 ( )은/는 목적 코드를 실행 파일로 변환하는 과정이다.

10 ( )은/는 저장하고자 하는 어떠한 값이 있을 때, 그 값을 주기억장치에 기억하기 위한 공간을 의미한다.

11 ( )은/는 정수, 실수 같은 여러 종류의 데이터를 식별하는 형태이다. 메모리 공간을 효율적으로 사용하고 2진수 데이터를 다양한 형태로 사용하기 위해 존재한다.

정답 7. C언어 8. 컴파일 9. 링킹 10. 변수 11. 데이터 타입

조건 커버리지

조건식 / (X,Y)	(-3, -2)	(1, 6)
X>=-2	F	T
Y<4	T	F
X>=-2 and Y<4	F	F

▲ 조건 커버리지 테스트

④ 조건/결정 커버리지 테스트

- 전체 조건식뿐만 아니라 개별 조건식도 참 한 번, 거짓 한 번 결과가 되도록 수행하는 커버리지 테스트이다.
- 조건 커버리지와 결정 커버리지를 최소한의 조합으로 달성하는 경우이다.
- 모든 개별 조건식이 참이고, 전체 조건식이 참일 경우와 모든 개별 조건식이 거짓이면서, 전체 조건식이 거짓일 경우를 의미한다.

"결정 커버리지 + 조건 커버리지 모두 만족"

조건 커버리지

조건식 / (X,Y)	(-3, -2)	(1, 6)	(2,1)
X>=-2	F	T	T
Y<4	T	F	T
X>=-2 and Y<4	F	F	T

결정 커버리지

▲ 조건/결정 커버리지 테스트

3 경험 기반 테스트

유사 소프트웨어나 유사 기술 평가에서 테스터의 경험을 기반으로 수행하는 테스트 기법이다.

▼ 경험 기반 테스트 유형

유형	설명
탐색적 테스팅	• 테스트 스크립트 또는 테스트 케이스를 문서로 작성하지 않고 경험에 바탕을 두고 탐색적으로 기능을 수행해 보면서 테스트하는 기법 • 무작위 테스팅이 아닌 중대한 테스트 위주, 테스트 엔지니어의 휴리스틱한 능력 필요, 제품을 익히면서 테스트를 설계하고 테스트를 수행

핵인싸 퀴즈

12 ( )은/는 반복문에서 강제로 반복을 종료한다.

13 ( )은/는 반복문에서 강제로 다음 반복을 진행한다.

14 ( )은/는 동일한 타입의 변수를 여러개 만드는 경우 사용한다.

정답 12. break 13. continue 14. 배열

학습 Point

경험 기반 테스트는 명세 기반 테스트, 구조 기반 테스트와 비교해서 알고 가세요!

두음쌤 한마디

경험 기반 테스트 유형

「탐오체특」

탐색적 테스팅 / 오류추정 / 체크 리스트 / 특성 테스팅

→ 탐관오리를 체포해서 특진함

유형	설명
오류 추정	• 테스트 중인 컴포넌트나 시스템에서 유입된 결과로 어떤 결함이 발생할 것인지를 테스터의 경험을 사용하여 예측하고, 해당 결함만을 중점적으로 검출하는 테스트 기법
체크 리스트	• 테스트하고 평가해야 할 내용과 경험을 분류하여 나열한 이후 하나씩 확인하는 테스트 기법 • 체계적 도출보다는 경험과 노하우를 정리하고 목록화하여 재사용 목적으로 활용
특성 테스팅	• 국제표준인 ISO/IEC 9126 등의 품질모델에 있는 품질 특성을 염두에 두고 이를 근간으로 경험적으로 테스트 케이스를 설계하고 테스트하는 기법 • ISO/IEC 9126-2에 명기된 품질 특성 활용

(4) 테스트 목적에 따른 테스트의 종류

학습 Point

각각의 테스트 유형의 개념을 가볍게 알고 가세요!

테스트 목적에 따른 분류에는 회복 테스트, 안전 테스트, 강도 테스트, 성능 테스트, 구조 테스트, 회귀 테스트, 병행 테스트가 있다.

▼ 테스트 목적에 따른 분류

두음쌤 한마디

테스트 목적에 따른 분류

「회안강성 구회병」

회복 테스트 / 안전 테스트 / 강도 테스트 / 성능 테스트 / 구조 테스트 / 회귀 테스트 / 병행 테스트

→ 중국의 회안 강성에 사는 구회병씨

분류	설명
회복(Recovery) 테스트	시스템에 고의로 실패를 유도하고 시스템의 정상적 복귀 여부를 테스트하는 기법
안전(Security) 테스트	불법적인 소프트웨어가 접근하여 시스템을 파괴하지 못하도록 소스 코드 내의 보안 결함을 미리 점검하는 테스트 기법
강도(Stress) 테스트	시스템에 과다 정보량을 부과하여 과부하 시에도 시스템이 정상적으로 작동되는지를 검증하는 테스트 기법
성능(Performance) 테스트	사용자의 이벤트에 시스템이 응답하는 시간, 특정 시간 내에 처리하는 업무량, 사용자 요구에 시스템이 반응하는 속도 등을 측정하는 테스트 기법
구조(Structure) 테스트	소프트웨어 내부 논리 흐름에 따라 테스트 케이스를 작성하고 결함을 발견하는 테스트 기법
회귀(Regression) 테스트	변경 또는 수정된 코드에 대하여 새로운 결함발견 여부를 평가하는 테스트 기법
병행(Parallel) 테스트	변경된 시스템과 기존 시스템에 동일한 데이터를 입력 후 결과를 비교하는 테스트 기법

(5) 테스트 자동화 도구

1 테스트 자동화 도구 배경

- 소프트웨어 테스트는 소프트웨어 개발에 소요되는 총 시간과 비용의 절반 이상을 차지하기도 할 정도로 많은 자원이 투입되는 프로세스이다.
- 따라서 테스트의 정확성을 유지하면서 시간과 비용을 줄일 수 있는 자동화 도구가 매우 중요하게 부각되고 있다.

2 테스트 자동화

① 테스트 자동화 개념

테스트 도구를 활용하여 반복적인 테스트 작업을 스크립트 형태로 구현함으로써, 테스트 시간 단축과 인력 투입 비용을 최소화하는 한편, 쉽고 효율적인 테스트를 수행하는 기법이다.

② 테스트 자동화 도구의 장단점

▼ 테스트 자동화 도구의 장단점

장점	단점
• 반복되는 테스트 데이터 재입력과 재구성 작업의 자동화로 인력과 시간 최소화 • 사용자 요구 기능의 일관성 검증에 유리 • 테스트 결괏값에 대한 객관적인 평가 기준 제공 • 향상된 요구사항 정의, 성능 및 스트레스 테스트, 품질 측정 최적화 • 빌드 확인, 회귀, 다중 플랫폼 호환성, 소프트웨어 구성, 기본 테스트 등의 향상된 테스트 품질 보장 • 테스트 결과의 통계 작업과 그래프 등 다양한 표시 형태 제공 • UI가 없는 서비스의 경우에도 정밀한 테스트 가능	• 도구 도입 후 도구 사용 방법에 대한 교육 및 학습이 필요 • 도구를 프로세스 단계별로 적용하기 위한 시간, 비용, 노력이 필요 • 비공개 상용 도구의 경우 고가이고, 인력과 교육에 대한 유지 관리 비용이 높아 추가 투자가 필요

③ 테스트 자동화 수행 시 고려사항

▼ 테스트 자동화 수행 시 고려사항

고려사항	설명
테스트 절차	테스트 절차를 고려하여 재사용 및 측정이 불가능한 테스트 프로그램은 제외
설계기준	반복적인 빌드에서 스크립트 재사용성이 가능하도록 설계기준 고려
도구 사용	도구의 한계성으로 모든 수동 테스트 과정을 자동화할 수 없기 때문에 용도에 맞는 적절한 도구 사용 필요
도구 환경 설정 및 습득 기간	프로젝트의 지연 방지를 위해 도구 환경 설정 및 습득 기간 고려
테스트 엔지니어 투입 시기 및 계획	테스트 엔지니어 늦은 투입은 프로젝트의 이해 부족으로 불완전한 테스트 초래 적절한 투입 시기와 계획을 프로젝트 초기에 수립 필요

3 테스트 자동화 도구 유형

① 정적 분석 도구(Static Analysis Tools)

• 정적 분석 도구는 만들어진 애플리케이션을 실행하지 않고 분석하는 방법이다.

핵인싸 퀴즈

15 ( )은/는 연산과 대입을 한 꺼번에 표현하는 연산자로 +=, -=, *=, /=, %= 등이 있다.

16 ( )은/는 소프트웨어 세계에 구현할 대상이다.

정답 15. 복합대입 연산자 16. 객체

잠깐! 알고가기

빌드(Build)
소스 코드 파일을 실행 가능한 소프트웨어 산출물로 만드는 일련의 과정을 말한다.

스크립트(Script)
컴퓨터 프로그래밍에서의 스크립트란 (컴퓨터 프로세서가 아닌) 다른 프로그램에 의해 번역되거나 수행되는 프로그램이나 명령어들의 나열을 말한다.

학습 Point

테스트 자동화 도구가 필요하게 된 배경, 자동화 도구의 유형은 이해 기반으로 알고 가세요!

두음쌤 한마디

테스트 자동화 도구 유형

「정실성통」

정적 분석 도구 / 테스트 실행 도구 / 성능 테스트 도구 / 테스트 통제 도구

→ 정신적으로 실성한 환자들 통제 도구

잠깐! 알고가기

테일러링(Tailoring)

프로젝트의 특성과 필요에 따라 소프트웨어 개발 프로세스를 적합한 규모로 가공하는 과정 및 방법론이다.

형상 관리(Configuration Management)

SDLC 단계의 산출물을 체계적으로 관리하여 S/W의 가시성 및 추적성을 부여하여 품질 보증을 향상시키는 관리적 활동이다.

- 대부분은 소스 코드에 대한 코딩 표준, 코딩 스타일, 코드 복잡도 및 남은 결함을 발견하기 위하여 사용한다.
- 테스트를 수행하는 사람이 작성된 소스 코드에 대한 이해를 바탕으로 도구를 이용해서 분석하는 것을 말한다.

② 테스트 실행 도구(Test Execution Tools)

- 테스트를 위해 작성된 스크립트를 실행하는 도구이다.
- 작성된 스크립트는 스크립트마다 특정 데이터와 테스트 수행 방법을 포함하고 있다.
- 데이터 주도 접근 방식과 키워드 주도 접근 방식으로 나눌 수 있다.

▼ 테스트 실행 도구 유형

도구 유형	설명
데이터 주도 접근 방식	• 테스트 데이터를 스프레드시트에 저장하고, 이 데이터를 읽고 실행 • 다양한 테스트 데이터를 이용하여 동일한 테스트 케이스를 반복해서 실행할 수 있으며, 스크립트 언어에 익숙지 않은 테스터도 미리 작성된 스크립트에 테스트 데이터만 추가하여 쉽게 테스트를 수행
키워드 주도 접근 방식	• 일반적으로 테스트를 수행할 동작을 나타내는 키워드와 테스트 데이터를 스프레드시트에 저장 • 키워드를 이용하여 테스트 수행 동작을 정의할 수 있으며, 테스트 대상 애플리케이션의 특성에 맞추어 키워드에 대해 테일러링을 수행할 수 있음

③ 성능 테스트 도구(Performance Test Tools)

애플리케이션의 처리량, 응답시간, 경과시간, 자원 사용률에 대해 가상의 사용자를 생성하고 테스트를 수행함으로써 성능 목표를 달성하였는지를 확인하는 도구이다.

④ 테스트 통제 도구(Test Control Tools)

- 테스트 통제 도구에는 테스트 계획 및 관리를 위한 테스트 관리 도구, 테스트 수행에 필요한 데이터와 도구를 관리하는 형상 관리 도구, 테스트에서 발생한 결함에 대해 관리하거나 협업을 지원하기 위한 결함 추적/관리 도구 등이 있다.
- 조직의 요구사항에 최적화된 형태의 정보를 생성, 관리하기 위하여 스프레드시트 등 다른 도구들과 연계하여 사용할 수도 있다.

⑤ 테스트 하네스(Test Harness)

- 애플리케이션 컴포넌트 및 모듈을 테스트하는 환경의 일부분으로, 테스트를 지원하기 위한 코드와 데이터를 말하며, 단위 또는 모듈 테스트에 사용하기 위해 코드 개발자가 작성한다.

• 테스트 하네스의 구성요소는 아래와 같다.

▼ 테스트 하네스 구성 요소

구성요소	설명
테스트 드라이버 (Test Driver)	테스트 대상 하위 모듈을 호출하고, 파라미터를 전달하고, 모듈 테스트 수행 후의 결과를 도출하는 등 상향식 테스트에 필요
테스트 스텁 (Test Stub)	제어 모듈이 호출하는 타 모듈의 기능을 단순히 수행하는 도구로 하향식 테스트에 필요
테스트 슈트 (Test Suites)	테스트 대상 컴포넌트나 모듈, 시스템에 사용되는 테스트 케이스의 집합
테스트 케이스 (Test Case)	입력값, 실행 조건, 기대 결과 등의 집합
테스트 스크립트 (Test Script)	자동화된 테스트 실행 절차에 대한 명세
목 오브젝트 (Mock Object)	사용자의 행위를 조건부로 사전에 입력해 두면, 그 상황에 예정된 행위를 수행하는 객체

⑥ 테스트 단계별 테스트 자동화 도구

테스트 계획, 테스트 분석/설계, 테스트 수행, 테스트 통제 등의 테스트 활동에 따라 각각 요구사항 관리, 테스트 케이스 생성, 커버리지 분석, 정적/동적 테스트 수행, 성능 테스트, 모니터링, 형상 관리, 테스트 관리, 결함 추적/관리 등을 지원하는 테스트 도구가 있다.

▼ 테스트 단계별 테스트 자동화 도구

테스트 단계	자동화 도구	설명
테스트 계획	요구사항 관리	고객 요구사항 정의 및 요구사항 관리를 지원
테스트 분석/설계	테스트 케이스 생성	테스트 기법에 따른 테스트 케이스 작성과 테스트 데이터 생성을 지원
테스트 수행	테스트 자동화	기능 테스트와 UI 테스트 등 단위 테스트 및 통합 테스트를 지원
	정적 분석	코딩 표준, 런타임 오류 등을 검증
	동적 분석	대상 시스템 시뮬레이션을 통한 오류 검출
	성능 테스트	부하 생성기 등을 이용하여 가상 사용자를 생성하고, 시스템의 처리 능력을 측정하는 도구
	모니터링	시스템 자원(CPU, 메모리 등)의 상태 확인 및 분석 지원
테스트 통제	커버리지 측정	테스트 완료 후 테스트 충분성 여부 검증 지원
	형상 관리	테스트 수행에 필요한 도구, 데이터 및 문서 관리
	테스트 관리	전반적인 테스트 계획 및 활동에 대한 관리
	결함 추적/관리	테스트에서 발생한 결함 추적 및 관리 활동 지원

학습 Point

테스트 장치 구성요소는 각종 시험에서 자주 출제되는 토픽입니다. 이번 기회를 통해서 알고 가세요!

두음쌤 한마디

테스트 장치 구성요소

「드 스슈케 스목」

드라이버 / 스텁 / 슈트 / 케이스 / 스크립트 / 목 오브젝트

4 테스트 도구 평가방법 및 요소

학습 Point

테스트 단계별 테스트 자동화 도구는 테스트 프로세스를 기준으로 매칭되는 자동화 도구를 이해하고 넘어가세요!

▼ 테스트 도구 평가방법 및 요소

구분	설명
테스트 도구 평가방법	• 테스트 도구가 지원하는 모든 사항에 대해 종류별로 나열하고 기록 • 테스트에 해당하는 요소에 대한 목표 또는 평균값을 설정하고 설정된 기준으로 평가 • 비용을 최소화하기 위해 필수적인 몇 가지 요소만을 고려해야 할 경우 최소한의 요구사항을 만족하는 도구 선정
테스트 도구 평가요소	• 테스트 도구 사용 편의성, 다중 사용자 접속, 결함 추적, 도구 기능성, 보고 능력, 성능 및 강도 테스트, 버전 제어, 테스트 계획과 관리, 가격, 테스트 도구 제조사의 지원 능력 등 객관적이고 수치화되는 항목을 이용 평가

(6) 테스트 환경 구축

1 테스트 환경 구축 개념

개발된 응용 소프트웨어가 실제 운영 시스템에서 정상적으로 작동되는지 테스트하기 위하여 실제 운영 시스템과 동일한(또는 유사한) 사양의 하드웨어, 소프트웨어, 네트워크 등의 환경 시설을 구축하는 활동이다.

2 테스트 환경 구축 유형

잠깐! 알고가기

WAS(Web Application Server)
사용자 요청 스레드를 처리하고, 데이터베이스에 접속하여 SQL 쿼리문에 대한 결과 값을 반환하는 역할을 수행하는 서버이다.

가상 머신(Virtual Machine)
컴퓨터상에 가상으로 컴퓨터를 구동시키는 것으로 물리적인 하드웨어를 가상화하여, 하나의 물리적 하드웨어 상에서 여러 컴퓨터가 구동되는 것처럼 에뮬레이션하는 소프트웨어이다.

▼ 테스트 환경 구축 유형

유형	설명
하드웨어 기반의 테스트 환경 구축	• 서버 장비(WAS 서버, DBMS 서버), 클라이언트 장비(노트북 또는 PC), 네트워크(내부 LAN 또는 공용 인터넷 라인) 장비를 설치하는 작업
소프트웨어 기반의 테스트 환경 구축	• 구축된 하드웨어 환경에 테스트할 응용 소프트웨어를 설치하고 필요한 데이터를 구축하는 작업
가상 시스템 기반의 테스트 환경 구축	• 물리적으로 개발 환경 및 운영 환경과 별개로 독립된 테스트 환경을 구축하기 힘든 경우 사용 • 가상 머신 기반의 서버 또는 클라우드 환경을 이용하여 테스트 환경을 구축하고, 네트워크는 VLAN과 같은 기법을 이용하여 논리적 분할 환경을 구축하는 작업

3 테스트 데이터

▼ 테스트 데이터

구분	설명
개념	• 컴퓨터의 동작이나 시스템의 적합성을 시험하기 위해 특별히 개발된 데이터 집합 • 프로그램의 기능을 하나씩 순번에 따라 확실하게 테스트할 수 있도록 조건을 갖춘 데이터

구분	설명
필요성	• 테스트 수행 시 잘못된 데이터를 사용하면 잘못된 결과가 도출되어 시간을 낭비하고, 비용만 소진하는 결과가 나옴 • 테스트를 효율적으로 운용하고 데이터의 기밀을 유지하며 신뢰 및 예측 가능한 테스트를 위해 테스트 데이터 준비가 필요
유형	• 선행된 연산에 의해 얻어진 실제 데이터와 인위적으로 만들어진 가상 데이터로 구분
준비	• 실제 데이터는 연산에 의해 준비하거나 실제 운영 데이터를 복제하여 준비 가능 • 가상의 데이터는 스크립트를 통해서 생성 가능

4 테스트 조건

▼ 테스트 조건

구분	설명
테스트 시작 조건	• 테스트 계획의 수립, 사용자 요구사항에 대한 테스트 명세의 작성, 투입 조직 및 참여 인력의 역할과 책임의 정의, 테스트 일정의 확정, 테스트 환경의 구축 등이 완료되었을 때 테스트 시작 조건 정의 가능 • 단계별 또는 회차별 테스트 수행을 위해 모든 조건을 만족하지 않아도 테스트 시작 가능
테스트 종료 조건	• 정상적인 테스트를 모두 수행한 경우, 차기 일정의 도래로 테스트 일정이 만료되었을 경우, 테스트에 소요되는 비용을 모두 소진한 경우 등 업무 기능의 중요도에 따라 조건 설정 변경 가능
테스트 성공과 실패의 판단 기준	• 기능 및 비기능 테스트 시나리오에 기술된 예상 결과를 만족하면 성공으로, 아니라면 실패로 판단 가능 • 동일한 데이터 또는 이벤트를 중복하여 테스트하여도 여전히 이전 테스트와 같은 결과가 나올 때 성공으로 판단 가능

학습 Point

테스트 조건은 가볍게 읽고 넘어가세요!

(7) 테스트 수행 프로세스

1 테스트 계획서 확인

기존에 작성된 테스트 계획서를 활용하여, 사용자의 요구사항에 관한 내용을 일관성 있게 계획, 설계, 실행할 수 있도록 전체적인 테스트 순서와 방법을 확인한다.

