<1과목>

***현행 시스템 파악 개념**

- 현행시스템이 어떤 하위시스템으로 구성되어있고, 제공 기능 및 연계정보는 무엇이며 어떤 기술요소를 사용하는지를 파악하는 활동

***현행시스템 파악 절차**

구성/기능/인터페이스 파악 → 아키텍처 및 소프트웨어 구성 파악 → 하드웨어 및 네트워크 구성 파악

***소프트웨어 아키텍처**

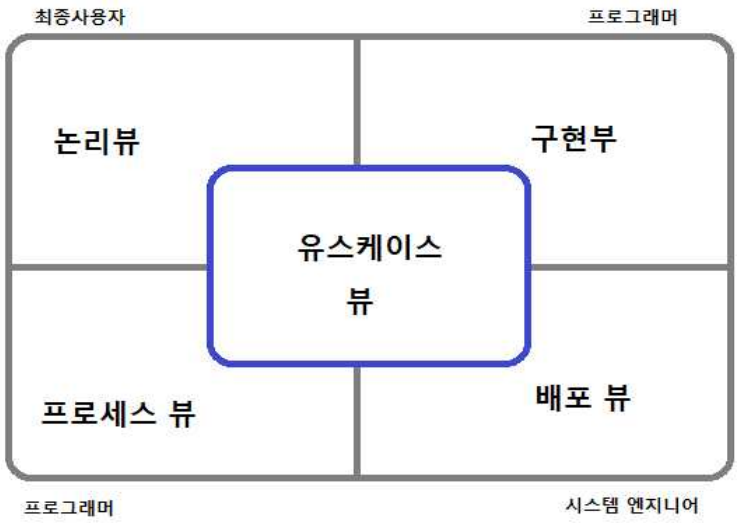
- 여러 가지 소프트웨어 구성요소와 그 구성요소가 가진 특성 중에서 외부에 드러나는 특성, 그리고 구성요 소간의 관계를 표현하는 시스템의 구조나 구조체
- 소프트웨어를 설계하고 전개하기 위한 지침이나 원칙

***소프트웨어 아키텍처 프레임워크 구성요소**

- 아키텍처 명세서
- 이해관계자
- 관심사
- 관점
- 뷰

***소프트웨어 아키텍처 4+1뷰**

- 고객의 요구사항을 정리해 놓은 시나리오를 4개의 관점에서 바라보는 소프트웨어적인 접근 방법 → 유스케이스 사용함



***운영체제의 개념**

- 컴퓨터 시스템이 제공하는 모든 하드웨어, 소프트웨어를 사용할 수 있도록 해주고, 컴퓨터 사용자와 컴퓨터 하드웨어 간의 인터페이스를 담당하는 프로그램

***OSI 7계층 (아파서 티내다, 피나다)**

계층	설명	프로토콜	전송단위
응용계층 Application Layer	사용자와 네트워크 간 응용서비스 연결, 데이터 생성	HTTP FTP	데이터 (Data)
표현계층 Presentation Layer	데이터 형식 설정과 부호교환, 암/복호화	JPEG MPEG	
세션 계층 Session Layer	연결 접속 및 동기제어	SSH TLS	
전송계층 Transport Layer	신뢰성 있는 통신 보장 데이터 분할과 재조립, 흐름 제어, 오류 제어, 혼잡 제어 등을 담당	TCP UDP	세그먼트 (Segment)
네트워크계층 Network Layer	단말 간 데이터 전송을 위한 최적화된 경로 제공	IP ICMP	패킷 (Packet)
데이터 링크 계층 Data Link Layer	인접 시스템 감 데이터 전송, 전송오류 제어 동기화, 흐름 제어 등의 전송 기능 제공 오류검출/ 재전송 등 기능 제공	이더넷	프레임 (Frame)
물리 계층 Physical Layer	0과 1의 비트 정보를 회선에 보내기 위한 전기적 신호 변환	RS-232C	비트 (Bit)

***DBMS(Database Management System)**

- 데이터베이스라는 데이터의 집합을 만들고, 저장 및 관리할 수 있는 기능들을 제공하는 응용 프로그램
- 기능 : 중복제어, 접근 통제, 인터페이스 제공, 관계 표현, 시딩/파티셔닝/ 무결성 계약조건, 백업 및 회복