▼ 테스트 계획서 확인

절차	설명
테스트 목적과 범위 확인	• 테스트 전략 문서가 있는 경우에는 정렬된 테스트의 목적이 상세하게 기술 • 테스트 범위는 성능 테스트의 기간과 진행 자원에 따라 테스트 범위가 변경
테스트 실행 계획 확인	• 테스트 시작/종료 기준 확인

핵인싸 퀴즈

17 ( )은/는 객체를 만들어 내기 위한 설계도 혹은 틀이다.

18 ( )은/는 설계도를 바탕으로 소프트웨어 세계에 실체화된 형태이다.

정답 17. 클래스 18. 인스턴스

절차	설명
테스트 케이스 확인	• 테스트 케이스로 선정된 주요 업무는 테스트 시나리오에 대한 세부적인 테스트(실제 업무 처리) 절차를 정의 • 해당 절차에 대해 정상 처리 여부를 판단하기 위한 구체적인 확인 방안도 체크해야 함
테스트 데이터 확인	• 테스트가 진행되는 동안 데이터의 변경 작업(생성/수정/삭제)이 발생할 수 있으므로, 테스트 상황에 맞는 테스트 데이터의 생성 및 유지 방침을 확인해야 함 • 테스트 수행 시 별도의 테스트 데이터가 구성되면 좋으나 만일 실 운영 데이터를 사용할 경우에는 특히 데이터의 유지 절차를 고려해야 함
테스트 환경 확인	• 테스트 환경은 테스트 대상 장비, 테스트 도구, 테스트용 데이터, 네트워크 환경 등 테스트와 관련된 일반적인 IT 자원과 도구를 명시 • 자원별 담당 부서는 사전 점검 항목을 참고하여 테스트 수행을 위한 최소 수준 이상의 환경 구성 여부를 확인

2 기능 단위 테스트 수행

학습 Point

테스트 수행 프로세스는 프로젝트의 성격과 업무 특성에 따라 달라지며, 표준 테스트 프로세스 기반으로 이해해야 합니다.

- 사용자의 요구사항을 반영한 프로그램의 업무 흐름에 따라 시나리오를 작성하고 테스트를 수행한다.
- 테스트 수행 중 해당 프로그램의 서버 모듈, 화면 모듈, 데이터 입출력, 인터페이스 등에 대한 테스트를 실시한다.

▼ 기능 단위 테스트 수행

구분	설명
업무별 기능 정의	• 서버 모듈 정의 • 화면 모듈 정의 • 데이터 입출력 정의 • 인터페이스 정의
테스트 항목 정의	• 업무별 기능에 따라 각각의 기능 단위가 연계되어 단위 테스트가 이루어지며, 테스트 내용에 따라 테스트 항목이 변경
단위 테스트 수행	• 테스트 환경 관련 사항들 준비 • 입출력 확인 • 산술식 또는 단순 계산 결과 확인 • 연동 시스템 확인 • 배치 처리와 인터페이스 확인 • 인쇄 및 보고서 확인
통합 테스트 수행	• 통합 테스트 환경 구성 • 테스트 실행 및 결과 기록 • 통합 테스트 종료 여부 평가
성능 테스트 수행	• 성능 테스트 계획서 확인 • 성능 테스트를 수행하고 결과 보고서 작성 • 통합 테스트 수행 결과 목표 TPS와 허용 가능 응답시간, 허용 가능 시스템 사용률 등을 만족하는지 확인 • 미달 시 병목 지점을 개선하는 튜닝 작업을 반복적으로 실시

잠깐! 알고가기

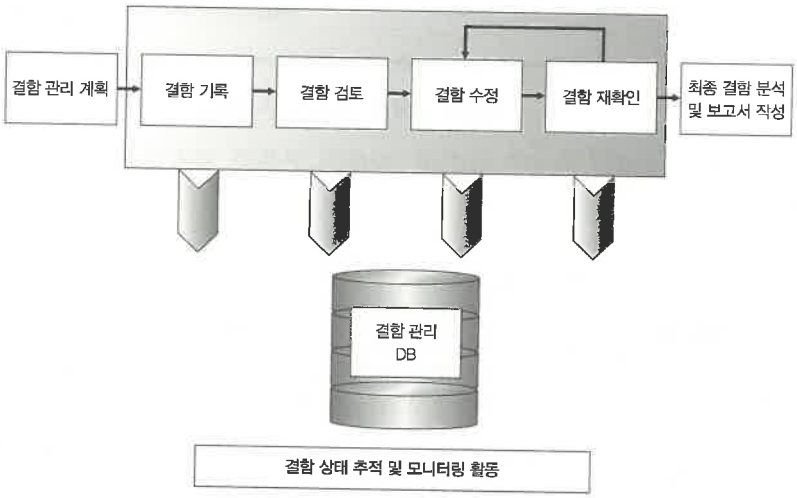
튜닝 작업
애플리케이션의 병목을 찾아서 부분적으로 수정하거나, 하드웨어나 미들웨어의 구성을 개선하는 작업

2 결함 관리★★☆

(1) 결함의 개념

- 결함은 프로그램과 명세서 간의 차이, 업무 내용 불일치이다.
- 결함은 기대 결과와 실제 관찰 결과 간의 차이이다.
- 시스템이 사용자가 기대하는 타당한 기대치를 만족시키지 못할 때 변경이 필요한 모든 것은 결함이다.

(2) 결함 관리 프로세스



▲ 결함 관리 프로세스

결함 관리 프로세스는 7개의 활동으로 구성된다.

▼ 결함 관리 프로세스

프로세스	설명
결함 관리 계획	결함 관리 계획은 전체 프로세스에서 결함 관리에 대한 일정, 인력, 업무 프로세스를 확보하여 계획을 수립하는 것
결함 기록	테스터는 발견된 결함에 대한 정보를 결함 관리 DB에 기록
결함 검토	등록된 결함에 있어서 주요 내용을 검토하고, 결함을 수정할 개발자에게 전달
결함 수정	개발자는 할당된 결함의 프로그램 수정
결함 재확인	테스터는 개발자가 수정한 내용을 확인하고 다시 테스트 수행
결함 상태 추적 및 모니터링 활동	결함 관리 팀장은 결함 관리 데이터베이스를 이용하여 대시보드 또는 게시판 형태의 서비스 제공
최종 결함 분석 및 보고서 작성	발견된 결함에 관한 내용과 이해관계자들의 의견이 반영된 보고서를 작성하고 결함 관리 종료

학습 Point

결함 관리 프로세스는 두음 기반으로 암기하세요. 시험에 나올 확률이 높습니다!

두음쌤 한마디

결함 관리 프로세스

「계기검수 재추최」

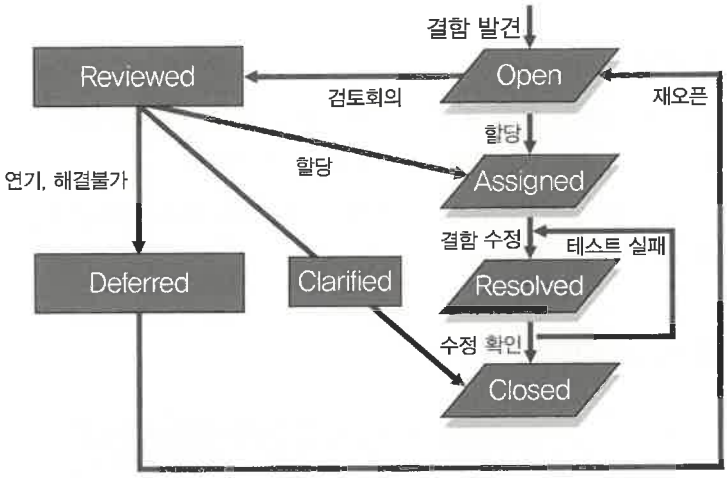
결함 관리 계획 / 결함 기록 / 결함 검토 / 결함 수정 / 결함 재확인 / 결함 상태 추적 및 모니터링 활동 / 최종 결함 분석 및 보고서 작성

→ 비행기 계기판을 검수하고, 업체를 재추천하기로 최종 결정함

(3) 결함의 상태 및 추적

학습 Point

결함의 상태 및 추적 업무 흐름 프로세스는 이해 기반으로 학습하세요! 두음이 외워지지 않으면 넘어가세요!



▲ 결함의 상태 및 추적 업무 흐름도

결함은 여러 상태를 가지고 있으며, 상태의 변화를 지속해서 추적 관리를 해야 한다.

두음쌤 한마디

결함의 상태 및 추적 업무 흐름 프로세스

「등검할수 조종해」

결함 등록 / 결함 검토 / 결함 할당 / 결함 수정 / 결함 조치 보류 / 결함 종료 / 결함 해제 → 등산을 검단산으로 할 수 있으니 일정 조종(정)해!

▼ 결함의 상태 및 추적 업무 흐름 프로세스

결함 상태	설명
결함 등록(Open)	테스터와 품질 관리(QC) 담당자에 의해 결함이 처음 발견되어 등록되었지만, 아직 분석되지 않은 상태
결함 검토(Reviewed)	등록된 결함을 담당 모듈 개발자, 테스터, 프로그램 리더, 품질 관리(QC) 담당자와 검토하는 상태
결함 할당(Assigned)	결함의 영향 분석 및 수정을 위해 개발자와 문제 해결 담당자에게 할당된 상태
결함 수정(Resolved)	개발자에 의해 결함의 수정이 완료된 상태
결함 조치 보류(Deferred)	수정이 필요한 결함이지만 현재 수정이 불가능해서 연기된 상태로서 우선순위, 일정 등을 고려하여 재오픈을 준비하는 상태
결함 종료(Closed)	발견된 결함이 해결되고 테스터와 품질 관리(QC) 담당자에 의해 종료 승인을 한 상태
결함 해제(Clarified)	테스터, 프로그램 리더, 품질 관리(QC) 담당자가 결함을 검토한 결과, 결함이 아니라고 판명된 경우

(4) 결함 분류

• 결함 유형은 시스템 결함, 기능 결함, GUI 결함, 문서 결함으로 분류한다.

▼ 결함 분류

유형	설명	세부 유형
시스템 결함	• 비정상적인 종료/중단, 응답 시간 지연, 데이터베이스 에러 등 주로 애플리케이션 환경과 데이터베이스 처리에서 발생하는 결함	비정상적인 종료/중단, 응답 시간 지연, 데이터베이스 에러
기능 결함	• 사용자의 요구사항 미반영/불일치, 부정확한 비즈니스 프로세스, 스크립트 에러, 타 시스템 연동 시 오류 등 기획, 설계, 업무 시나리오 단계에서 발생한 결함	요구사항 미반영/불일치, 부정확한 비즈니스 프로세스, 스크립트 에러, 시스템 연동 시 오류
GUI 결함	• GUI 결함은 응용 프로그램의 UI 비일관성, 부정확한 커서/메시지, 데이터 타입의 표시 오류 등으로 사용자 화면 설계에서 발생한 결함	응용 프로그램 UI 비일관성, 부정확한 커서/메시지, 데이터 타입의 표시 오류
문서 결함	• 기획자, 사용자, 개발자 간의 의사소통과 기록이 원활하지 않은 경우에 발생하는 결함 • 사용자의 온라인/오프라인 매뉴얼의 불일치, 요구사항 분석서와 기능 요구사항의 불일치로 인한 불완전한 상태의 문서의 경우	

학습 Point

결함의 분류는 사례 기반으로 어느 유형에 속하는지 확인하세요!

두음쌤 한마디

결함 분류

「시기지문」

시스템 결함 / 기능 결함 / GUI 결함 / 문서 결함

→ 시기가 지난 문제

(5) 결함 심각도

결함 심각도는 여러 개의 결함 중 전체 시스템에 결함이 미치는 영향을 레벨별로 나타내고, 우선순위를 High, Medium, Low로 정하는 것을 말한다.

▼ 결함 심각도에 따른 우선순위

우선순위	설명
High	• 시스템이 중단(또는 다운)되어 더 이상 프로세스를 진행할 수 없게 만드는 결함 • 시스템의 핵심 요구사항 미구현, 시스템 다운, 장시간 시스템 응답 지연, 시스템 복구 후 데이터 왜곡 등의 결함
Medium	• 시스템의 흐름에 영향을 미치는 결함 • 부정확한 기능, 부정확한 업무 프로세스, 데이터 필드 형식의 오류, 데이터베이스 에러, 보안 관련 오류 등의 결함
Low	• 시스템의 흐름에는 영향을 미치지 않는 결함이나 상황에 맞지 않는 용도와 화면 구성 결함 • 부정확한 GUI 및 메시지, 에러 시 메시지 미출력, 화면상의 문법/철자 오류 등의 결함

학습 Point

결함 심각도별 우선순위에 대해서는 정확하게 알고 가세요!

두음쌤 한마디

결함 우선순위 분류

「하미로」

High / Medium / Low

→ 하얀 미로

(6) 결함 관리 항목

테스트 수행 후 발견된 결함은 결함 관리 시스템에 등록하여 관리해야 하며, 등록 시 다음 항목들은 필수로 등록한다.

▼ 결함 관리 항목

항목	설명
결함 내용	• 테스트 중 발생한 결함 내용을 상세히 기재 • 조치자가 결함 내용을 명확히 이해할 수 있을 정도로 상세히 기술
결함 ID	• 결함 내용별로 부여하는 결함 번호를 기재
결함 유형	• 결함 내용에 따라 결함이 발생한 유형의 코드를 기재
발견일	• 결함을 발견한 일자
심각도	• 애플리케이션에 발생한 결함이 어떤 영향을 끼치며, 그 결함이 얼마나 치명적인지를 나타내는 척도 • 치명적(Critical) 결함 〉 주요(Major) 결함 〉 보통(Normal) 결함 〉 경미한(Minor) 결함 〉 단순(Simple) 결함
우선순위	• 발생한 결함이 얼마나 빠르게 처리되어야 하는지를 결정하는 척도 • High 〉 Medium 〉 Low
시정 조치 예정일	• 결함에 대한 조치를 완료할 일자
수정 담당자	• 결함이 발생한 내용을 보완한 담당자 명을 기재
재 테스트 결과	• 결함 내용이 정상적으로 수정되었는지 확인하기 위해 테스트한 결과를 작성
종료일	• 결함에 대해 모든 조치가 완료된 일자

(7) 결함 관리 프로세스

1 애플리케이션 테스트 수행으로 발견된 결함의 기록 및 분류

▼ 결함 기록 및 분류

순서	절차	내용
1	발견된 결함을 관리할 테스트 관리 양식 작성	• 관리 항목 정의: 테스트 ID, 테스트 시나리오 명, 테스트 케이스 명, 테스트 수행 결과, 테스트 담당자/테스트 계획일, 테스트 수행일, 결함 내용, 기타 업무적 필요 사항을 항목으로 생성 • 관리할 항목을 연결하여 테스트 관리 양식을 정의하고 관리가 용이하도록 항목들을 배치하여 양식 정의
2	테스트 관리 양식에 발견된 결함 기록	• 수행한 테스트의 테스트 ID, 테스트 시나리오 명, 테스트 케이스 ID와 명칭 기록 • 수행한 테스트 결과 기록 • 테스트를 수행한 담당자 이름과 계획 시간, 수행 시간 기록 • 수행한 테스트의 결함 내용 기록 • 수행한 테스트 결과를 테스트 관리 양식에 기록
3	기록된 테스트 관리 양식의 결함을 유형별로 분류하고 결함 상태 확인	• 결함 상태는 결함 등록(Open), 결함 검토(Reviewed), 결함 할당(Assigned), 결함 수정(Resolved), 결함 조치 보류(Deferred), 결함 종료(Closed), 결함 해제(Clarified)로 분류됨 • 정의한 결함 유형을 이용하여 결함 관리 현황에 결함 상태를 작성

핵인싸 퀴즈

19 ( )은/는 접근제어 지시자 중 클래스 내부에서만 접근을 허용한다.

20 ( )은/는 접근제어 지시자 중 클래스 내부와 동일 패키지, 상속받은 클래스에서만 접근을 허용한다.

정답 19. private 20. protected

학습 Point

결함 관리 프로세스에 대해서는 프로세스 단계별로 수행 내용을 이해하고 가세요!

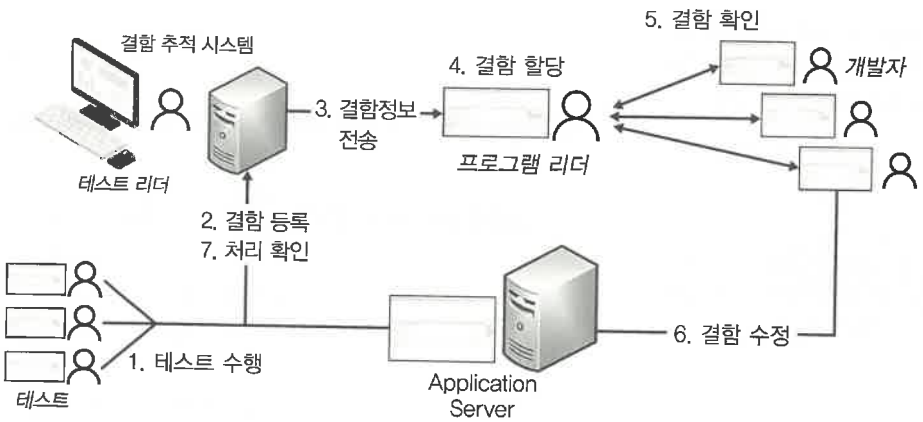
테스트 ID	테스트 시나리오명	테스트 케이스 ID	테스트 케이스명	결과
DOCTS-30-01	전자문서 안내와 전자문서작성 (업무사용자)	DOCTS-30-01-01	전자안내	PASS
		DOCTS-30-01-02	보낸주소#메일확인	FAIL
		DOCTS-30-01-03	받는주소등록	PASS
		DOCTS-30-01-04	첨부문서등록	PASS
		DOCTS-30-01-05	발송 요청	NOT EXECUTED
DOCTS-80-01	받은 문서함 관리 (업무 사용자)	DOCTS-80-01-01	검색 조건 등록	CLARIFIED
		DOCTS-80-01-02	문서 목록 조회	PASS
		DOCTS-80-01-03	문서 상세 조회	PASS

테스트 케이스 ID	테스트 담당자	결함 유형	결함 상태	결함 내용
DOCTS-30-01-01	테스터1		Close	
DOCTS-30-01-02	테스터1	개발	Open	보낸 주소로 메일이 전달되지 않음
DOCTS-30-01-03	테스터1		Close	
DOCTS-30-01-04	테스터1		Close	
DOCTS-30-01-05	테스터1	개발	Fiexed	발송 요청이 문서가 전달되지 않음
DOCTS-80-01-01	테스터3	분석	Clarified	문서 검색 시 특수문자 사용불가 (SQL 인젝션 모듈 작동)
DOCTS-80-01-02	테스터3		Close	
DOCTS-80-01-03	테스터3		Close	

▲ 결함 관리 현황 예시

2 결함에 대한 원인을 분석하고 개선 방안 도출

- 테스트 수행 시 발견된 결함에 대한 원인을 유형별로 통합하고 분류한다.
- 분류된 결함에 대하여 파레토 다이어그램, 피시본 다이어그램 등의 통계적 분석 기법을 적용하여 분석하고 개선 방안을 도출한다.
- 결함 추적 시스템을 통해 결함을 효율적으로 관리할 수 있다.



▲ 결함 추적 시스템 흐름도

잠깐! 알고가기

파레토 다이어그램(Pareto Diagram)
자료들이 어떤 범주에 속하는가를 나타내는 계수형 자료일 때 각 범주에 대한 빈도를 막대의 높이로 나타낸 도표이다.

피시본 다이어그램(Fishbone Chart)
다양한 요인과 잠재적 문제 및 결과 간 연결 가능성을 시각화해서 표현하는 도표이다.

CHAPTER

02 애플리케이션 결함 조치하기

1 결함 조치 우선순위 결정★★★

(1) 소프트웨어 테스트 기법

1 단위 테스트 기법

① JUnit을 활용한 테스트

Java 환경이라면 대부분 JUnit이라는 단위 테스트 프레임워크를 통해 단위 테스트를 할 수 있어야 한다.

② Mock 테스트

- 단위 테스트 시 Mock 객체를 사용하여 테스트하는 기법이다.
- Mock 테스트는 특정 기능 또는 모듈에 대한 응답 결과를 미리 정의해 놓고 테스트한다.
- 특정 모듈이나 기능이 완벽히 개발 완료되지 않은 상태에서도 진행할 수 있다.
- 테스트 전용 객체를 테스트 더블이라 부르며, 테스트를 위해 실제 객체를 대신해서 사용되는 용어이다.
- 객체의 유형은 아래와 같다.

▼ Mock 테스트

객체 유형	수행 내용
Dummy	• 객체의 전달에만 사용되고 실제로는 사용하지 않음 • 주로 매개 변수 목록을 채우는 데 쓰임
Fake	• 실제로 동작하도록 구현되지만, 보통 빠른 구현을 위해 실제 환경과는 다르게 구현
Stubs	• 테스트를 위해 미리 준비한 응답만을 제공하는 객체 • 그 외의 상황에 대해서는 정상적인 작동을 하지 못함 • 스텁은 호출에 대한 정보를 기록하는 경우도 있음
Mocks	• 스펙을 통해 정의된 응답을 받고 다른 응답을 받을 경우 예외를 발생하도록 구현 • 응답에 대한 확인을 수행

잠깐! 알고가기

프레임워크(Framework)
소프트웨어의 구체적인 부분에 해당하는 설계와 구현을 재사용이 가능하게끔 클래스들을 제공하는 틀이다.

두음쌤 한마디

Mock 테스트 객체 유형
「덤페스목」
Dummy / **Fa**ke / **S**tubs / **Mo**cks
→ 항상 덤덤한 페이커는 스타가 될 재목이다.

잠깐! 알고가기

매개 변수
어떠한 시스템이나 함수의 특정한 성질을 나타내는 변수이다.

2 통합 테스트 기법

- 전체 시스템이 통합 완료될 때까지 단위 시스템 간의 연계성 및 기능 요구사항들을 확인하고, 하드웨어와 소프트웨어 구성요소 간의 상호작용을 테스트하는 것이 주요 목적이다.
- 업무 간의 연계성과 상호 운영성 중심의 테스트를 수행한다.

학습 Point
통합 테스트 기법은 개념 위주로 학습하세요!

① 테스트 설계 기법

- 테스트 설계는 개발된 소프트웨어나 시스템의 요구사항, 요구사항 명세서, 업무 구조, 시스템 구조 등을 기반으로 소프트웨어의 어떤 부분을 어떻게 접근하여 테스트할지에 대한 테스트 상황과 방법을 파악하는 것이다.
- 이를 체계적으로 구체화시켜 테스트 케이스를 도출하고 작성하는 것을 테스트 구현이라고 한다.
- 테스트 상황과 방법을 구체화하기 위한 수단 및 도구를 테스트 설계 방법이라고 한다.

두음쌤 한마디

테스트 설계 방법
「케버대」
보다 작은 케이스 / 보다 많은 버그 찾기 / 테스트 대상을 빠짐없이 실행하기
→ 케임브리지 버클리 대학교

② 테스트 설계 방법

▼ 테스트 설계 방법

분류	수행 내용
보다 작은 케이스	동등 클래스(Equivalence Class) 예 한 자리 양의 정수를 더하는 프로그램인 경우 1+1, 1+2…9+9까지 81가지 케이스는 모두 동등한 경우이다. 따라서 이런 경우를 모두 모아서 몇 가지 케이스만 테스트한다.
보다 많은 버그 찾기	경계 테스트(Boundary Test) 예 위의 동등 클래스 중 대표 클래스를 뽑을 때 가장자리, 즉 경곗값을 뽑는 경우이다.
테스트 대상을 빠짐없이 실행하기	모델 기초(Model-based) 테스트 설계 예 상태 전이(State-transition) 테스트 모델을 UML의 State Chart Diagram으로 표현하여 노드와 링크를 모두 대상으로 하여 테스트한다. 또는 모든 테스트 요소를 매트릭스 형태로 나열하여 테스트한다.

잠깐! 알고가기
UML(Unified Modeling Language)
객체지향 소프트웨어 개발과정에서 산출물을 명세화, 시각화, 문서화할 시 사용되는 모델링 기술과 방법론을 통합해 만든 표준화된 범용 모델링 언어이다.

3 시스템 테스트 기법

시스템 테스트 업무 진행 전체를 총괄할 수 있도록 절차 및 프로세스별 세부 업무를 숙지해야 하고 결과에 대한 분석 및 해결 방안을 제시할 수 있어야 한다.

▼ 시스템 테스트 기법

유형	설명
부하 테스트	• 동시접속으로 시스템에 많은 요청(업데이트, 조회 등)이 발생할 때 어떻게 가동되는지 확인하는 테스트 • 시스템에 과다 정보량을 부과하여 과부하 시에도 시스템이 정상적으로 작동되는지 검증
성능 테스트	• 성능 요구사항에 정의된 임계점을 넘어 부하가 계속 증가하는 상황에서 시스템이 정상적인 수준의 반응을 보이는지 테스트 • 사용자 이벤트에 시스템이 응답하는 시간, 특정 시간 내에 처리하는 업무량, 사용자 요구에 시스템이 반응하는 속도 등을 측정
장애 복구 테스트	• 장애 복구 테스트는 각종 장애(하드웨어 장애, 네트워크, 정전, 운영 오류 등) 상황에서의 복구 및 재가동 테스트 • 애플리케이션의 장애 복구 테스트는 서비스의 지속적인 유지라는 측면에서 발생할 수 있는 장애 목록을 작성하고 장애 상황을 발생시켜서 테스트
보안 테스트	• 보안 테스트는 별도의 보안 전문가에 의해 애플리케이션 전반에 걸쳐 검증받아야 함 • 보안성 검증은 애플리케이션의 기능적 요구사항과는 별도로 해킹이나 침투에 대비할 수 있는 최신 기술 트렌드가 반영된 검증이 필요

학습 Point

시스템 테스트 기법 유형별 테스트 방식에 대한 정확한 이해와 암기가 필요해요!

두음쌤 한마디

시스템 테스트 기법

「부성장보」

부하 테스트 / 성능 테스트 / 장애 복구 테스트 / 보안 테스트

→ 부하가 성장해서 보호해줌

4 인수 테스트 기법

- 인수 테스트는 최종 사용자가 요구한 기능이 제대로 반영되었는지, 인수 조건에 만족하는지를 테스트하는 기법이다.
- 요구 기능 만족 여부, 사용 편리성에 대하여 실제 운영 환경에서 실행되며 고객이 주도하는 테스트이다.

(2) 결함 관리의 이해

1 결함 관련 용어

▼ 결함 관련 용어

결함 관련 용어	설명
에러(Error)	• 소프트웨어 개발 또는 유지보수 수행 중에 발생한 부정확한 결과 • 개발자의 실수로 발생한 오타, 개발 명세서의 잘못된 이해, 서브루틴의 기능 오해 등의 에러 존재
오류(Fault)	• 프로그램 코드상에 존재하는 것으로 비정상적인 프로그램과 정상적인 프로그램 버전 간의 차이로 인하여 발생
실패(Failure)	• 정상적인 프로그램과 비정상적인 프로그램의 실행 결과의 차이를 의미 • 프로그램 실행 중에 프로그램의 실제 실행 결과를 개발 명세서에 정의된 예상 결과와 비교함으로써 발견
결함(Defect)	• 버그, 에러, 오류, 실패, 프로그램 실행에 대한 문제점, 프로그램 개선 사항 등의 전체를 포괄하는 용어

학습 Point

결함 관련 용어에 대한 정확한 이해가 필요합니다.

두음쌤 한마디

결함 관련 용어

「에오실결」

에러 / 오류 / 실패 / 결함

→ LOL 에이전트 오류로 실검 결과 1위

2 결함의 판단 기준

기능 명세서를 기준으로 다음 5가지 규칙 중 하나 이상에 해당하면 결함이라 볼 수 있다.

▼ 결함의 판단 기준

구분	판단 기준
규칙 1	기능 명세서에 가능하다고 명시된 동작을 수행하지 않는 경우
규칙 2	기능 명세서에 불가능하다고 명시된 동작을 수행하는 경우
규칙 3	기능 명세서에 명시되어 있지 않은 동작을 수행하는 경우
규칙 4	기능 명세서에 명시되어 있지 않지만 수행해야 할 동작을 수행하지 않는 경우
규칙 5	테스터의 시각에서 볼 때 문제가 있다고 판단되는 경우

테스터의 시각에서 결함으로 판단되는 경우(예시)

이해하기 어려운 기능, 사용이 까다로운 기능, 비정상적으로 느린 기능 등

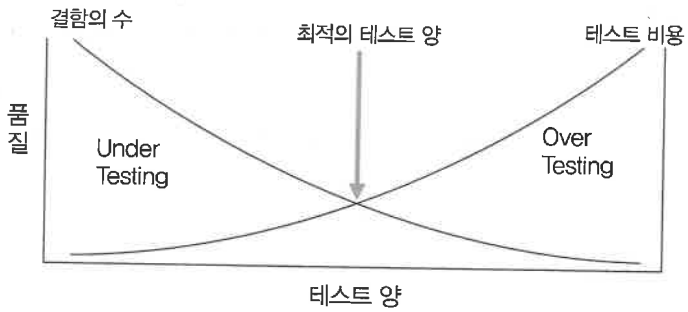
(3) 테스트 격언(Testing Axioms)

학습 Point

테스트 격언은 중요도는 낮습니다. 이해를 위해 가볍게 읽고 가는 정도로 봐주세요!

▼ 테스트 격언

테스트 격언	설명
소프트웨어를 완벽하게 테스트하는 것은 불가능	• 가능한 입력의 수가 너무 많음 • 가능한 출력의 수가 너무 많음 • 소프트웨어 명세서가 주관적임 • 소프트웨어 결함도 주관적임
소프트웨어 테스트는 위험을 수반하는 훈련	• 가능한 모든 테스트 시나리오와 테스트 케이스를 테스트하지 않을 경우 위험을 감수하기로 했다는 뜻 • 테스터는 무엇이 중요하고 무엇이 중요하지 않은지 현명한 판단을 하여야 함
테스트 작업으로 결함이 존재하지 않는다는 사실을 입증할 수는 없음	• 테스트를 통해 소프트웨어에 결함이 없다는 것을 보장할 수는 없으며, 단지 테스트 결과로 결함이 있다는 사실만을 보여줄 수 있음 • 테스트 결과 결함이 발견되지 않았다 하더라도 이는 결함이 없음을 의미하는 것은 아님
발견한 모든 결함을 수정할 수는 없음	• 쫓기는 작업 일정으로 인해 모든 결함 수정 불가 • 외부적인 요인(테스트 환경 미비, 사용자 환경 자체 장애, 법률 또는 규정 변경 등)으로 결함이 아님에도 결함으로 판단되는 경우 존재 • 각각의 프로그램 간에 서로 결합도가 높은 경우의 고치기 위험한 결함 존재 • 현실에서 거의 발생할 가능성이 없는 경우(자연재해)의 고칠 가치가 없는 결함 존재
발견한 결함과 남아 있는 결함의 수는 비례	• 발견한 결함이 많을수록 남아 있는 결함의 수도 많음



▲ 최적의 테스트 양

결함의 수 및 테스트 비용을 고려하여 최적의 테스트 양을 산정하는 것이 중요하다.

(4) 소프트웨어 테스터의 역할과 능력

▼ 테스터의 역할과 능력

구분	설명
소프트웨어 테스터의 역할	• 결함을 발견 역할 • 결함을 가능한 한 빨리 발견해야 하는 역할 • 가능한 한 결함을 빨리 발견하여 결함이 수정 보완되었는지 확인하는 역할
소프트웨어 테스트의 능력	• 탐구심과 문제 해결력을 갖추어야 함 • 때로는 창의력을 발휘하여 결함을 찾기 위해 새로운 접근법도 사용해야 함 • 완벽함을 추구하지만, 가능한 한도 내에서 적당한 수준의 완벽성을 추구해야 함 • 테스터는 항상 나쁜 소식을 전하는 사람으로 개발자에게 그들의 작품에 결함이 있다고 말하려면 재치와 능숙함, 그리고 설득력이 필요함

(5) 조치 우선순위 결정 프로세스

1 결함에 대한 개선 방안 확인

▼ 결함에 대한 개선 방안 확인

절차	설명
테스트 결함 목록과 개선 방안 확인	• 테스트 결함 목록 확인: 테스트 결함 목록 확인 시 테스트 시나리오를 확인하고 단위 테스트인지 통합 테스트인지 확인한 다음 분류 및 정렬 • 결함 목록의 세부 사항과 개선 방안 확인: 작성된 내용은 최종적으로 현업 담당자와 협의 후 개선 방안 적용 여부 결정

핵인싸 퀴즈

21 ( )은/는 접근제어 지시자 중 접근을 제한하지 않는다.

22 ( )은/는 객체들이 다 같이 공유하는 데이터를 의미한다.

23 ( )은/는 객체들의 데이터와 관계없이 완벽하게 공통적인 로직을 정의할 때 사용한다.

24 ( )은/는 자바에서 화면에 데이터를 출력하는 메서드로 줄 바꿈 없이 출력한다.

25 ( )은/는 프로그램 실행 시간 전에 변수와 변수에 관련된 속성을 연결하는 방식이다.

정답 21. public 22. static 변수 23. static 메서드 24. System.out.print 25. 정적 바인딩

핵인싸 퀴즈

26 ( )은/는 프로그램 실행 시간에 변수와 변수에 관련된 속성을 연결하는 방식이다.

27 ( )은/는 변수에 메모리 공간을 바인딩하는 작업이다.

28 ( )은/는 데이터 처리를 위해 연산을 표현하는 기호이다.

29 ( )은/는 조건이 참인지 거짓인지 판단하고자 할 때 사용하는 데이터 타입이다.

30 ( )은/는 4바이트로 보통 소수점 6자리까지 표현 가능한 데이터 타입이다.

31 ( )은/는 8바이트 소수점 15자리까지 표현 가능한 데이터 타입이다.

정답 26. 동적 바인딩 27. 할당 28. 연산자 29. 불린 타입 30. float 31. double

<table>
<tr><th>절차</th><th>설명</th></tr>
<tr><td>테스트 별로 결함 유형 분류</td><td>• 결함 내용을 파악하고 결함 유형 정의
[결함 유형 정의]
<table>
<tr><th>프로젝트 단계</th><th>테스트 단계</th><th>결함 유형</th></tr>
<tr><td>개발</td><td>단위</td><td>기능 관련 결함</td></tr>
<tr><td>분석</td><td>통합, 인수</td><td>프로세싱 결함</td></tr>
<tr><td rowspan="2">설계, 개발</td><td>단위, 통합</td><td>데이터 관련 결함</td></tr>
<tr><td>시스템</td><td>인터페이스 결함</td></tr>
<tr><td>분석</td><td>통합</td><td>표준화 결함</td></tr>
<tr><td>설계</td><td>시스템</td><td>소프트웨어 아키텍처 문제</td></tr>
<tr><td>테스트</td><td>테스트 전체</td><td>테스트 결함</td></tr>
<tr><td>분석, 설계, 개발</td><td rowspan="2">시스템, 인수</td><td>원인 파악 불가능 결함</td></tr>
<tr><td>기타</td><td>위 항목에 포함되지 않은 결함</td></tr>
</table>
• 결함 관리 대장을 확인하고 결함 유형 분류</td></tr>
<tr><td>결함의 개선 방안 범위 지정</td><td>• 개선할 테스트 케이스 ID 선정: 결함 관리 대장에서 발견된 결함에 대해 가장 접근 빈도가 높은 업무, 중요 업무 또는 비기능적으로 반드시 점검해야 할 부분을 개선 대상으로 선정하고 결함 검증 방법 제시
• 개선 조건과 범위 설정: 일반적으로 개선 사항을 검증하기 위해서는 개발 환경에서 진행하고, 시스템 개선인 경우에는 운영 서버에서 실제 환경을 구성한 후 검증하는 것이 필요</td></tr>
<tr><td>결함 유형별로 개선 효과 정의</td><td>• 프로젝트 성격에 따라 결함 유형별로 개선 효과는 다르게 정의되어 작성되며, 개발자 · 시스템 · 서비스 사용자 · 서비스 · 아키텍트 관점 등 테스트의 목표에 따라 개선 효과는 각각 다르게 정의
• 프로젝트의 전체적인 개선 효과는 각 결함 유형별 자료를 수집하여 종합 보고서로 작성</td></tr>
</table>

2 개선 방안에 대한 우선순위 결정

▼ 개선 방안에 대한 우선순위 결정

절차	설명
업무별 기준에 가중치 부여	• 업무/서비스 중요도를 파악하기 위해 이전 시스템의 접속 로그와 업무 일일 데이터 처리량을 통해 작성 • 시스템 의존도는 시스템 없이 수작업이 가능한 업무와 불가능한 업무로 나누어 작성 • 사용자 범위는 현재 사용 대상의 범위와 향후 변경되는 시스템의 확대 가능성을 판단하여 작성 • 월 이용자 수는 현재 전담 인력과 공통 성분을 포함하여 월 이용자 수를 산정하고 전체적인 평균치를 고려해서 작성
단위 업무와 연결하여 심각도 계산	• 단위 업무별 평가 등급 부여 • 단위 업무별 심각도 계산

절차	설명
부여된 가중치와 심각도를 고려하여 우선순위 결정	• 우선순위 결정: 정량적으로 나타난 결함 심각도를 바탕으로 결함 조치의 우선순위를 나열하며, 업무별 담당자와 팀별 회의를 통해 정성적으로 평가하고 구분(이때 업무별로 결함 심각도가 가장 높다고 해서 그 결함을 무조건 수정해야 하는 것은 아님) • 결함 우선순위를 결정하고 표현: 높음, 보통, 낮음으로 표현 순위를 나타내며, 각 우선순위에 따라 인력 배치와 결함 조치 일정을 결정

2 결함 조치 관리★

(1) 프로그램 코드 검토 기법

1 소프트웨어 인스펙션(Software Inspection)의 개념

- 소프트웨어 인스펙션은 설계서, 코드 등의 중간 산출물을 검사하여 결함을 발견하고 소프트웨어의 품질 개선과 비용 절감을 위한 품질보증기법이다.
- 소프트웨어 인스펙션은 코드 인스펙션 외에도 설계 및 설계 산출물까지 포괄하는 테스트 방법이다.
- 소프트웨어 인스펙션은 워크스루와 같이 몇 시간 동안 수행되는 단위 미팅과는 구별되며, 며칠 동안의 수행 기간이 필요하고 포함된 에러의 90%까지 찾아낼 수 있고, 도구의 도움이 있어야 한다.
- 적절한 인스펙션은 소프트웨어 개발의 전체 수명 주기에 걸친 리소스 절감과 그에 따른 비용 감소, 산출물의 품질 향상이 가능하다.

2 인스펙션 중점항목

▼ 인스펙션 중점항목

구분	설명
인스펙션 주관 및 지원	• 프로젝트에서 소프트웨어 인스펙션 업무는 품질 보증(QA) 주관 부서 및 담당이 수행 • 아키텍트는 각 단계에서 원활한 업무 수행을 위해 지원 필요 • 소프트웨어 인스펙션 중 특히 코드 인스펙션과 관련하여, 아키텍트는 코드 인스펙션 프로세스 전반과 각 단계별 수행 업무 등의 전체적 이해 필요
아키텍트 지원 업무	• 자동 코드 인스펙션을 위한 환경 지원, 계획 수립 지원 활동 • 체크 리스트 정합성 검토 지원 활동 • 인스펙션 결과 리뷰 참석 • 발견된 결함을 수정하기 위한 개발자 리딩 지원 활동

잠깐! 알고가기

품질 보증(QA)
고객의 요구사항과 개발된 산출물이 일치하는지 확인하기 위한 체계적인 행위이다.

학습 Point

코드 인스펙션 프로세스는 실기 시험에 나올 확률이 높습니다. 두음 기반으로 암기하세요!

두음쌤 한마디

코드 인스펙션 프로세스

「범시 준회 재후」

범위 계획 / 시작 / 준비 / 인스펙션 회의 / 재작업 / 후속 처리

학습 Point

코드 인스펙션 태스크별 수행 내용은 단계별 수행 내용을 이해하고 넘어가세요!