***JDBC(Java Database Connectivity)**

- 자바에서 데이터베이스를 사용할 수 있도록 연결해주는 응용 프로그램 인터페이스
- SQL을 사용하여 DBMS에 질의하고 데이터를 조작하는 API 제공

***ODBC(Open Database Connectivity)**

- 데이터베이스를 액세스하기 위한 표준 개방형 응용 프로그램 인터페이스

***미들웨어(Middleware)**

- 분산 컴퓨팅 환경에서 응용 프로그램과 프로그램이 운영되는 환경 간에 원만한 통신이 이루어질 수 있도록 제어해주는 소프트웨어
- OS와 SW사이에 위치
- ex) WAS

***웹 애플리케이션 서버(WAS)**

- 웹 애플리케이션 서버는 서버계층에서 애플리케이션이 동작할 수 있는 환경을 제공하고 안정적인 트랜잭션 처리와 관리, 다른 기종 시스템과의 애플리케이션 연동을 지원하는 서버

***요구사항**

- 문제의 해결 또는 목적 달성을 위하여 고객에 의해 요구되거나, 표준이나 명세 등을 만족하기 위하여 시스템이 가져야 하는 서비스 또는 제약사항

***요구사항 분류**

구분	기능적 요구사항	비기능적 요구사항
개념	시스템이 제공하는 기능, 서비스에 대한 요구사항	시스템이 수행하는 기능 이외의 사항, 시스템 구축에 대한 제약사항에 관한 요구사항
도출 방법	특정 입력/상황에 대해 시스템이 어떻게 반응/동작해야 하는지에 대한 기술	품질 속성에 관련하여 시스템이 갖춰야 할 사항에 대한 기술 시스템이 준수해야 할 제약 조건에 대한 기술
특성	**기능성, 완전성, 일관성**	**신뢰성, 사용성, 효율성, 유지보수성, 이식성**
사례	최종주문 완료되면 배송추적 가능해야 함	특정 함수의 호출시간은 3초가 넘지 않아야 한다

*요구사항 개발 프로세스

1) 요구사항 도출
2) 요구사항 분석
3) 요구사항 명세
4) 요구사항 확인

동료검토	- 요구사항 명세서 작성자가 요구사항 명세서를 설명하고 이해관계자들이 설명을 들으면서 결함을 발견하는 형태로 진행
워크스루	검토 자료를 회의 전에 배포해서 사전검토한 후 짧은 시간 동안 회의를 진행하는 형태, 리뷰로 오류검출 문서화
인스펙션	소프트웨어 요구, 설계, 원시 코드 등의 저작자 외의 다른 전문가 또는 팀이 검사하여 오류를 찾아내는 공식적 검토방법

***요구사항 관리 프로세스**

1) 요구사항 협상
2) 요구사항 기준선
3) 요구사항 변경 관리
4) 요구사항 확인 및 검증

***CCB(Configuration Control Board; 형상 통제 위원회)**

형상 항목에 대한 형상 베이스라인이 승인된 후, 발생되는 형상 항목의 변경에 대하여 평가, 조정, 승인/보류/기각을 결정하는 심의 조직

***요구사항 분석기법**

- 요구사항 분류, 개념 모델링, 요구사항 할당, 요구사항 협상, 정형 분석

***요구사항 확인 기법**

- 요구사항 검토, 프로토타이핑, 모델 검증, 인수 테스트

***프로토타이핑**

- 사용자가 요구한 주요 기능을 Prototype으로 구현하여, 사용자의 피드백을 통해 개선, 보완하여 완성 소프트웨어를 만들어가는 기법

***UML**

- 객체지향 소프트웨어 개발과정에서 산출물을 명세화, 시각화, 문서로 만들 시 사용되는 모델링 기술과 방법론을 통합해 만든 표준화된 범용 모델링 언어

***요구사항 확인 프로세스**

1) 요구사항 목록 확인
2)요구사항 정의서 작성 여부 확인

3) 비기능적 요구사항의 확인
4) 타 시스템 연계 및 인터페이스 요구사항 확인

***비용 산정 모델**

: 소프트웨어 규모 파악을 통한 투입자원, 소요 시간을 파악하여 실행 가능한 계획을 수립하기 위해 비용을 산정하는 기법

분류	설명	종류
하향식 산정방법	경험 많은 전문가에게 비용 산정을 의뢰하거나 여러 전문가와 조정자를 통해 산정하는 방식	- 전문가 판단 - 델파이 기법
상향식 산정방법	세부적인 요구사항과 기능에 따라 필요한 비용을 계산하는 방식	- 코드 라인 수(Loc) - Man Month - COCOMO 모형 - Putam 모형 - FP 모형