3 코드 인스펙션 프로세스와 수행 내용

① 코드 인스펙션 프로세스

▼ 코드 인스펙션 프로세스

구분	수행 단계	주요 내용
자동 수행	범위 계획	인스펙션의 범위와 범위 선정 기준 결정
	시작	자동 인스펙션 수행
준비 단계	준비	계획서 작성, 체크 리스트 작성, 계획 공지, 대상 산출물 준비
이행 단계	인스펙션 회의	사전 검토 실시, 미팅 실시
시정 조치	재작업	개발 원작자가 직접 작업
	후속 처리	결과 분석서 작성 및 보고

② 코드 인스펙션 태스크별 수행 내용

▼ 코드 인스펙션 태스크별 수행 내용

구분	수행 단계	주요 내용	산출물
자동 수행	자동 인스펙션 수행	• 전수 검사, Quality Metric, 결함 분석	코드 인스펙션 결과
준비 단계	계획서 작성	• 일정 및 관련자, 대상 산출물 및 준비물 정의	인스펙션 계획서
	체크 리스트 작성	• 표준 체크 리스트를 테일러링하여 인스펙션 체크 리스트 작성	인스펙션 체크 리스트
	계획 공지	• 메일이나 공지를 통해 관련자에게 사전 공지	
	대상 산출물 준비	• 산출물 작성자가 인스펙션 시 필요한 자료 준비	
이행 단계	착수 회의 실시	• 진행자는 참여자에게 검토 주안점, 검토 방법, 역할 등을 교육 • 작성자는 대상 산출물 및 참조 자료에 대한 개요 소개	
	사전 검토 실시	• 산출물, 체크 리스트, 사전검토서 양식 배포 • 참여자별로 자료를 개별 검토하여 발견된 부적합 사항 기록	인스펙션 결과서
	미팅 실시	• 사전 검토에서 발견한 부적합 사항 검증 • 미팅에서 발견된 부적합 사항 추가 기재 • 부적합 사항 목록 정리 또는 조치 계획 수립	인스펙션 결과서
	결과 정리	• 진행자는 최종 확정된 결함 내용을 인스펙션 결과서에 정리한 후 작성자에게 배포	
시정 조치	보완 작업 실시	• 작성자는 각 결함에 대한 보완 작업 실시	
	시정 조치 결과 확인	• 진행자는 보완 완료 여부 확인(필요시 재검토 실시)	
	결과 보고	• 진행자는 결과서를 작성하여 관련자에게 보고	인스펙션 결과서

③ 코드 인스펙션 유형별 적용 대상

▼ 코드 인스펙션 유형별 적용 대상

유형	적용 대상
자동 코드 인스펙션	• 전체 개발된 프로그램을 대상으로 자동 인스펙션 수행
수동 코드 인스펙션	• 자동 코드 인스펙션 코드 중 에러가 많은 경우 • 업무 중에 복잡한 처리 로직이 있는 경우 • 처음 투입되는 개발자의 산출물

④ 코드 인스펙션 수행 시 고려사항

- 인스펙션은 개발 가이드에 따른 표준 준수성을 파악하기 위한 목적이 있으므로 기능적으로 이상이 없는 소스 코드를 대상으로 검증한다.
- 인스펙션의 효과는 개발 가이드에 따른 체크 항목 파악, 결함 유형 파악에 따른 차후 코딩 시 유념, 다른 개발자의 기술 습득 등으로 다양하다.
- 인스펙션의 실제적 효과는 테스트 전 결함발견에 따른 이익을 수행 팀원들이 인식하는 것이다.

4 설계 인스펙션을 위한 체크사항

▼ 설계 인스펙션을 위한 체크사항

구분	체크사항
요구사항 측면	• 설계 내역들은 요구사항들로 추적 가능해야 함 • 설계의 외형 부분은 고객의 요구사항 분석과 연결되어야 하고 고객 프로파일 정보 포함 • 모든 요구사항의 제약조건들이 설계에 반영 • 설치 및 데이터 이동 요구사항이 설계에 반영 • 경곗값에 관련된 조건들을 식별하고, 관련된 로직을 분명하고 완전하게 정의해서 요구사항과 매핑시켜야 함
명확성 측면	• 모든 에러 메시지가 정의되어야 함 • 설계 내역은 명확하게 정의되어 오해가 생기지 않도록 함 • 하나 이상의 의미를 가진 용어들은 의도했던 의미가 무엇인지 분명하게 표현 • 모든 의존성이 명확히 표현되어야 함 • 컴포넌트와 모듈의 구조가 명확히 정의되어야 함 • 설계에 나타난 모든 알고리즘을 명확히 표현하여 코드를 작성할 때 혼란스럽지 않게 해야 함

핵인싸 퀴즈

32 ( )은/는 서로 연관된 정수형 상수들의 집합을 정의할 수 있는 사용자 정의 자료형이다.

33 ( )은/는 사용자가 기본 타입을 가지고 새롭게 정의할 수 있는 사용자 정의 자료형이다.

34 ( )은/는 모든 멤버 변수가 하나의 메모리 공간을 공유하는 사용자 정의 자료형이다.

35 ( )은/는 입력 자료를 출력 자료로 변환하는 과정을 추상화하는 방법이다.

정답 32. 열거체 33. 구조체 34. 공용체 35. 기능 추상화

구분	체크사항
설계 및 검증 측면	• 모든 설계는 최소 하나 이상의 테스트 활동에서 검증되어야 함 • 에러 복구 및 백업 요구사항이 모두 설계 내역에 반영되어 있어야 함 • (요구사항에 명시된) 성능 기준이 정제되어야 하고, 설계 내역이 그것을 만족하는지 보여야 함 • 안전이나 보안 등의 산업계 요구사항이 설계에 반영되어야 함 • 산업 표준 알고리즘을 사용하는 경우에, 의도했던 알고리즘을 정확히 반영하게 해야 하며, 어떤 알고리즘의 제약사항도 설계 요구사항을 만족시켜야 함 • 모든 에러 조건이 정의되어야 함 • 모듈과 컴포넌트의 명명법 표준이 준수되어야 함

5 인스펙션과 워크스루의 차이점

- 인스펙션은 워크스루와 달리 체크 리스트를 기반으로 검토하며 발견된 모든 결함을 제거해야 한다는 특징이 있다.
- 인스펙션은 완성도가 기준 이상일 때 수행함으로써 모든 결함을 없애는 데 주요 목적이 있다.

▼ 인스펙션과 워크스루 비교

구분	인스펙션	워크스루
목적	결함 파악 및 제거	산출물 평가 및 개선
수행 조건	완성도가 기준 이상일 때	팀이나 관리자의 필요 시
결함 수정 여부	모든 결함은 제거되어야 함	저자 결정
변경 사항 검증	진행자가 재작업 결과 확인	저자 결정
검토자 인원	3~6명	2~7명
참여자	동료	기술 전문가와 동료
검토 인도자	교육받은 진행자(Moderator)	저자
검토 준비 여부	체크 리스트를 이용한 검토	일반적으로 검토 준비하지 않음
검토 분량	상대적으로 많음	상대적으로 적음
검토 속도	상대적으로 느림	빠름
발표자	산출물에 의존도가 높은 사람(Reader)	저자
지표 수집 여부	모든 검토자가 기록함	하지 않음
보고자	결함 리스트 및 측정 지표	워크스루 보고서
데이터 측정 여부	필수	권장 사항
체크 리스트 사용 여부	사용함	사용하지 않음

핵인싸 퀴즈

36 ( )은/는 상위 수준 그룹의 모든 특성을 하위 수준 그룹이 이어받아 재사용 또는 확장하는 특성을 의미한다.

37 ( )은/는 컴퓨터에 저장된 명령어들이 순차적으로 실행되는 프로그래밍 방식으로 절차형 언어라고도 불린다.

38 ( )은/는 객체 간의 메시지 통신을 이용하여 프로그래밍하는 방식이다.

39 ( )은/는 수학적 수식과 같은 함수들로 프로그램을 구성하여 호출하는 방식이다.

정답 36. 상속 37. 명령형 언어 38. 객체지향 언어 39. 함수형 언어

(2) 프로그램 코드 검토 기법

1 소프트웨어 형상 관리 정의

- 소프트웨어 형상 관리 대상은 컴퓨터 프로그램, 컴퓨터 프로그램 설명 문서, 데이터 등 소프트웨어 프로세스의 모든 출력물 정보이다.
- 소프트웨어 프로세스 전반에 걸쳐 소프트웨어 형상의 변경 요인에 대한 관리를 통해 소프트웨어 형상을 보호하는 활동이다.

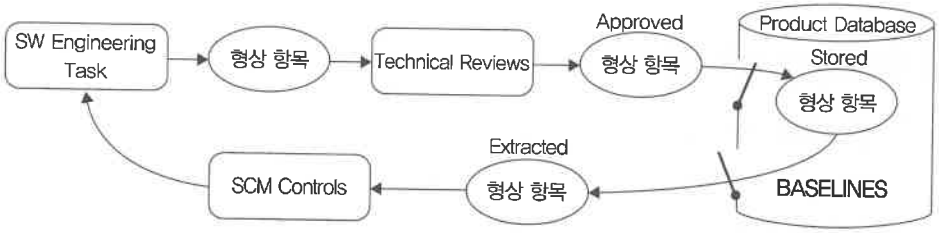
▼ 소프트웨어 형상 관리와 소프트웨어 지원 비교

소프트웨어 형상 관리	소프트웨어 지원
소프트웨어 엔지니어링 프로젝트 개시에서 소프트웨어 소멸 시점까지의 활동	소프트웨어가 고객에게 인도되고 운영되는 시점에 발생하는 소프트웨어 엔지니어링 활동

2 기준선과 소프트웨어 형상 항목

① 기준선(Baseline)

- 기준선은 시스템 생명 주기(SDLC, Software Development Life Cycle)의 일정 시점마다, 산출물을 검토하고, 그 결과를 반영하여 다음 단계로 이전할 때의 시점과 산출물이다.
- 변경을 통제하게 도와주는 기준선은 정식으로 검토 및 합의된 명세서나 제품 개발의 바탕으로서, 정식의 변경 통제 절차를 통해서만 변경할 수 있다.



▲ 기준선 개념도

- 형상 관리에서 베이스라인(Baseline)은 Software 개발 · 유지보수 산출물인 형상 항목에 대한 단계별 기준선이다.

② 소프트웨어 형상 항목(SCI; Software Configuration Item)

- 소프트웨어 형상과 개발 도구의 합성으로서, SCI는 아래와 같이 개발 단계별로 기준선을 기준으로 형상 항목을 관리한다.

학습 Point
개발단계별 기준선은 두음 기반으로 암기하세요!

두음쌤 한마디

소프트웨어 형상 관리 기준선

「요분 설시 제운」

사용자 요구사항 / 사용자 요구 기능이 하위 시스템 간에 어떻게 분배되는가 여부 / 개발 전 설계 명세 / 시험계획서 / 제품 / 운영

▼ 소프트웨어 형상 항목 예시

개발 단계	기준선(Baseline)	소프트웨어 형상 항목	기준선 설명
계획	사용자 요구사항	시스템 명세서, 개발계획서, 구성관리계획서, 품질평가계획서, 개발 표준 및 절차 매뉴얼	사용자 요구사항 명세서 또는 시스템 기능 요구 정의서를 검토하는 시점
요구 분석	사용자 요구 기능이 하위 시스템 간에 어떻게 분배되는가 여부	자료흐름도, 자료 사전, 자료흐름도 명세서	사용자 요구 기능들이 하위 시스템들 사이에 어떻게 분배되는가를 정의하는 기본 설계 명세서를 검토하는 시점
설계	개발 전 설계 명세	입출력명세서, 화면설계서, 초기 사용자 매뉴얼, 초기 시스템 매뉴얼, 자료 구조도, 시스템 구조도	프로그래밍하기 위한 설계 명세서를 검토하는 시점
구현	시험계획서	원시 코드, 목적 코드, 실행 코드, 단위시험 보고서	소프트웨어 기능 수행과 성능 충족도를 평가할 수 있는 원시 코드, 실행 코드 및 시험 사례를 포함한 시험 계획서를 검토하는 시점
시스템 통합 및 시험	제품	통합시험 보고서, 기능/성능/과부하 시험 보고서, 인증시험 보고서	하나의 시스템으로 개발 완료된 제품과 그의 품질을 보증하는 시점
설치 및 운영	운영	목적/실행 코드, 운영자 매뉴얼, 사용자 매뉴얼	사용자 환경에 설치되고 운용되기 시작한 소프트웨어나 그 품질을 사용자 입장에서 평가하는 시점

3 형상 관리의 주요 활동

① 형상 관리 기능

- 형상 관리의 주요 기능은 형상을 식별하고 관리하는 데 있다.
- 아래와 같이 형상 식별, 버전 관리, 변경 통제, 형상 감사, 상태 보고 등의 활동을 말한다.

학습 Point

단계별 형상 관리 기능은 두음 기반으로 암기해서 시험에 대비하세요!

▼ 단계별 형상 관리 기능

기능	설명
형상 식별	형상 관리 대상을 구분하고 관리 목록 번호 부여
버전 관리	진화 그래프 등을 통해 SCI의 버전 부여/갱신
변경 통제	소프트웨어 형상 항목에 대한 접근 및 동기화 제어
형상 감사	소프트웨어 형상 항목 무결성을 평가하여 공식적으로 승인
상태 보고	개발자와 유지보수자에게 변경 사항을 공지

② 형상 식별

• 소프트웨어 형상 항목에 대해 식별하고 고유한 이름을 부여하는 활동으로, 아래와 같이 식별자 항목을 정의하고 세부 내용을 파악하여 식별한다.

▼ 형상 식별

식별자 항목	내용
이름	객체 명
서술	소프트웨어 형상 항목 타입, 프로젝트 식별자, 버전 정보
자원	제공/처리/참조/요구되는 객체
실현	기본 객체인 경우 단위 텍스트에 대한 포인터 실현

두음쌤 한마디

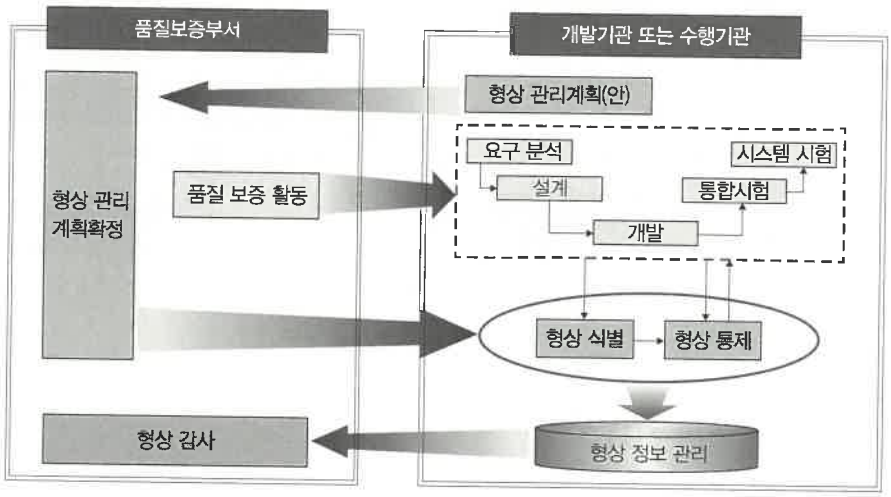
형상 관리 기능

「식버 변감상」

형상 식별 / 버전 관리 / 변경 통제 / 형상 감사 / 상태 보고

③ 버전 관리

• 형상 관리 활동 중 버전 관리는 여러 버전의 형상 객체를 관리하기 위한 절차와 도구, 그리고 형상 항목의 여러 버전을 표현하는 기능을 아래 그림과 같이 구성할 수 있다.



▲ 버전 관리 구성도

• 소프트웨어 버전 관리 도구는 아래와 같다.

▼ 소프트웨어 버전 관리 도구 유형

유형	설명
공유 폴더 방식 (RCS, SCCS)	• 매일 개발 완료 파일은 약속된 위치의 공유 폴더에 복사하는 방식 • 담당자 한 명이 매일 공유 폴더의 파일을 자기 PC로 복사하고 컴파일하여 에러 확인과 정상 동작 여부 확인 • 정상 동작일 경우 다음날 각 개발자가 동작 여부 확인
클라이언트/서버 방식 (CVS, SVN)	• 중앙에 버전 관리 시스템을 항시 동작시킴 • 개발자들의 현재 작업내용과 이전 작업내용 축적에 용이 • 서로 다른 개발자가 같은 파일을 작업했을 때 경고 메시지 출력

학습 Point

소프트웨어 버전 관리 도구 유형과 유형별 도구에 대해서는 시험 출제 가능성이 큽니다. 정확한 암기로 시험에 대비하세요!

두음쌤 한마디

소프트웨어버전 관리 도구 유형

「공클분」

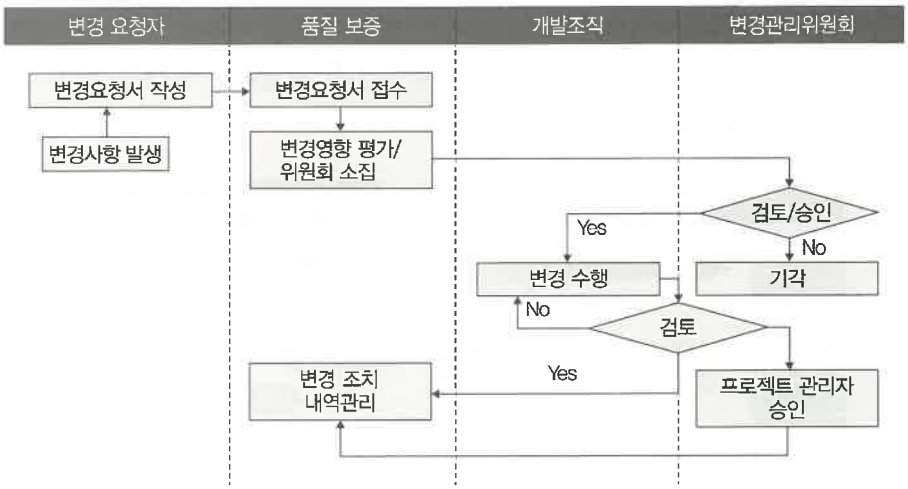
공유 폴더 방식 / 클라이언트/서버 방식 / 분산 저장소 방식

→ 공격 위주의 클럽이 분데스리가에 많다.

유형	설명
분산 저장소 방식 (Git, Bitkeeper 등)	• 로컬 저장소와 원격저장소로 분리된 구조 • 중앙의 저장소에서 로컬에 복사(clone)한 순간 개발자 자신만의 로컬 저장소에 생성 • 개발 완료한 파일을 수정한 다음에 로컬 저장소에 우선적으로 커밋(Commit)한 이후, 다시 원격저장소에 반영(Push)하는 방식

4 변경 통제에 대한 업무별 활동

변경 통제에 대한 업무별 주요 활동은 변경 요청에 대하여 프로세스에 어떠한 영향이 있는지 확인하는 과정으로 검토와 승인이 완료된 후 결함 변경을 수행한다.



▲ 버전 통제 흐름도

핵인싸 퀴즈

40 ( )은/는 현실 세계에서 개체를 데이터 속성과 메서드를 결합한 형태로 표현한 것을 의미하며 개체, 속성, 메서드로 구성된다.

41 ( )은/는 객체지향 프로그래밍에서 객체를 표현하는 추상 데이터 타입으로 객체를 생성하는 틀이다.

42 ( )은/는 프로그래밍 언어가 기본적으로 가지고 있는 라이브러리를 의미한다.

정답 40. 객체 41. 클래스 42. 표준 라이브러리

(3) 결함 조치 관리 절차

1 우선순위에 따른 결함 제거

① 결함 조치 일정 수립

▼ 결함 조치 일정 수립

구분	설명
결함 관리 대장에서 우선순위 확인	• 우선순위 확인 시 조치자의 우선순위가 같은 경우에는 2명 이상의 개발자 또는 디자이너, 아키텍트가 필요한 경우를 우선
결함 조치 일정 결정	• 결함 조치 일정을 결정할 때 사전에 테스터와 일정을 협의하는 것이 중요 • 테스터의 일정이 겹치더라도 우선순위에 따라 결함을 조치하는 것이 필요 • 심각한 결함의 경우에는 테스트 일정을 조정하고 결함 조치를 수행

테스트 케이스 ID	시나리오명	우선 순위	테스터	실시일	조치자	결함 유형
IT_F_01	일본어 반영 테스트	H	테스터1	20.1.10	개발자1	기능 관련
IT_F_02	Framework에 대한 전반적인 점검	L	테스터1	20.1.12	아키텍처1	소프트웨어 아키텍처
UT_S_01	기준정보 전체 구조 점검	M	테스터2	20.1.16	디자이너1 개발자2	표준화 결함
UT_S_02	공통 코드 관리	H	테스터3	20.1.18	개발자3	인터페이스 결함

우선순위를 결정하고, 실시일자를 결정한다.

▲ 우선순위의 결정과 일정 수립에 따른 결함 관리 대장 예시

② 결함 가이드라인 확인

▼ 결함 가이드라인 확인

구분	가이드라인
환경(Environment) 결함 가이드라인 확인	• 컴파일, 테스트, 또는 다른 지원 시스템 문제 • Configuration, 타이밍, 메모리 오류
개발 표준(Standard) 결함 가이드라인 확인	• 용어사전, 응용 프로그램 코딩 가이드, 성능 가이드, 응용 프로그램 명명 가이드 등 • 철자 오류, format, 스펠링, 구두점 등 • 버튼, 레이어명, 메시지, UI 등이 표준 미 준수
프로그램 일관성(Consistency) 결함 가이드라인 확인	• 구현되지 않은 누락된 기능 • 모듈 구조가 설계 구조와 미 일치(코딩 문제) • 분석, 설계 산출물(Input 산출물) 오류로 인한 코딩 에러
데이터(Data) 결함 가이드라인 확인	• 부적절한 데이터 변수 선언, 정의, 초기화, 중복 • 시험 데이터의 부적절로 인한 오류 • SQL을 사용한 DB 질의/갱신 오류 • 연계 화면 간의 데이터 전달 오류
조건 상태(Condition)와 연산(Calculation) 결함 가이드라인 확인	• 조건연산, 연산 오류 • Logic, Pointers, Loops, Recursion, Computation, 기능상의 오류 • 기능 결과가 예상 결과와 맞지 않음 • 버튼 동작 및 연계 화면 오류
기능 체크와 예외 처리 결함 가이드라인 확인	• 에러 메시지, 부적절한 기능 체크 • 예외 처리(Exception Handling) 오류

③ 결함 제거 유형

▼ 결함 제거 유형

번호	결함 제거 유형
1	프로젝트 표준/공통으로 정의한 개발표준을 따르지 않은 경우(명명 규칙, 변수 선언, 문장, 소스 들여쓰기, 주석 등)에 대한 결함 제거
2	언어 문법을 따르지 않은 경우(데이터 타입 선언, IF 조건문, WHILE 반복문, 연산자 사용 등)에 대한 결함 제거

학습 Point

결함 가이드라인 확인은 이해 기반으로 가볍게 보세요!

번호	결함 제거 유형
3	구문(Syntax)상에는 문제가 없으나 성능 저하(메모리 또는 클래스 할당과 해제, 반복문 내의 변수 선언 등)에 대한 결함 제거
4	개발표준 구조를 따르지 않거나 코드의 가독성이 좋지 않은 경우(성능이 동일할 경우 불필요한 로직 구현으로 가독성이 떨어지는 등)에 대한 결함 제거
5	입력과 출력 오류, 공통 클래스 또는 모듈(DB 처리, 로그 처리, 전역변수 등)의 공통적인 오류와 비즈니스 로직 구현에 오류가 있어 기본 요건을 불만족하는 경우에 대한 결함 제거
6	그 외 분류가 어려운 항목에 대한 결함 제거

2 결함 조치 후 변경 사항에 대한 형상 관리

▼ 결함 조치 후 변경 사항에 대한 형상 관리

구분	형상 관리
변경되는 소스의 버전 관리	• 형상 항목 식별 및 기술 • 형상 항목의 식별자 정의 • 버전 관리 기준과 기준선(Baseline) 정의
결함 조치 결과 이력 관리	• 변경된 결함에 대한 개선 사항을 변경 처리 절차에 따라 수행하고, 정확히 기록하였는지를 검토하고 형상 통제 • 승인된 결함에 대해 변경을 수행하고 종료
변경된 결함의 형상 상태 기록	• 형상 상태를 기록하고 보고하는 일정 수립 • 현재 소프트웨어 요구사항 명세서를 기준으로 구조 설계서, 상세 설계서, 사용자 매뉴얼 등이 잘 관리되고 있는지 기록
형상 감사 계획을 수립하고 기록	• 감사 대상 항목과 기간 수립 • 형상 관리에 대한 대상과 항목 그리고 결과물이 명확하게 기록되어 있는지 확인 • 감사에 대한 결과물과 자료를 기록하고 감사를 종료

(4) 형상 관리 도구

1 형상 관리 도구의 개념

- 소프트웨어 변경 사항을 관리하기 위해서 형상 식별, 통제, 감사, 기록을 수행하는 도구이다.
- 소스 코드나 문서의 버전 관리, 이력 관리, 추적 등 변경 사항을 체계적으로 관리할 수 있는 기능을 제공하는 도구이다.
- 대표 도구로 CVS, SVN, Git가 있다.

2 형상 관리 도구의 기능

▼ 형상 관리 도구의 기능

기능	설명
체크아웃(Check-out)	형상 관리 저장소로부터 최신 버전을 개발자 PC로 다운로드하는 기능
체크인(Check-in)	개발자가 수정한 소스를 형상 관리 저장소로 업로드하는 기능
커밋(Commit)	개발자가 소스를 형상 관리 저장소에 업로드 후 최종적으로 업데이트가 되었을 때 형상 관리 서버에서 반영하도록 하는 기능

두음쌤 한마디

형상 관리 도구 기능

「인아커」

체크인 / 체크아웃 / 커밋

3 형상 관리 도구 사례

① CVS(Concurrent Versions System)

- 가장 오래된 형상 관리 도구 중의 하나로서 중앙 집중형 서버 저장소를 두고 클라이언트가 접속해서 버전 관리를 실행한다.
- 주요 기능으로 check out/check in, update/commit 등 서버 저장소와 클라이언트의 변경 사항 전송, diff를 통한 파일 내용 비교, 파일 단위의 변경 사항 관리이다.

② SVN(Subversion)

- CVS와 같은 중앙 집중형 클라이언트-서버 방식이나, CVS의 단점을 보완해 가장 널리 사용되고 있는 형상 관리 도구이다.
- SVN은 CVS를 대체하기 위해 만든 도구로, 사용법이 CVS와 유사하여 CVS 사용자가 어렵지 않게 SVN을 도입하여 사용할 수 있다.
- 빠른 속도, 저장 공간 절약, 작업 모음 단위 기반으로 동작 등이 개선되었다.

③ Git

- 리누스 토발즈가 2005년 리눅스 커널의 개발을 위해 만든 형상 관리 시스템이다.
- Git는 중앙 집중형 방식이 아닌 분산형 방식으로 각 PC 스스로 완전한 저장소가 구성되며, 필요에 따라 중앙 집중형 방식으로도 운영할 수 있다.

두음쌤 한마디

형상 관리 도구 사례

「CSG」

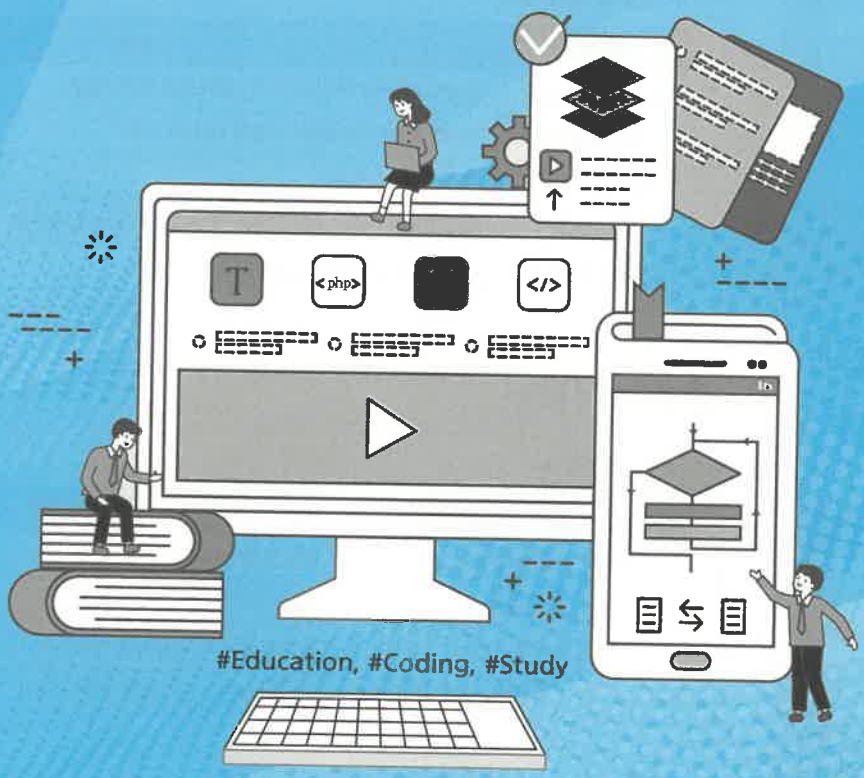
CVS / SVN / Git

→ LOL은 CS GAME이다.

Ⅳ SQL 활용

Chapter 01 기본 SQL 작성하기

Chapter 02 고급 SQL 작성하기



CHAPTER

01 기본 SQL 작성하기

1 데이터 정의어(DDL)★★★

(1) 데이터 정의어(DDL; Data Definition Language)의 개념

- 데이터 정의어는 데이터를 정의하는 언어로서 데이터를 담는 그릇을 정의하는 언어이다.
- 테이블과 같은 데이터 구조를 정의하는 데 사용되는 명령어들로 특정 구조를 생성, 변경, 삭제, 이름을 바꾸는 데이터 구조와 관련된 명령어들을 데이터 정의어라고 부른다.

두음쌤 한마디

DDL의 대상 오브젝트
「도스테뷰인」
도메인 / 스키마 / 테이블 / 뷰 / 인덱스
→ 18세기 독일에 도스테 부인(뷰인)이라는 사람이 있었다.

잠깐! 알고가기

객체(Object)
저장 공간을 할당받거나 식별자(오브젝트명)에 의해 참조되는 공간으로서 DB에서는 테이블, 인덱스 등을 일컫는다.

(2) DDL의 대상 객체

DDL을 사용할 수 있는 객체로는 도메인, 스키마, 테이블, 뷰, 인덱스가 있다.

▼ DDL의 대상 객체

DDL 대상	설명
도메인(Domain)	속성의 데이터 타입과 크기, 제약조건 등의 정보
스키마(Schema)	DBMS 특성과 구현을 고려한 데이터 구조
테이블(Table)	데이터 저장 공간
뷰(View)	하나 이상의 물리 테이블에서 유도되는 가상의 테이블
인덱스(Index)	검색을 빠르게 하기 위한 데이터 구조

(3) DDL 명령어

두음쌤 한마디

DDL 명령어
「크알드트」
CREATE / ALTER / DROP / TRUNCATE
→ 큰 알 두 트럭

▼ DDL 명령어

구분	DDL 명령어	설명
생성	CREATE	데이터베이스 오브젝트 생성
수정	ALTER	데이터베이스 오브젝트 변경
삭제	DROP	데이터베이스 오브젝트 삭제
	TRUNCATE	데이터베이스 오브젝트 내용 삭제

(4) DDL 활용

▼ DDL 관련 SQL문법

구분	문법

테이블 생성	CREATE TABLE 테이블명
	(
	  속성명 데이터 타입 [NOT NULL],
	  …
	  PRIMARY KEY(기본키),
	  UNIQUE(속성명, …),
	  FOREIGN KEY(외래키) REFERENCES 참조테이블(기본키),
	  CONSTRAINT 제약조건명 CHECK(조건식)
	);
	
	예
	CREATE TABLE 사원
	(
	  이름 VARCHAR(10) NOT NULL,
	  사번 VARCHAR(10) NOT NULL,
	  생년월일 VARCHAR(8),
	  입사일 DATE,
	  PRIMARY KEY(사번),
	  FOREIGN KEY(업무) REFERENCES 부서(부서코드),
	  CONSTRAINT 나이제한 CHECK(생년월일 < '19800101')
	);

열 추가	ALTER TABLE 테이블명 ADD 속성명 데이터 타입 [DEFAULT 값];
열 타입 변경	ALTER TABLE 테이블명 MODIFY 속성명 데이터 타입 [DEFAULT 값];
테이블 삭제	DROP TABLE 테이블명;
테이블 내용 삭제	TRUNCATE TABLE 테이블명;

(5) 제약조건 적용

- 테이블 생성 시 부적절한 자료가 입력되는 것을 방지하기 위해 사용된다.
- 제약조건은 종속성이 존재할 경우 테이블 삭제 등을 방지한다.
- 테이블의 행이 삽입, 갱신, 삭제될 때마다 테이블에서 규칙을 적용한다.

▼ 테이블 생성 시 사용되는 제약조건

제약 조건	설명
PRIMARY KEY	• 테이블의 기본키를 정의 • 유일하게 테이블의 각 행을 식별
FOREIGN KEY	• 외래키를 정의 • 참조대상을 테이블(컬럼명)로 명시 • 열과 참고된 테이블의 열 사이의 외래키 관계를 적용하고 설정

핵인싸 퀴즈

1 ( )(이)란 개발된 응용 애플리케이션이나 시스템이 사용자가 요구하는 기능과 성능, 사용성, 안정성 등을 만족하는지 확인하고, 노출되지 않은 숨어있는 소프트웨어의 결함을 찾아내는 활동이다.

2 테스트는 오류 발견, 오류 ( ), 품질 향상 측면에서 필요하다.

3 ( )(이)란 소프트웨어 요구, 설계, 원시 코드 등의 저작자 외의 다른 전문가 또는 팀이 검사하여 오류를 찾아내는 공식적 검토 방법이다.

정답 1. 테스트 2. 예방 3. 인스펙션(Inspection)

제약 조건	설명
UNIQUE	• 테이블 내에서 얻은 유일한 값을 가져야 함 • 테이블 내에서 같은 값을 가져서는 안 되는 항목 지정
NOT NULL	• 해당 컬럼은 NULL 값을 포함하지 않음을 지정
CHECK	• 개발자가 정의하는 제약조건 • 참(TRUE)이어야 하는 조건을 지정

테이블 생성을 위한 CREATE문에 제약조건을 명시하는 형태로 사용되며, ALTER를 통해 테이블 제약조건의 변경이 가능하다.

2 데이터 조작어(DML)★★★

(1) 데이터 조작어(DML; Data Manipulation Language)의 개념

데이터베이스에 저장된 자료들을 입력, 수정, 삭제, 조회하는 언어이다.

(2) 데이터 조작어(DML)의 유형

데이터 조작어(DML)의 유형에는 SELECT, INSERT, DELETE, UPDATE가 있다.

▼ DML의 유형

유형	동작	설명
SELECT	데이터 조회	해당 테이블을 구성하는 튜플 중에서 전체 또는 조건을 만족하는 튜플을 검색하는 명령문
INSERT	데이터 삽입	해당 테이블에 새로운 튜플을 삽입할 때 사용하는 명령문
UPDATE	데이터 변경	해당 테이블에 있는 튜플 중에서 특정 튜플의 내용을 변경할 때 사용하는 명령문
DELETE	데이터 삭제	해당 테이블에 있는 튜플 중에서 특정 튜플을 삭제할 때 사용하는 명령문

잠깐! 알고가기

관계형 데이터베이스에서는 행(Row)을 레코드(Record) 또는 튜플(Tuple)로 불린다.

두음쌤 한마디

DML의 유형

「세인업데」

SELECT / **IN**SERT / **UP**DATE / **DE**LETE

→ 내 친구 세인이 집에 업데

(3) 데이터 조작어(DML) 명령문

1 SELECT(데이터 조회) 명령문

① SELECT 명령문 개념

데이터의 내용을 조회할 때 사용하는 명령어이다.

```
SELECT [옵션] 컬럼명1, 컬럼명2, …
  FROM 테이블명1, …
[WHERE 조건]
[GROUP BY 컬럼명1, 컬럼명2, …]
[HAVING 그룹조건]
[ORDER BY 속성 [ASC | DESC] ];
```

▼ SELECT 명령문

항목	설명	
SELECT 절	• 검색하고자 하는 컬럼명, 계산식 지정 • 2개 이상의 테이블을 대상으로 검색할 때는 '테이블명.속성명'으로 표현 • 옵션을 둘 수 있는데, 옵션이 없으면 ALL이 기본값	
	ALL	중복을 포함한 조회 결과
	DISTINCT	중복을 제거한 조회 결과
FROM 절	• 질의에 의해 검색될 데이터들을 포함하는 테이블명을 기술	
WHERE 절	• 검색할 조건을 기술	
GROUP BY 절	• 속성값을 그룹으로 분류하고자 할 때 사용	
HAVING 절	• GROUP BY에 의해 분류한 후 그룹에 대한 조건 지정	
ORDER BY 절	• 속성값을 정렬하고자 할 때 사용 • ASC, DESC 키워드 생략 시 오름차순 정렬	
	ASC	오름차순 정렬
	DESC	내림차순 정렬

② SELECT 명령어 사례

학번	성명	학년	수강과목
1001	홍길동	1	수학
1004	임꺽정	2	계산이론
1005	장길산	3	수학

〈학생 테이블〉

'학생' 테이블에서 사람들이 수강하고 있는 과목들을 조회

```
-- DISTINCT 옵션이 없을 경우(수학이 중복되지만 모두 출력됨)
SELECT 수강과목
  FROM 학생;
 → 결괏값: 수학, 계산이론, 수학
```

```
-- DISTINCT 옵션이 있을 경우(수학이 중복되어 하나만 출력됨)
SELECT DISTINCT 수강과목
  FROM 학생;
 → 결괏값: 수학, 계산이론
```

학습 Point

SELECT 명령문의 형식과 SQL 문장 작성 방법에 대한 학습을 권장합니다.

두음쌤 한마디

SELECT 명령문

「셀프 웨 구해오」

SELECT / **FR**OM / **WHE**RE / **GRO**UP BY / **HA**VING / **OR**DER BY

→ 셀프 웨이터를 구해오라

핵인싸 퀴즈

4 동일한 테스트 케이스에 의한 반복적 테스트는 새로운 버그를 찾지 못한다는 소프트웨어 테스트 원리를 ( )(이)라고 한다.

5 요구사항을 충족시켜주지 못한다면, 결함이 없다고 해도 품질이 높다고 볼 수 없다는 소프트웨어 테스트 원리를 ( )(이)라고 한다.

6 적은 수의 모듈에서 대다수의 결함이 발견되고, 20%의 모듈에서 80%의 결함이 발견된다는 소프트웨어 테스트 원리를 ( )(이)라고 한다.

정답 4. 살충제 패러독스 5. 오류-부재의 궤변 6. 결함 집중

'학생' 테이블에서 2학년 이상 1인 학생들을 조회

```
-- WHERE 절을 이용해서 조회 조건을 추가할 수 있음
SELECT 성명
  FROM 학생
 WHERE 학년 >= 2;
 → 결괏값: 임꺽정, 장길산
```

'학생' 테이블에서 2학년 이상이면서 수학을 수강하는 학생들을 조회

```
-- WHERE 절에 AND, OR를 이용해서 상세한 조회 조건을 추가할 수 있음
SELECT 성명
  FROM 학생
 WHERE 학년 >= 2 AND 수강과목 = '수학';
 → 결괏값: 장길산
```

'학생' 테이블에서 학생들의 성명과 수강과목을 출력하고 성명을 기준으로 가나다순으로 정렬

```
-- ORDER BY 절을 이용해서 정렬 가능
SELECT 성명, 수강과목
  FROM 학생
 ORDER BY 성명 ASC;
 → 결괏값: (임꺽정, 계산이론), (장길산, 수학), (홍길동, 수학)
```

2 INSERT(데이터 삽입) 명령문

① INSERT 명령문 개념

- 테이블에 새로운 데이터를 삽입할 때 사용하는 명령어이다.
- 속성명의 포함 여부에 따라 두 가지 형태의 명령문 형식을 제공한다.

▼ INSERT 명령문

형식	구문	설명
속성명 포함	INSERT INTO 테이블명(속성명1, …) VALUES(데이터1, …);	속성과 데이터 개수, 데이터 타입이 일치해야 함
속성명 미포함	INSERT INTO 테이블명 VALUES(데이터1, …);	테이블에 해당되는 컬럼의 개수와 데이터의 개수가 일치해야 함

② INSERT 명령어 사례

학번	성명	학년	수강과목
1001	홍길동	1	수학
1004	임꺽정	2	계산이론

〈학생 테이블〉

'학생' 테이블에 학번이 1005, 성명 '장길산', 학년이 3학년, 수강과목은 '수학' 인 학생을 삽입하는 경우

```
-- 속성명을 포함하는 INSERT문
INSERT INTO 학생(학번, 성명, 학년, 수강과목)
    VALUES(1005, '장길산', 3, '수학');
```

```
-- 속성명을 포함하지 않는 INSERT문
INSERT INTO 학생
    VALUES(1005, '장길산', 3, '수학');
```

3 UPDATE(데이터 변경) 명령문

① UPDATE 명령문 개념

데이터의 내용을 변경할 때 사용하는 명령어이다.

▼ UPDATE 명령문

구문	설명
UPDATE 테이블명 SET 속성명 = 데이터, … [WHERE 조건];	• UPDATE 명령문은 WHERE 절을 통해 어떤 조건이 만족하는 경우에만 특정 컬럼의 값을 수정하는 용도로 자주 사용됨 • WHERE 절이 없는 경우 테이블 전체에 대해 데이터가 변경됨

두음쌤 한마디

UPDATE 명령문

「업셋웨」

UPDATE / **SET** / **WHE**RE

잠깐! 알고가기

테이블 변경(ALTER)과 데이터 변경(UPDATE)과의 관계

테이블 변경(ALTER)은 테이블의 스키마(테이블 정의)를 변경하는 데이터 정의어(DDL)이고, 데이터 변경(UPDATE)은 테이블 내의 인스턴스(값)를 변경하는 데이터 조작어(DML)이다.