*** 예제**

예) Loc=500,000 / 한 프로그래머 한달에 25,000라인 개발 Man month는?
-> 500,00 / 25,000 = 50개월
예) Man month가 50개월일 때 10명이 프로젝트를 수행한다면 프로젝트 총 기간은?
-> 50/10=5개월

하향식 비용산정 모델	전문가 판단	- 조직 내에 있는 경험이 많은 두 명 이상의 전문가에게 비용산정을 의뢰하는 기법
	델파이 기법	- 전문가의 경험적 지식을 통한 문제해결 및 미래 예측을 위한 기법
상향식 비용산정 모델	LoC	- 소프트웨어 각 기능의 원시 코드 라인 수의 비관치, 낙관치, 기대치를 측정하여 예측치를 구하고 이를 이용하여 비용 산정 => 이해 쉽고 측정 쉬움
	Man Month	한사람이 1개월동안 할 수 있는 양을 기준으로 프로젝트 비용을 산정하는 기법
	COCOMO	- Construcive cost Model의 약자 - 보헴이 제안, 프로그램 규모에 따라 비용을 산정 - 개발 노력 승수를 결정

<2과목>

***논리데이터 모델링 (개체/속성/관계로 구성)**

- 데이터베이스 설계프로세스의 기초 설계 단계로 비즈니스 정보의 구조와 규칙을 명확하게 표현할 수 있는 기법

***논리데이터 모델링 특징**

- 정규화, 포용성, 완전성, 독립성

***계체-관계(E-R)모델**

- 현실 세계에 존재하는 데이터와 그들 간의 관계를 사람이 이해할 수 있는 형태로 명확하게 표현하기 위해서 가장 널리 사용되고 있는 모델
- 요구사항으로부터 얻어낸 정보들을 개체, 속성, 관계로 기술한 모델
- 개체 - 사각형
- 관계 - 마름모
- 속성 - 타원
- 다중 값 속성 - 이중타원
- 관계-속성 연결 - 선

***정규화**

: 관계형 데이터베이스 설계에서 중복을 최소화하여 데이터를 구조화하는 프로세스

***정규화 과정 (두부이걸다조)**

단계	조건
1NF	도메인이 원자 값으로 구성
2NF	부분 함수 종속 제거
3NF	이행 함수 종속 제거
BCNF	결정자 함수이면서 후보키가 아닌 것 제거
4NF	다치(다중 값) 종속성 제거
5NF	조인 종속성 제거

이상 현상 유형	설명
삽입 이상	정보 저장 시 해당 정보의 불필요한 세부정보를 입력해야 하는 경우
삭제이상	정보 삭제 시 원치 않는 다른 정보가 같이 삭제되는 경우
갱신 이상	중복 데이터 중에서 특정 부분만 수정되어 중복된 값이 모순을 일으키는 경우

***물리데이터 모델링**

- 논리 모델을 적용하고자 하는 기술에 맞도록 상세화해가는 과정

***물리데이터 모델링 변환 절차**

1) 개체를 테이블로 변환
2) 속성을 컬럼으로 변환
3) UID를 기본키로 변환
4) 관계를 외래키로 변환
5) 컬럼 유형과 같이 정의

***참조 무결성 계약 조건**

- 두 개의 릴레이션이 기본키, 외래키를 통해 참조 관계를 형성할 경우, 참조하는 외래키의 값은 항상 참조되는 릴레이션의 기본키로 존재해야함

***튜플(가로)** :행 / **카디널리티(세로)** : 열

***파티션의 종류**

- 레인지 파티셔닝 : 연속적 숫자/날짜 기준 - 쉬운 관리 시간 단축
- 해시 파티셔닝 : 해시 함수 값에 의한 파티셔닝 - 균등분할 가능 성능향상
- 리스트 파티셔닝 : 명시적 제어 가능 - 데이터 많으면 유리
- 컴포지드 파티셔닝 : 해시함수 적용하여 재분할