② UPDATE 명령어 사례

학번	성명	학년	수강과목
1001	홍길동	1	수학
1004	임꺽정	2	계산이론

〈학생 테이블〉

'학생' 테이블에서 학번이 1004인 학생의 수강과목 '계산이론'을 '이산수학'으로 변경하는 경우

```
-- WHERE 절에 해당하는 대상인 학번이 1004인 데이터들만 변경을 수행
UPDATE 학생
   SET 수강과목 = '이산수학'
 WHERE 학번 = '1004';
```

'학생' 테이블에서 모든 학생의 학년을 1씩 증가시키는 경우

```
-- 모든 행을 변경할 경우 WHERE 절을 포함하지 않음
UPDATE 학생
   SET 학년 = 학년+1;
```

4 DELETE(데이터 삭제) 명령문

① DELETE 명령문 개념

데이터의 내용을 삭제할 때 사용하는 명령어이다.

두음쌤 한마디

DELETE 명령문

「델프웨」

DELETE / **FR**OM / **WH**ERE

잠깐! 알고가기

테이블 삭제(DROP)와 데이터 삭제(DELETE)와의 관계

테이블 삭제(DROP)는 테이블의 스키마(테이블 정의)를 삭제하는 데이터 정의어(DDL)이고, 데이터 삭제(DELETE)는 테이블 내의 인스턴스(값)를 삭제하는 데이터 조작어(DML)이다.

▼ DELETE 명령문

구문	설명
DELETE FROM 테이블명 [WHERE 조건];	• 레코드를 삭제해도 테이블 구조는 남아 있어서 디스크에서 테이블을 완전히 삭제하는 DROP 명령과는 다름 • 모든 레코드를 삭제할 때는 WHERE 절 없이 DELETE만 사용

② DELETE 명령어 사례

학번	성명	학년	수강과목
1001	홍길동	1	수학
1004	임꺽정	2	계산이론

〈학생 테이블〉

'학생' 테이블에서 학번이 1004인 학생의 데이터를 삭제하는 경우

```
-- WHERE 절에 해당하는 학번이 1004인 데이터들만 삭제를 수행
DELETE FROM 학생
 WHERE 학번 = '1004';
```

'학생' 테이블에서 모든 학생의 데이터를 삭제하는 경우

```
-- 모든 행을 변경할 경우 WHERE 절을 포함하지 않음
DELETE FROM 학생;
```

3 데이터 제어어(DCL)★★★

(1) 데이터 제어어(DCL; Data Control Language)의 개념

데이터베이스 관리자가 데이터 보안, 무결성 유지, 병행제어, 회복하기 위해 관리자(DBA)가 사용하는 제어용 언어이다.

▼ DCL 조작 대상

오브젝트	목적	내용
사용자	권한 접근 통제	사용자를 등록하고, 사용자에게 특정 데이터베이스를 사용할 수 있는 권리를 부여하는 작업 예 GRANT, REVOKE
트랜잭션	안전한 거래 보장	동시에 다수의 작업을 독립적으로 안전하게 처리하기 위한 상호작용 단위 예 COMMIT, ROLLBACK, CHECKPOINT

(2) 데이터 제어어의 유형

- 데이터 제어어(DCL)의 유형에는 GRANT, REVOKE, COMMIT, ROLL BACK, CHECKPOINT가 있다.
- COMMIT, ROLLBACK, CHECKPOINT는 DCL이기도 하지만 TCL (Transaction Control Language)이라고도 불린다.

▼ DCL의 유형

유형	명령어	동작	설명
DCL	GRANT	사용 권한 부여	관리자(DBA)가 사용자에게 데이터베이스에 대한 권한을 부여하는 명령어
	REVOKE	사용 권한 회수	관리자(DBA)가 사용자에게 부여했던 권한을 회수하기 위한 명령어
TCL	COMMIT	트랜잭션 확정	데이터베이스 트랜잭션의 내용 업데이트를 영구적으로 확정하는 명령어
	ROLLBACK	트랜잭션 취소	데이터베이스에서 업데이트 오류가 발생할 때, 이전 상태로 되돌리는 명령어
	CHECKPOINT (SAVEPOINT)	복귀지점 설정	트랜잭션의 특정 지점에 이름을 지정하고, 그 지점 이전에 수행한 작업에 영향을 주지 않고 그 지점 이후에 수행한 작업을 롤백할 수 있는 명령어

두음쌤 한마디

DCL 유형
「그래, 코로사」
GRANT / **RE**VOKE /
COMMIT / **RO**LLBACK /
SAVEPOINT
→ 그래, 코로나 바이러스가 사라졌다.

(3) 데이터 제어어 명령문

1 GRANT(권한 부여) 명령문

① GRANT 명령문 개념

- 데이터베이스 관리자(DBA; Database Administrator)가 사용자에게 데이터베이스에 대한 권한을 부여하는 명령어이다.

▼ GRANT 명령문

권한	구문	설명
시스템 권한	GRANT 권한 TO 사용자;	관리자가 사용자에게 시스템 권한을 부여
객체 권한	GRANT 권한 ON 테이블 TO 사용자;	관리자가 사용자에게 테이블에 대한 권한을 부여

- GRANT 구문에 WITH GRANT OPTION 키워드를 붙이면 권한이 필요할 경우 다른 사용자에게 부여할 수 있는 권한을 부여할 수 있다.

두음쌤 한마디

GRANT 명령문
「그온투」
GRANT / **ON** / **TO**

핵인싸 퀴즈

7 ( )(이)란 특정 요구사항에 준수하는지를 확인하기 위해 개발된 입력값, 실행 조건, 예상된 결과의 집합이다.

8 소프트웨어 생명 주기의 ( )은/는 소프트웨어 생명 주기(Life Cycle)의 각 단계별로 개발자 관점에서의 공정 과정상 Verification(검증)과 사용자 관점에서의 최종 산출물에 대한 Validation(확인)을 지원하기 위한 테스트 모델이다.

9 ( )(이)란 개발단계의 제품이 단계의 시작 부분에서 부과된 조건을 만족시키는지를 결정하기 위해 소프트웨어를 평가하는 과정이다.

정답 7. 테스트 케이스(Test Case) 8. V-모델 9. Verification (검증)

② GRANT(권한 부여) 명령문으로 부여할 수 있는 권한의 유형

▼ GRANT 명령문으로 부여할 수 있는 권한의 유형

구분	명령어	설명
시스템 권한	CREATE USER	계정을 생성할 수 있는 권한
	DROP USER	계정을 삭제할 수 있는 권한
	DROP ANY TABLE	테이블을 삭제할 수 있는 권한
	CREATE SESSION	데이터베이스를 접속할 수 있는 권한
	CREATE TABLE	테이블을 생성할 수 있는 권한
	CREATE VIEW	뷰를 생성할 수 있는 권한
	CREATE SEQUENCE	시퀀스를 생성할 수 있는 권한
	CREATE PROCEDURE	함수를 생성할 수 있는 권한
객체 권한	ALTER	테이블을 변경할 수 있는 권한
	INSERT	데이터를 조작할 수 있는 권한
	DELETE	
	SELECT	
	UPDATE	
	EXECUTE	프로시저 실행 권한

③ GRANT 명령어 사례

- 관리자가 사용자인 '장길산'에게 '학생' 테이블에 대해 수정(UPDATE)할 수 있는 권한을 부여

```
GRANT UPDATE ON 학생 TO 장길산;
```

- 관리자가 사용자 '장길산'에게 '학생' 테이블에 대해 조회(SELECT)할 수 있는 권한과 그 권한을 다른 사용자에게 부여할 수 있는 권한 부여

```
GRANT SELECT ON 학생 TO 장길산 WITH GRANT OPTION;
```

2 REVOKE(권한 회수) 명령어

① REVOKE 명령문 개념

- 데이터베이스 관리자(DBA)가 사용자에게 부여했던 권한을 회수하기 위한 명령어이다.

▼ REVOKE 명령어

권한	구문	설명
시스템 권한	REVOKE 권한 FROM 사용자;	관리자가 사용자에게 시스템 권한을 회수
객체 권한	REVOKE 권한 ON 테이블 FROM 사용자;	관리자가 사용자에게 테이블에 대한 권한을 회수

- REVOKE 구문에 CASCADE CONSTRAINTS 키워드를 붙이면 WITH GRANT OPTION으로 부여된 사용자들의 권한까지 회수할 수 있다.

잠깐! 알고가기

데이터베이스 관리자(DBA)
한 조직 내에서 조직 내 데이터베이스를 설치, 구성, 업그레이드, 관리, 감시하는 업무를 수행하는 사람이다.

두음쌤 한마디

REVOKE 명령문
「리온프」
REVOKE / **ON** / **FROM**

② REVOKE 명령어 사례

• 관리자가 사용자인 '장길산'에게 '학생' 테이블에 대해 수정(UPDATE)할 수 있는 권한을 회수

```
REVOKE UPDATE ON 학생 FROM 장길산;
```

• 관리자가 사용자 '장길산'에게 '학생' 테이블에 대해 조회(SELECT)할 수 있는 권한과 WITH GRANT OPTION으로 부여된 사용자들의 권한까지 회수

```
REVOKE SELECT ON 학생 FROM 장길산 CASCADE CONSTRAINT;
```

(4) TCL 명령문

1 TCL(Transaction Control Language) 명령문 개념

TCL은 트랜잭션을 제어하는 언어로 COMMIT, ROLLBACK, CHECKPOINT (SAVEPOINT)가 있다.

▼ TCL 명령문

TCL 명령문	구문	설명
커밋	COMMIT;	이전까지 트랜잭션의 내용 업데이트 확정
롤백	ROLLBACK;	COMMIT 이후에 발생한 트랜잭션 취소
세이브 포인트 지정	SAVEPOINT 이름;	특정 지점을 지정
세이브 포인트 롤백	ROLLBACK TO SAVEPOINT 이름;	특정 지점 지정한 부분으로부터 발생한 트랜잭션 취소

2 TCL 명령문 사례

• '임시' 테이블에서 COMMIT과 ROLLBACK을 같이 사용하는 경우(삽입은 1과 2 두 개가 됐지만, 실행 완료 후 테이블에 1만 남아있음)

```
INSERT INTO 테이블 VALUES(1);
COMMIT; -- COMMIT 명령어 동작 시 기존 명령어가 DB에 영구적으로 삽입
INSERT INTO 테이블 VALUES(2);
ROLLBACK; --ROLLBACK 명령어 동작 시 COMMIT 이후 명령어는 원상 복구
```

• '임시' 테이블에서 SAVEPOINT, COMMIT, ROLLBACK을 같이 사용하는 경우(실행 완료 후 테이블에 1, 3이 남아있음)

```
INSERT INTO 테이블 VALUES(1);
SAVEPOINT aaa; -- SAVEPOINT를 aaa라는 이름으로 설정
INSERT INTO 테이블 VALUES(2);
ROLLBACK TO SAVEPOINT aaa;
-- SAVEPOINT aaa 지정된 부분부터 현재 위치 사이에 실행한 명령어 원상복구
INSERT INTO 테이블 VALUES(3);
COMMIT;
```

핵인싸 퀴즈

10 ( )(이)란 소프트웨어가 특정 요구조건을 만족시키는가를 결정하기위해 개발과정 중 또는 끝에 소프트웨어를 평가하는 과정이다.

11 ( )은/는 작은 소프트웨어 단위(컴포넌트 또는 모듈)를 테스트하는 것으로서, 일반적으로 개발자가 수행하는 테스트이다.

12 ( )은/는 단위 테스트를 통과한 모듈 사이의 인터페이스, 통합된 컴포넌트 간의 상호 작용을 테스트하는 단계이다.

정답 10. Validation(확인) 11. 단위 테스트 12. 통합 테스트

4 접근제어☆

(1) 접근제어(Access Control) 개념

- 불법적인 데이터의 접근으로부터 데이터베이스를 보호하는 기법이다.
- 데이터베이스는 가장 내부에 위치하고, DBMS 자체는 강력한 보안 기능을 제공하기 때문에 접근 권한을 가진 사용자가 권한을 남용하여 유출하거나 변조가 가장 큰 위험이다.

(2) 접근제어 구성요소

▼ 접근제어 구성요소

구성요소	설명
접근제어 정책	자원에 접근하는 사용자의 접근 모드 및 모든 접근 제한 조건 정의
접근제어 메커니즘	시도된 접근 요청을 정의된 규칙에 대응시켜 검사함으로써 불법적 접근 방어
접근제어 보안 모델	시스템의 보안 요구를 나타내는 요구 명세로부터 출발하여 정확하고 간결한 기능적 모델 표현

두음쌤 한마디

접근제어 구성요소

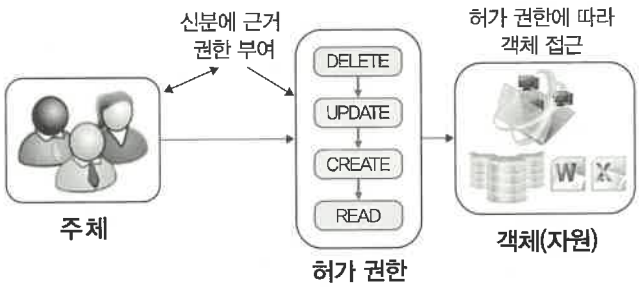
「정메보」

정책 / 메커니즘 / 보안모델

(3) 접근제어 정책

1 신원 기반(Identity-based) 접근제어 정책

- 주체나 그들이 속해있는 그룹들의 신분에 근거하여 객체에 대한 접근을 제한하는 방법이다.
- DAC(Discretionary Access Control)이라고 불린다.



▲ DAC(Discretionary Access Control)

핵인싸 퀴즈

13 ( )은/는 통합된 단위 시스템의 기능이 시스템에서 정상적으로 수행되는지를 테스트로 성능 및 장애 테스트가 있다.

14 ( )은/는 최종 사용자와 업무의 이해관계자 등이 테스트를 수행함으로써 개발된 제품에 대해 운영 여부를 결정하는 테스트이다.

15 ( )은/는 프로그램 내부 구조 및 복잡도를 검증하는 화이트박스(White Box) 테스트이다.

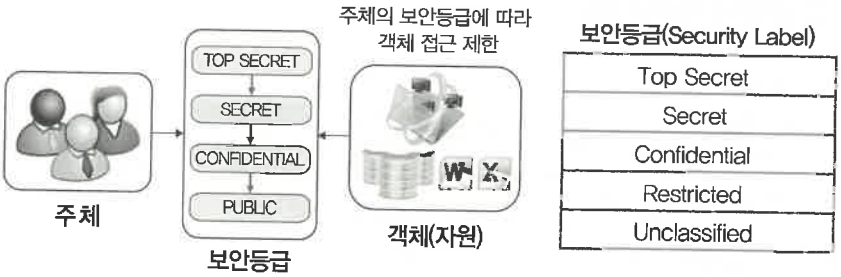
정답 13. 시스템 테스트 14. 인수 테스트 15. 구조 기반 테스트

▼ 신원 기반 접근제어 정책

정책	설명
IBP(Individual-Based Policy)	단일 사용자가 하나의 객체에 대해 허가를 부여받아 사용하는 정책
GBP(Group-Based Policy)	복수 사용자가 하나의 객체에 대하여 같은 허가를 함께 부여받아 사용하는 정책

2 규칙 기반(Rule-based) 접근제어 정책

- 객체에 포함된 정보의 비밀성과 이러한 비밀성의 접근정보에 대하여 주체가 갖는 권한에 근거하여 객체에 대한 접근을 제한하는 방법이다.
- MAC(Mandatory Access Control)이라고 불린다.



▲ MAC(Mandatory Access Control)

▼ 규칙 기반 접근제어 정책

정책	설명
MLP(Multi-Level Policy)	사용자 및 객체가 각각 부여된 기밀 분류에 따른 정책
CBP(Compartment-Based Policy)	조직 내 특정 집단별로 구분된 기밀 허가에 따른 정책

3 역할 기반(Role-based) 접근제어 정책

- 중앙관리자가 주체와 객체의 상호관계를 제어하며 조직 내에서 맡은 역할에 기초하여 자원에 대한 접근 허용 여부 결정한다.
- RBAC(Role Based Access Control)이라고 불린다.

핵인싸 퀴즈

16 ( )은/는 목적 및 실행 코드 기반의 실행을 통한 블랙박스(Black Box) 테스트이다.

17 ( )은/는 메인 제어 모듈(프로그램)로부터 아래 방향으로 제어의 경로를 따라 이동하여 하향식으로 통합하면서 테스트를 진행한다.

18 ( )은/는 애플리케이션 구조에서 최하위 레벨의 모듈 또는 컴포넌트로부터 위쪽으로 제어의 경로를 따라 이동하면서 구축과 테스트를 시작하는 방식이다.

정답 16. 명세 기반 테스트 17. 하향식 통합 18. 상향식 통합

두음쌤 한마디

접근제어 정책

「댁맥알백」

DAC / MAC / RBAC

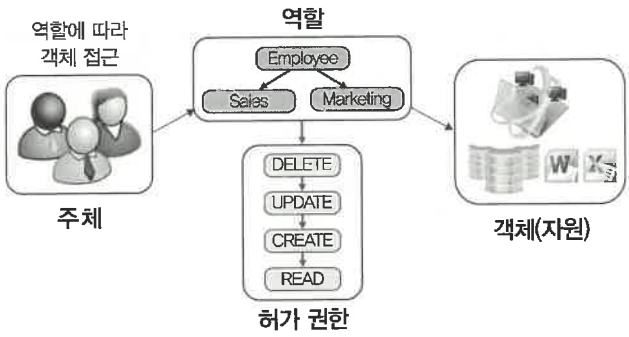
핵인싸 퀴즈

19 ( )(이)란 모듈 및 모든 하위 컴포넌트를 대신하는 더미 모듈로 하향식 통합에 사용된다.

20 ( )(이)란 상위의 모듈에서 데이터의 입력과 출력을 확인하기 위한 더미 모듈로 상향식 통합에 사용된다.

21 ( )(이)란 모든 모듈을 동시에 통합 후 테스트를 수행하는 방식으로 드라이버/스텁 없이 실제 모듈로 테스트한다.

정답 19. 스텁(Stub) 20. 드라이버(Driver) 21. 빅뱅 테스트



▲ RBAC(Role Based Access Control)

(4) 접근제어 메커니즘

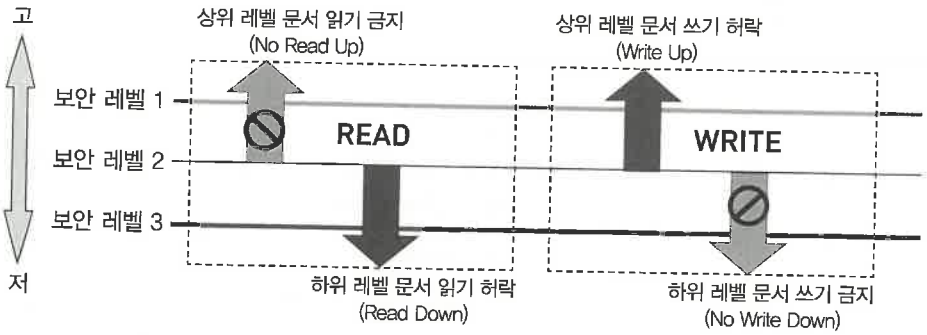
▼ 접근제어 메커니즘

메커니즘	설명
ACL(Access Control List)	주체가 디렉터리나 파일과 같은 특정 시스템 객체에 접근할 수 있는 허가 받은 접근 종류들이 기록된 목록
CL(Capability List)	주체에게 허가된 자원 및 권한의 목록

(5) 접근제어 보안 모델

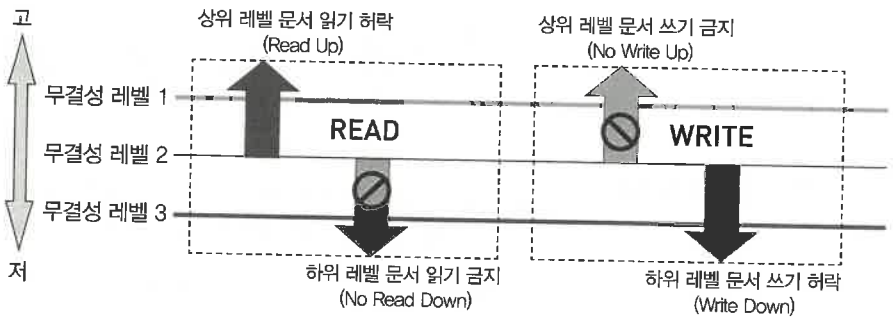
▼ 접근제어 보안 모델

모델	설명
접근제어 행렬	• 임의적 접근제어를 관리하기 위한 보안 모델 • 행은 주체, 열은 객체를 표시 • 행과 열은 주체 및 객체의 권한 유형을 표시
기밀성 모델	• 기밀성(Confidentiality)에 중점을 둔 수학적인 모델 • 일반적인 상용 환경에서는 기밀성보다는 무결성의 중요성이 높아 부적합(군대 시스템 등 특수 환경에서 사용되는 모델) • 모델은 주체(사용자, 계정 등)와 객체(릴레이션, 튜플, 속성, 뷰)의 보안 등급을 근거로 제약 조건을 준수 • 등급 기준: 극비(Top Secret) 〉 비밀(Secret) 〉 일반(Confidential) 〉 미분류(Unclassified) • 벨라파듈라 모델(Bell-LaPadula Model)이 존재
무결성 모델	• 정보 비밀성을 위해 정보의 일방향 흐름 제어를 활용하는 경우 발생 • 기밀성 모델과 같이 주체 및 객체의 보안 등급을 기반으로 하며, 제약조건 역시 유사하게 적용 • 비바 모델(Biba Model)과 클락 윌슨 모델(Clark-Wilson Model)이 존재



▲ 벨라파듈라 모델(Bell-LaPadula Model)

- MITRE의 D.Bell과 L.LaPadula에 의해 개발되어 처음으로 수학적 모델을 사용한 접근통제 모델이다.



▲ 비바 모델(Biba Model)

- 무결성 통제를 위해 개발된 모델로, BLP 모델에서 불법 수정 방지 내용을 추가로 정의한 접근통제 모델이다.

핵인싸 퀴즈

22 (　　)은/는 최종 사용자와 업무의 이해관계자 등이 테스트를 수행함으로써 개발된 제품에 대해 운영 여부를 결정하는 테스트이고 실제 업무 적용 전에 수행된다.

23 (　　) 테스트는 개발하는 조직 내 잠재 고객에 의해 테스트를 수행한다.

24 코드 커버리지 유형 중 (　　)은/는 프로그램 내 모든 문장을 적어도 한 번 이상 실행하는 것을 기준으로 수행하는 테스트 커버리지로 조건문 결과와 관계없이 구문 실행 개수로 계산한다.

정답 22. 인수 테스트 23. 알파 24. 구문 커버리지

CHAPTER

02 고급 SQL 작성하기

1 인덱스★★

(1) 인덱스(Index) 개념

- 인덱스는 데이터를 빠르게 찾을 수 있는 수단으로서, 테이블에 대한 조회 속도를 높여 주는 자료구조이다.
- 인덱스는 테이블의 특정 레코드 위치를 알려주는 용도로 사용한다.

Index_name

Index(이름)	주소
김구	5
이육사	4
이순신	6
을지문덕	3
장보고	1
홍길동	2

Table_Avengers

이름	주소	생년월일	출생
장보고	5	19850111	완도
홍길동	4	19131130	장성
을지문덕	6	20001201	서울
이육사	3	19930105	평택
김구	1	19870712	해주
이순신	2	19500322	거제

▲ 인덱스 개념

- 기본키(PK; Primary Key) 컬럼은 자동으로 인덱스가 생성된다.
- 연월일이나 이름을 기준으로 하는 인덱스는 자동으로 생성되지 않는다.
- 조건절에 '='로 비교되는 컬럼을 대상으로 인덱스를 생성하면 검색 속도를 높일 수 있다.

핵인싸 퀴즈

25 코드 커버리지 유형 중 ( ) 은/는 결정 명령문 내의 각 조건이 적어도 한 번은 참과 거짓의 결과가 되도록 수행하는 테스트 커버리지이다.

26 코드 커버리지 유형 중 ( ) 은/는 전체 조건식뿐만 아니라 개별 조건식도 참 한 번, 거짓 한 번 결과가 되도록 수행하는 테스트 커버리지이다.

27 코드 커버리지 유형 중 ( ) 은/는 각 개별 조건식이 다른 개별 조건식에 영향을 받지 않고 전체 조건식의 독립적으로 영향을 주도록 함으로써 조건/결정 커버리지를 향상시킨 테스트 커버리지이다.

정답 25. 조건 커버리지 26. 조건/결정 커버리지 27. 변경 조건/결정 커버리지

(2) 인덱스 사례

```
SELECT *
  FROM 사원
 WHERE 이름 = '홍길동';
```

- 테이블의 '이름' 컬럼에 대한 인덱스가 없는 경우, 테이블의 전체 내용을 검색(테이블 전체 스캔; Table Full Scan)하여 속도가 느리다.
- 테이블의 '이름' 컬럼에 대한 인덱스가 있는 경우(인덱스 범위 스캔; Index Range Scan), 데이터를 빠르게 찾을 수 있다.
- 조건절(WHERE 절)에 '='로 비교하여 검색 속도가 빠르다.

(3) 인덱스 종류

▼ 인덱스 종류

유형	설명
순서 인덱스 (Ordered Index)	• 데이터가 정렬된 순서로 생성되는 인덱스 • B-Tree 알고리즘 활용(오름차순/내림차순 지정 가능)
해시 인덱스 (Hash Index)	• 해시함수에 의해 직접 데이터에 키 값으로 접근하는 인덱스 • 데이터 접근비용이 균일, 튜플(Row) 양에 무관
비트맵 인덱스 (Bitmap Index)	• 각 컬럼에 적은 개수 값이 저장된 경우 선택하는 인덱스 • 수정 변경이 적을 때 유용(생년월일, 상품번호 등)
함수기반 인덱스 (Functional Index)	• 수식이나 함수를 적용하여 만든 인덱스
단일 인덱스 (Singled Index)	• 하나의 컬럼으로만 구성한 인덱스 • 주로 사용 컬럼이 하나일 경우 사용
결합 인덱스 (Concatenated Index)	• 두 개 이상의 컬럼으로 구성한 인덱스 • WHERE 조건으로 사용하는 빈도가 높은 경우 사용
클러스터드 인덱스 (Clustered Index)	• 기본키(PK) 기준으로 레코드를 묶어서 저장하는 인덱스 • 저장 데이터의 물리적 순서에 따라 인덱스가 생성 • 특정 범위 검색 시 유리함

(4) 인덱스 조작

1 인덱스 생성

- DBMS는 인덱스를 사용하여 빠른 검색을 수행한다.
- DB 사용자는 DBMS가 인덱스를 사용할 수 있게 준비해 주어야 한다.

```
CREATE [UNIQUE] INDEX 인덱스명 ON 테이블명(컬럼명);
```

- 각 파라미터가 의미하는 내용은 다음과 같다.

핵인싸 퀴즈

28 ( ) 테스트는 입력 데이터의 영역을 유사한 도메인별로 유효값/무효값을 그룹핑하여 대표값 테스트 케이스를 도출하는 기법이다.

29 ( ) 테스트는 동등 분할 후 경계 값 부분에서 오류 발생 확률이 높기에 경곗값을 포함하여 테스트 케이스를 설계하는 기법이다.

30 ( ) 테스트는 요구사항의 논리와 발생 조건을 테이블 형태로 나열하여, 조건과 행위를 모두 조합한 테스트 기법이다.

정답 28. 동등 분할 29. 경곗값 분석 30. 결정 테이블

두음쌤 한마디

인덱스 종류

순해비함 단결클

순서 인덱스 / 해시 인덱스 / 비트맵 인덱스 / 함수기반 인덱스 / 단일 인덱스 / 결합 인덱스 / 클러스터드 인덱스

→ 순해비함의 함장이 단결! 인사하니 클클하며 웃는다.

학습 Point

앞에서 언급했던 DDL을 이용하여 인덱스를 생성, 변경, 삭제를 할 수 있습니다. 테이블 생성은 "CREATE TABLE 테이블명" 이라면 인덱스 생성은 "CREATE INDEX 인덱스명"으로 생성을 위한 명령어가 거의 비슷합니다.

▼ 인덱스 명령문 요소

파라미터	설명
[UNIQUE]	인덱스 걸린 컬럼에 중복 값을 허용하지 않음(생략 가능)
인덱스명	생성하고자 하는 인덱스 테이블명
테이블명	인덱스 대상 테이블명
컬럼명	테이블의 특정 컬럼명(복수 컬럼 지정 가능)

2 인덱스 변경

```
ALTER [UNIQUE] INDEX 인덱스명 ON 테이블명(컬럼명);
```

- 일부 제품은 인덱스에 대한 변경 SQL문이 없다.
- 기존 인덱스를 삭제하고 신규 인덱스를 생성하는 방식으로 사용이 권고된다.

3 인덱스 삭제

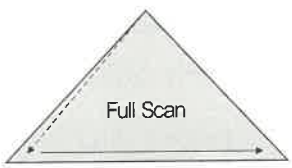
인덱스 삭제 명령 형식은 다음과 같다.

```
DROP INDEX 인덱스명;
```

- '인덱스명'은 생성된 인덱스명을 의미한다.
- 인덱스를 테이블의 종속 구조로 간주, 삭제 시 테이블 변경 명령어가 사용된다.
- ALTER TABLE 명령 뒤에 DROP INDEX 명령이 추가되는 형태로 사용된다.

4 인덱스 스캔

▼ 인덱스 스캔 방식

구분	설명	개념도
인덱스 범위 스캔 (Index Range Scan)	인덱스 루트 블록에서 리프 블록까지 수직적으로 탐색한 후에 리프 블록을 필요한 범위만 스캔하는 방식	
인덱스 전체 스캔 (Index Full Scan)	수직적 탐색 없이 인덱스 리프 블록을 처음부터 끝까지 수평적으로 탐색하는 방식	

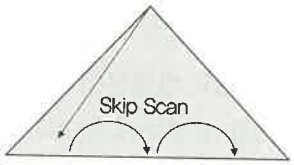
핵인싸 퀴즈

31 ( ) 테스트는 테스트 대상/시스템이나 객체의 상태를 구분하고, 이벤트에 의해 어느 한 상태에서 다른 상태로 전이되는 경우의 수를 테스트하는 기법이다.

32 ( ) 테스트는 시스템이 실제 사용되는 유즈케이스로 모델링 되어 있을 때 프로세스 흐름을 기반으로 테스트케이스를 명세화하여 수행하는 기법이다.

33 ( ) 테스트는 소프트웨어의 일부 또는 전체를 트리 구조로 분석 및 표현하여 테스트 케이스를 설계하는 기법이다.

정답 31. 상태 전이 32. 유스케이스 33. 분류 트리

구분	설명	개념도
인덱스 단일 스캔 (Index Unique Scan)	수직적 탐색만으로 데이터를 찾는 스캔 방식	
인덱스 생략 스캔 (Index Skip Scan)	선두 컬럼이 조건절에 빠졌어도 인덱스를 활용하는 스캔 방식	

두음쌤 한마디

인덱스 스캔 방식

「범전단생」

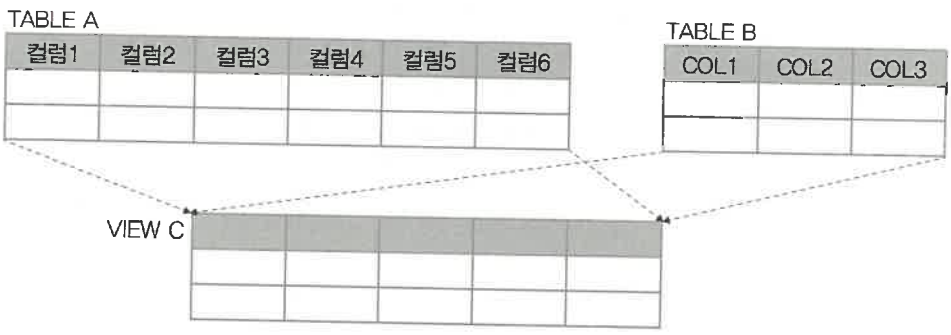
법위 스캔 / 전체 스캔 / 단일 스캔 / 생략 스캔

→ 법생이가 전단을 돌려 생계유지

2 뷰★★

(1) 뷰(View)의 개념

- 뷰는 기본 테이블로부터 유도된 가상 테이블로 물리적으로 구현되어 있지 않지만 기본 테이블과 같은 형태의 구조이다.
- 논리 테이블로서 사용자에게(생성 관점이 아닌 사용 관점에서) 테이블과 같다.
- 아래 그림에서 '테이블 A'와 '테이블 B'는 물리 테이블을 의미하고, '뷰 C'는 두 개의 테이블을 이용하여 생성한 뷰를 의미한다.



▲ 뷰의 개념

핵인싸 퀴즈

34 ( ) 테스트는 테스트 데이터값들 간에 최소한 한 번씩을 조합하는 방식이며, 이는 커버해야 할 기능적 범위를 모든 조합에 비해 상대적으로 적은 양의 테스트 세트를 구성하기 위한 기법이다.

35 ( ) 테스트는 유사 소프트웨어나 유사 기술 평가에서 테스터의 경험을 토대로 한, 직관과 기술 능력을 기반으로 수행하는 테스트 기법이다.

36 ( ) 테스트는 시스템에 고의로 실패를 유도하고 시스템의 정상적 복귀 여부 테스트 기법이다.

정답 34. 페어 와이즈 35. 경험 기반 36. 회복

잠깐! 알고가기

조인(Join)
결합을 의미하며, 관계형 데이터베이스에서의 조인은 교집합 결과를 가지는 결합 방법을 의미한다.

- 뷰는 테이블 'A'와 같은 하나의 물리 테이블로부터 생성 가능하며, 다수의 테이블 또는 다른 뷰를 이용해 만들 수 있다.
- 뷰와 같은 결과를 만들기 위해 조인 기능을 활용할 수 있으나, 뷰가 만들어져 있다면 사용자는 조인 없이 하나의 테이블을 대상으로 하는 단순한 질의어를 사용할 수 있다.

(2) 뷰의 특징

- 데이터의 논리적 독립성을 제공할 수 있다.
- 관리가 쉽고 명령문이 간단해진다.
- 데이터를 안전하게 보호하는 효율적인 기법이다.

(3) 뷰의 목적

뷰를 사용하는 주된 이유는 단순 질의어를 사용할 수 있기 때문이다.

```
SELECT * FROM 뷰 이름;
```

- FROM 절에 있는 하나의 뷰를 통해 뷰를 구성하는 복수의 테이블을 대체하는 단순성에 그 의의가 있다.
- 테이블의 중요 데이터 일부만을 제공할 수 있는 장단점이 있다.

▼ 뷰의 장단점

구분	장단점	설명
장점	논리적 독립성 제공	• 뷰는 논리 테이블(테이블의 구조가 변경되어도 뷰를 사용하는 응용 프로그램은 변경하지 않아도 됨)
	사용자 데이터 관리 용이	• 복수 테이블에 존재하는 여러 종류의 데이터에 대해 단순한 질의어 사용이 가능
	데이터 보안의 용이	• 중요 보안 데이터를 저장 중인 테이블에는 접근 불허 • 해당 테이블의 일부 정보만을 볼 수 있는 뷰에는 접근을 허용 • 보안 데이터에 대한 접근제어 가능
단점	뷰 자체 인덱스 증가	• 인덱스는 물리적으로 저장된 데이터를 대상으로 하기에 논리적 구성인 뷰 자체는 인덱스를 가지지 못함
	뷰 정의 변경 불가	• 뷰의 정의를 변경하려면 뷰를 삭제하고 재생성
	데이터 변경 제약 존재	• 뷰의 내용에 대한 삽입, 삭제, 변경 제약이 있음

핵인싸 퀴즈

37 ( ) 테스트는 불법적인 소프트웨어가 접근하여 시스템을 파괴하지 못하도록 소스 코드 내의 보안적인 결함을 미리 점검하는 테스트 기법이다.

38 ( ) 테스트는 시스템에 과다 정보량을 부과하여 과부하 시에도 시스템이 정상적으로 작동되는지를 검증하는 테스트 기법이다.

39 ( ) 테스트는 사용자의 이벤트에 시스템이 응답하는 시간, 특정 시간 내에 처리하는 업무량, 사용자 요구에 시스템이 반응하는 속도 등을 측정하는 테스트 기법이다.

정답 37. 안전 38. 강도 39. 성능

(4) 뷰 생성

뷰 생성의 일반 형태는 다음과 같다.

```
CREATE VIEW 뷰 이름 컬럼 목록 AS 데이터 조회 쿼리;
```

상황별로 뷰를 생성하는 방법은 다음과 같다.

▼ 뷰의 생성 방법

상황	뷰 생성 쿼리문
테이블 A 그대로 VW_A 뷰로 생성	CREATE VIEW VW_A AS SELECT * FROM A;
테이블 A 일부 컬럼을 VW_B 뷰로 생성	CREATE VIEW VW_B AS SELECT 컬럼1, 컬럼2 FROM A;
테이블 A와 테이블 B 조인 결과를 VW_C 뷰로 생성	CREATE VIEW VW_C AS SELECT * FROM A, B WHERE A.컬럼1 = B.컬럼1;

- SELECT문에는 UNION이나 ORDER BY 절을 사용할 수 없다.
- 컬럼명을 기술하지 않으면 SELECT문의 컬럼명이 자동으로 사용된다.

(5) 뷰 삭제 / 변경

- 뷰 정의 자체를 변경하는 것은 불가능하다.
- 뷰 이름이나 쿼리문을 변경하는 수단은 제공되지 않는다.
- 뷰 삭제와 재생성을 통해 뷰에 대한 정의 변경이 가능하다.

```
DROP VIEW 뷰 이름;
```

- 뷰를 통해 접근 가능한 데이터에 대한 변경이 가능하다.
- 모든 경우에 데이터의 변경이 가능한 것이 아니라 일부 제약이 존재한다.

예 기본키(PK; Primary Key)에 해당하는 컬럼이 뷰에 정의되어 있지 않으면 INSERT 불가능

핵인싸 퀴즈

40 ( ) 테스트는 변경 또는 수정된 코드에 대하여 새로운 결함 발견 여부를 평가하는 테스트 기법이다.

41 ( ) 테스트는 변경된 시스템과 기존 시스템에 동일한 데이터를 입력 후 결과를 비교하는 테스트 기법이다.

42 ( )은/는 테스트 도구를 활용하여 반복적인 테스트 작업을 스크립트 형태로 구현함으로써, 테스트 시간 단축과 인력 투입 비용을 최소화하는 한편, 쉽고 효율적인 테스트를 수행할 수 있는 방법이다.

정답 40. 회귀 41. 병행 42. 테스트 자동화

3 다중 테이블 검색★

(1) 다중 테이블 검색 방법

- 관계형 데이터베이스는 데이터의 중복을 최소화하기 위해 데이터를 분해하여 저장하고 통합하여 사용한다.
- 데이터를 분해하는 방법으로 정규화 기법이 사용되며, 통합하는 기법으로 다중 테이블에 대한 검색이 사용된다. 다중 테이블을 이용하는 보다 세부적인 기법은 다음과 같다.

▼ 다중 테이블 검색 방법

다중 테이블 검색 기법	내용
조인	두 개의 테이블을 결합하여 데이터를 추출하는 기법
서브 쿼리	SQL문 안에 포함된 SQL문 형태의 사용 기법
집합 연산	테이블을 집합 개념으로 조작하는 기법

두음쌤 한마디

다중 테이블 검색 방법

「조서집」

조인 / 서브 쿼리 / 집합 연산

(2) 조인

1 조인(Join) 개념

- 조인은 결합을 의미하며, 관계형 데이터베이스에서의 조인은 교집합 결과를 가지는 결합 방법을 의미한다.
- 두 릴레이션으로부터 관련된 튜플들을 결합하여 하나의 튜플로 만드는 가장 대표적인 데이터 연결 방법이다.

2 조인 유형

- 조인은 관계형 데이터베이스의 가장 큰 장점이면서 대표적인 핵심 기능이다.
- 논리적 조인과 물리적 조인으로 구분할 수 있다.

▼ 조인 유형

분류	설명	유형
논리적 조인	• 사용자 SQL문에 표현되는 테이블 결합 방식	• 내부 조인 • 외부 조인
물리적 조인	• 데이터베이스 옵티마이저에 의해 내부적으로 발생하는 테이블 결합 방식	• 중첩 반복 조인 • 정렬 합병 조인 • 해시 조인

두음쌤 한마디

조인 유형

「논내외, 물중정해」

논리적 조인 / 내부 조인 / 외부 조인 / 물리적 조인 / 중첩 반복 조인 / 정렬 합병 조인 / 해시 조인

3 논리적 조인

▼ 논리적 조인 유형(상세)

구분	조인 유형	설명
내부 조인 (Inner Join)	동등 조인 (Equi Join)	공통 존재 컬럼의 값이 같은 경우를 추출
	자연 조인 (Natural Join)	두 테이블의 모든 컬럼을 비교하여 같은 컬럼명을 가진 값이 같은 경우를 추출
	교차 조인 (Cross Join)	조인 조건이 없는 모든 데이터 조합을 추출
외부 조인 (Outer Join)	왼쪽 외부 조인 (Left Outer Join)	왼쪽 테이블의 모든 데이터와 오른쪽 테이블의 동일 데이터를 추출
	오른쪽 외부 조인 (Right Outer Join)	오른쪽 테이블의 모든 데이터와 왼쪽 테이블의 동일 데이터를 추출
	완전 외부 조인 (Full Outer Join)	양쪽의 모든 데이터를 추출

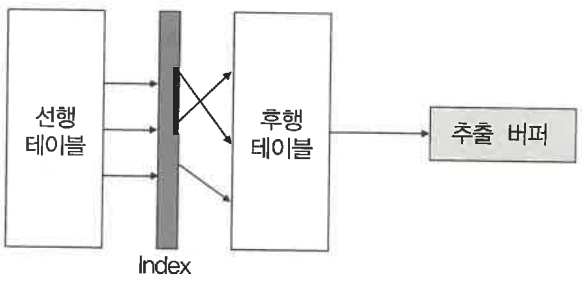
- 내부 조인의 세부 유형은 조인의 조건에 따라 세분화된다.
- 내부 조인에서 조인의 대상이 되는 컬럼을 명시적으로 선언하기 위하여 USING 조건절이나 ON 조건절이 사용된다.

4 물리적 조인

① 중첩 반복 조인

㉮ 중첩 반복 조인(Nested-Loop Join) 개념

- 2개 이상의 테이블에서 하나의 집합을 기준으로 순차적으로 상대방 Row를 결합하여 원하는 결과를 조합하는 방식이다.
- 선행 테이블의 처리 범위를 하나씩 액세스하면서 추출된 값으로 연결할 테이블을 조인한다.



▲ 중첩 반복 조인 수행 과정

두음쌤 한마디

내부 조인

「동자교」

동등 조인 / 자연 조인 / 교차 조인

→ 동자승을 위한 교육

핵인싸 퀴즈

43 ( )이란 컴퓨터상에 가상으로 컴퓨터를 구동시키는 것으로 물리적인 하드웨어를 가상화하여, 하나의 물리적 하드웨어상에서 여러 컴퓨터가 구동되는 것처럼 에뮬레이션하는 것이다.

44 컴퓨터의 동작이나 시스템의 적합성을 시험하기 위해 특별히 개발된 데이터 집합이고, 프로그램의 기능을 하나씩 순번에 따라 확실하게 테스트할 수 있도록 조건을 갖춘 데이터는 ( )이다.

45 정상적인 테스트를 모두 수행한 경우, 차기 일정의 도래로 테스트 일정이 만료되었을 경우, 테스트에 소요되는 비용을 모두 소진한 경우 등 업무 기능의 중요도에 따라 조건 설정 변경 가능한 조건을 ( )(이)라고 한다.

정답 43. 가상 머신 44. 테스트 데이터 45. 테스트 종료 조건

핵인싸 퀴즈

46 (　　)(이)란 테스트 수행을 위한 여러 개의 테스트 케이스의 집합으로 테스트 케이스의 동작 순서를 기술한 문서이며, 테스트 절차를 명세한 문서이다.

47 결함의 상태 및 추적 프로세스에서 결함 (　　)은/는 테스터와 품질 관리(QC) 담당자에 의해 결함이 처음 발견되어 등록되었지만, 아직 분석이 되지 않은 상태이다.

48 결함의 상태 및 추적 프로세스에서 결함 (　　)은/는 수정이 필요한 결함이지만 현재 수정이 불가능해서 연기된 상태로서 우선순위, 일정 등을 고려하여 재 오픈을 준비하는 상태이다.

정답 46. 테스트 시나리오 47. 등록 48. 조치 보류

㉯ 중첩 반복 조인 특징

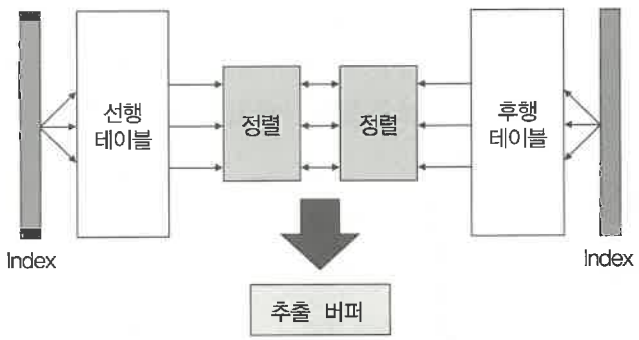
- 좁은 범위에 유리한 성능을 보여준다.
- 순차적으로 처리하며, 임의 접근(Random Access) 위주이다.
- 후행 테이블(Driven)에는 조인을 위한 인덱스 생성이 필요하다.

(실행속도) = (선행 테이블 사이즈) × (후행 테이블 접근 횟수)

② 정렬 합병 조인

㉮ 정렬 합병 조인(Sort-Merge Join) 개념

- 조인의 대상 범위가 넓을 때 발생하는 임의 접근(Random Access)을 줄이기 위한 경우나 연결고리에 마땅한 인덱스가 존재하지 않을 경우 해결하기 위한 조인 방법이다.
- 양쪽 테이블의 정렬한 결과를 차례로 검색하면서 연결고리 형태로 합병하는 방식이다.



▲ 정렬 합병 조인 수행 과정

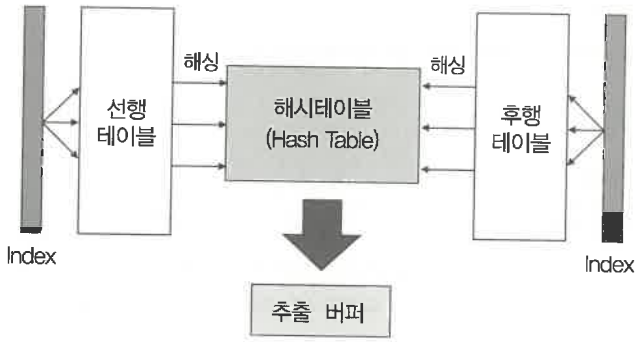
㉯ 정렬 합병 조인 특징

- 연결을 위해 랜덤 액세스를 하지 않고 스캔을 하면서 수행한다.
- 중첩 반복 조인(Nested Loop Join)처럼 선행집합 개념이 없다.
- 정렬을 위한 영역(Sort Area Size)에 따라 효율에 큰 차이가 발생한다.
- 조인 연산자가 '='이 아닌 경우 중첩 반복 조인(Nested Loop Join)보다 유리한 경우가 많다.

③ 해시 조인

㉮ 해시 조인(Hash Join) 개념

- 해싱 함수(Hashing Function) 기법을 활용하여 조인을 수행하는 방식이다.
- 해싱 함수는 직접적인 연결을 담당하는 것이 아니라 연결될 대상을 특정 지역(Partition)에 모아두는 역할만을 담당한다.



▲ 해시 조인 수행 과정

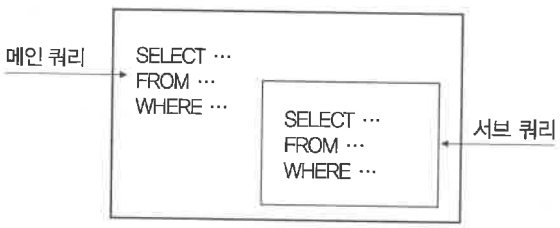
㉮ 해시 조인 특징

- 대용량 처리 선결 조건인 랜덤 액세스와 정렬에 대한 부담을 해결할 수 있는 대안이다.
- 비용기반 옵티마이저(CBO)에서만 가능하며, CPU 성능에 의존적이다.
- 해시 테이블(Hash Table) 생성 후 중첩 반복 조인(Nested-Loop Join)처럼 순차적인 처리 형태로 수행한다.

(3) 서브 쿼리

1 서브 쿼리(Sub-Query) 개념

- 서브 쿼리는 SQL문 안에 포함된 또 다른 SQL문을 의미한다.



▲ 서브 쿼리 개념

- 메인 쿼리와 서브 쿼리 관계는 주종 관계로서, 서브 쿼리에 사용되는 컬럼 정보는 메인 쿼리의 컬럼 정보를 사용할 수 있으나 역으로는 성립하지 않는다.

핵인싸 퀴즈

49 결함의 분류에서 ( ) 결함은 비정상적인 종료/중단, 응답 시간 지연, 데이터베이스 에러 등 주로 애플리케이션 환경과 데이터베이스 처리에서 발생하는 결함이다.

50 결함의 분류에서 ( ) 결함은 사용자의 요구사항 미반영/불일치, 부정확한 비즈니스 프로세스, 스크립트 에러, 타 시스템 연동 시 오류 등 기획, 설계, 업무 시나리오 단계에서 발생된 결함이다.

51 결함 심각도는 여러 개의 결함 중 전체 시스템에 결함이 미치는 영향을 레벨별로 나타내고, 우선순위를 ( ), Medium, Low 등으로 정하는 것을 말한다.

정답 49. 시스템 50. 기능 51. High

핵인싸 퀴즈

52 ( ) 테스트는 전체 시스템이 통합 완료될 때까지 단위 시스템 간의 연계성 및 기능 요구사항들을 확인하고, 하드웨어와 소프트웨어 구성요소 간의 상호작용을 테스트이다.

53 ( ) 테스트는 동시접속으로 시스템에 많은 요청(업데이트, 조회 등)이 발생될 때 어떻게 가동되는지 확인하는 테스트이다.

54 ( )은/는 소프트웨어 개발 또는 유지보수 수행 중에 발생한 부정확한 결과이다.

정답 52. 통합 53. 부하 54. 에러(Error)

2 서브 쿼리 유형

- 서브 쿼리는 동작하는 방식이나 반환되는 데이터의 형태에 따라 분류할 수 있다.

▼ 서브 쿼리 종류(동작 방식 기준)

서브 쿼리 종류	설명
비연관 서브 쿼리 (Un-Correlated Sub-Query)	• 서브 쿼리가 메인 쿼리의 컬럼을 가지고 있지 않은 형태 • 메인 쿼리에 서브 쿼리에서 실행된 결과 제공하는 용도로 사용
연관 서브 쿼리 (Correlated Sub-Query)	• 서브 쿼리가 메인 쿼리의 컬럼을 가지고 있는 형태 • 메인 쿼리가 먼저 수행되어 얻은 데이터를 서브 쿼리의 조건에 맞는지 확인하는 용도로 사용

▼ 서브 쿼리 종류(데이터 형태 기준)

서브 쿼리 종류	설명
단일 행(Single Row) 서브 쿼리	• 결과가 항상 1건 이하인 서브 쿼리 • 단일 행 비교 연산자(=, <, <=, >, >=, <>)가 사용
다중 행(Multiple Row) 서브 쿼리	• 실행 결과가 여러 건인 서브 쿼리 • 다중 행 비교 연산자(IN, ALL, ANY, SOME, EXISTS)가 사용
다중 컬럼(Multiple Column) 서브 쿼리	• 결과가 여러 컬럼으로 반환되는 서브 쿼리 • 메인 쿼리의 조건절에 여러 컬럼을 동시에 비교할 때, 서브 쿼리와 메인 쿼리에서 비교하는 컬럼 개수와 위치가 동일해야 함

(4) 집합 연산자

1 집합 연산자(Set Operator) 개념

- 테이블을 집합 개념으로 보고, 두 테이블 연산에 집합 연산자를 사용하는 방식이다.
- 집합 연산자는 2개 이상의 질의 결과를 하나의 결과로 만들어 준다.

2 집합 연산자 유형

▼ 집합 연산자 유형

집합 연산자	설명
UNION	중복 행이 제거된 쿼리 결과 집합
UNION ALL	중복 행이 제거되지 않은 쿼리 결과 집합
INTERSECTION	두 쿼리 결과에 공통적으로 존재하는 집합
MINUS	첫 쿼리에 있고 두 번째 쿼리에는 없는 집합

두음쌤 한마디

집합 연산자 유형

「유유인마」

유니온 / 유니온올 / 인터섹션 / 마이너스

→ 유유상종이다 인마!

① UNION

- UNION은 합집합을 의미한다.
- 두 개의 데이터 집합의 결과에 대해 중복을 제거하고 모두 포함한 결과를 반환한다.

학번	성명	학년	수강과목
1001	홍길동	1	수학
1004	임꺽정	2	계산이론
1007	장길산	2	알고리즘
1008	신영식	3	수학

〈학생 테이블〉

```
SELECT 성명
  FROM 학생
 WHERE 학년 >= 2
 UNION
SELECT 성명
  FROM 학생
 WHERE 학년 >= 3;
```

- 2학년 이상인 학생들의 성명을 조회하는 쿼리
- 결괏값: 임꺽정, 장길산, 신영식
- 3학년 이상인 학생들의 성명을 조회하는 쿼리
- 결괏값: 신영식

- UNION은 합집합이므로 중복되는 부분은 1번만 출력한다.
- 최종 결괏값은 임꺽정, 장길산, 신영식이다.

② UNION ALL

UNION ALL은 UNION과 유사하나 중복된 항목을 포함하여 결과를 반환한다.

학번	성명	학년	수강과목
1001	홍길동	1	수학
1004	임꺽정	2	계산이론
1007	장길산	2	알고리즘
1008	신영식	3	수학

〈학생 테이블〉

```
SELECT 성명
  FROM 학생
 WHERE 학년 >= 2
 UNION ALL
SELECT 성명
  FROM 학생
 WHERE 학년 >= 3;
```

- 2학년 이상인 학생들의 성명을 조회하는 쿼리
- 결괏값: 임꺽정, 장길산, 신영식
- 3학년 이상인 학생들의 성명을 조회하는 쿼리
- 결괏값: 신영식

- UNION ALL은 모든 결괏값을 그대로 출력한다.
- 최종 결괏값은 임꺽정, 장길산, 신영식, 신영식(신영식은 UNION ALL 앞의 쿼리에서 한 번, 뒤의 쿼리에서 한 번, 총 2번 조회되므로 최종 조회도 2번)이다.

핵인싸 퀴즈

55 ( )은/는 프로그램 코드상에 존재하는 것으로 비정상적인 프로그램과 정상적인 프로그램 버전 간의 차이로 인하여 발생한다.

56 ( )은/는 정상적인 프로그램과 비정상적인 프로그램의 실행 결과의 차이를 의미한다.

57 ( )은/는 버그, 에러, 오류, 실패, 프로그램 실행에 대한 문제점, 프로그램 개선 사항 등의 전체를 포괄하는 용어이다.

정답 55. 오류(Fault) 56. 실패(Failure) 57. 결함(Defect)

핵인싸 퀴즈

58 ( )은/는 코드 인스펙션 외에도 설계 및 설계 산출물까지 포괄하는 테스트 방법으로 워크스루와 같이 몇 시간 동안 수행되는 단위 미팅과는 구별되며, 며칠 동안의 수행 기간이 필요하고 도구의 도움이 있어야 한다.

59 ( )은/는 소프트웨어 프로세스 전반에 걸쳐 소프트웨어 형상의 변경 요인에 대한 관리를 통해 소프트웨어 형상을 보호하는 활동이다.

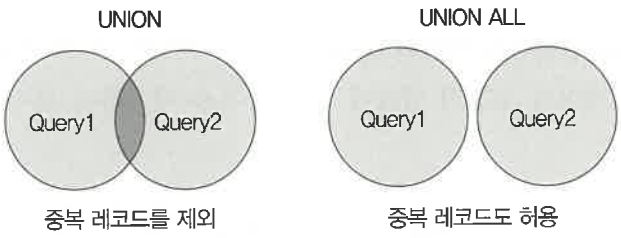
60 ( )은/는 시스템 생명 주기 일정 시점마다, 산출물을 검토하고, 그 결과를 반영하여 다음 단계 이전할 때의 시점과 산출물이다.

61 ( )의 주요 기능은 형상을 식별하고 관리하는 데 있고, 형상 식별, 버전 관리, 변경 통제, 형상 감사, 상태 보고 등의 활동을 한다.

62 ( )은/는 소프트웨어 형상 항목에 대해 식별하고 고유한 이름을 부여하는 활동으로, 식별자 항목을 정의하고 세부 내용을 파악하여 식별한다.

63 버전 관리 도구유형 중 ( ) 방식은 매일 개발 완료 파일은 약속된 위치의 공유 폴더에 복사하는 방식으로 담당자 한 명이 매일 공유 폴더의 파일을 자기 PC로 복사하고 컴파일하여 에러 확인과 정상 동작 여부를 확인한다.

정답 58. 소프트웨어 인스펙션 59. 소프트웨어 형상 관리 60. 기준선(Baseline) 61. 형상 관리 62. 형상 식별 63. 공유 폴더



▲ UNION / UNION ALL 결과 비교

③ INTERSECT

INTERSECT는 교집합을 추출한다.

학번	성명	학년	수강과목
1001	홍길동	1	수학
1004	임꺽정	2	계산이론
1007	장길산	2	알고리즘
1008	신영식	3	수학

〈학생 테이블〉

```
SELECT 성명
  FROM 학생
 WHERE 학년 >= 2
 INTERSECT
SELECT 성명
  FROM 학생
 WHERE 학년 >= 3;
```

- 2학년 이상인 학생들의 성명을 조회하는 쿼리
- 결괏값: 임꺽정, 장길산, 신영식
- 3학년 이상인 학생들의 성명을 조회하는 쿼리
- 결괏값: 신영식

• 최종 결괏값은 신영식(INTERSECT 키워드 앞뒤 쿼리에서 동시에 나오는 학생은 신영식 한 명)이다.

④ MINUS

- MINUS는 차집합을 의미한다.
- 데이터 집합을 기준으로 다른 데이터 집합과 공통 항목을 제외한 결과만 추출한다.

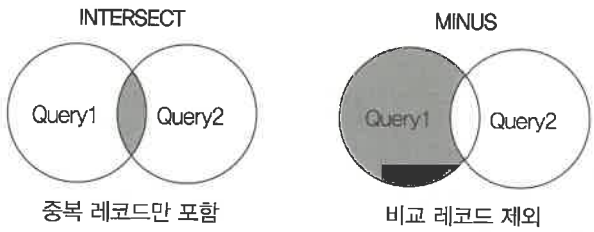
학번	성명	학년	수강과목
1001	홍길동	1	수학
1004	임꺽정	2	계산이론
1007	장길산	2	알고리즘
1008	신영식	3	수학

〈학생 테이블〉

```
SELECT 성명
  FROM 학생
 WHERE 학년 >= 2
 MINUS
SELECT 성명
  FROM 학생
 WHERE 학년 >= 3;
```

- 2학년 이상인 학생들의 성명을 조회하는 쿼리
- 결괏값: 임꺽정, 장길산, 신영식
- 3학년 이상인 학생들의 성명을 조회하는 쿼리
- 결괏값: 신영식

• 최종 결괏값은 임꺽정, 장길산(MINUS 키워드 앞의 쿼리에서 조회되지만 MINUS 키워드 뒤의 쿼리에서 조회되지 않는 학생만 출력)이다.



▲ INTERSECT / MINUS 결과 비교

핵인싸 퀴즈

64 버전 관리 도구유형 중 ( ) 방식은 로컬 저장소와 원격 저장소로 분리된 구조로 개발 완료한 파일을 수정한 다음에 로컬 저장소에 우선적으로 커밋(Commit)한 이후, 다시 원격 저장소에 반영(Push)하는 방식이다.

65 형상 관리 도구의 기능 중 ( )은/는 형상 관리 저장소로부터 최신 버전을 개발자 PC로 다운로드 받는 기능이다.

66 형상 관리 도구의 기능 중 ( )은/는 개발자가 수정한 소스를 형상 관리 저장소로 업로드하는 기능이다.

67 ( )은/는 가장 오래된 형상 관리 도구 중의 하나로서 중앙 집중형 서버 저장소를 두고 클라이언트가 접속해서 버전 관리를 실행하고, 주요 기능으로 check out/check in, update/commit이 있다.

68 ( )은/는 같은 중앙 집중형 클라이언트-서버 방식이나, CVS의 단점을 보완해 가장 널리 사용되고 있는 형상 관리 도구이다.

69 ( )은/는 리누스 토발즈가 2005년 리눅스 커널의 개발을 위해 만든 형상 관리 시스템으로 중앙 집중형 방식이 아닌 분산형 방식으로 각 PC 스스로 완전한 저장소가 구성된다.

정답 64. 분산 저장소 65. 체크아웃(Check-out) 66. 체크인(Check-in) 67. CVS(Concurrent Versions System) 68. SVN(Subversion) 69. Git