***파티션의 장점**

- 성능향상 / 가용성 향상/ 백업 기능/ 경합 감소

***프로시저(Procedure)**

- SQL을 이용해 생성된 데이터를 조작하는 프로그램
- 데이터베이스 내부에 저장되고 일정한 조건이 되면 자동으로 수행

***프로시저 작성/문법**

CREATE OF REPLACE PRECEDURE

Procedure 명
파라미터 1 데이터타입 [IN | OUT | INOUT]

IS[AS]

선언부

BEGIN

실행부

EXPECTION

예외 처리부

END;

***DBMS_OUPUT 패키지**

- 메시지를 버퍼에 저장하고 버퍼로부터 메시지를 읽어오기 위한 인터페이스를 제공하는 패키지하나의 프로시저, 함수, 트리거 등에 의해 저장된 메시지는 다른 프로시저, 함수, 트리거 등
- 에서 읽어올 수 있다.

***옵티마이저**

- SQL을 가장 빠르고 효율적으로 수행할 최적의 처리경로를 생성해주는 DBMS 내부의 핵심 엔진 (처리계획 => 실행 계획)
- 규칙기반 옵티마이저/ 비용 기반 옵티마이저

<3과목>

***연계 시스템**

- 송신 시스템 / 수신 시스템 / 증계 서버

***연계 요구사항 주요 분석기법**

- 인터뷰 / 체크리스트 / 브레인스토밍 / 설문지/ 델파이 기법

***델파이 기법**
- 전문가들의 의견 수립, 중재, 타협의 방식으로 반복저인 피드백을 통한 하향식 의견 도출 방법으로 문제를 해결하는 기법

***WSDL**
- 웹 서비스명, 제공 위치, 메시지 포맷, 프로토콜 정보 등 웹 서비스에 대한 상세 정보를 기술한 파일

***인스턴스**
- 객체지향 프로그래밍에서 해당 클래스의 구조로 컴퓨터에 저장 공간에서 할당된 실체

***SOAP**
- HTTP, HTTPS, SMTP 등을 사용하여 XML기반의 메시지를 네트워크 상태에서 교환하는 프로토콜

***UDDI**
- 웹서비스에 대한 정보인 WSDL을 등록하고 검색하기 위한 저장소로 공개적으로 접근, 검색이 가능한 레지스트리

***JSON**
- 비동기 브라우저 / 서버 통신(AJAX)을 위해 “속성-값 쌍”,“키-값 쌍”으로 이루어진 데이터 오브젝트를 전달하기 위해 인간이 읽을 수 있는 텍스트를 사용하는 개방형 표준 포맷

***EAI(Enterprise Application Integreation)**
- 기업에서 운영되는 서로 다른 플랫폼 및 애플리케이션들 간을 정보 전달, 연계, 통합을 가능하게 해주는 솔루션
- 연대성 증대 -> 효율 높임 -> 시스템간 확장성 높여줌

***ESB(Enterprise Service Bus)**
- 기업에서 운영되는 서로 다른 플랫폼 및 애플리케이션들 간을 하나의 시스템으로 관리 운영할 수 있도록 서비스 중심의 통합을 지향하는 아키텍처 또는 기술

- 버스를 중심으로 각각 프로토콜이 호환이 가능 -> 느슨한 결합

***REST(Representational State Transfer)**

- HTTP URL을 통해 자원을 명시하고, HTTP Method를 통해 해당 자원에 대한 생성, 조회, 갱신, 삭제 등의 명령을 적용하는 기술

<4과목>

***개발환경 구축**

- 개발환경 구성 시 구현될 시스템 요구사하으이 명확한 이해가 필요
- 도구와 서버의 선정이 이루어져야 하고, 도구들의 사용 편의성과 라이선스 확인

***개발 도구 분류**

빌드 도구	작성한 코드의 빌드 및 배포를 수행하는 도구 각각의 구성요소와 모듈에 대한 의존성 관리를 지원
구현 도구	개발자의 코드 작성과 디버깅, 수정 등과 같은 작업을 지원하는 도구 프로그램 개발할 때 가장 많이 사용되는 도구
테스트 도구	코드의 기능 검증과 전체의 품질을 높이기 위해 사용되는 도구 코드 테스트/ 테스트 계획/ 수행 및 분석 등의 작업 가능
형상 관리 도구	개발자들이 작성한 코드와 리소스 등 산출물에 대한 버전 관리를 위한 도구 프로젝트 진행 시 필수로 포함되는 도구

***서버 하드웨어 개발환경**

- 웹 서버
- 웹 애플리케이션 서버(WAS)
- 데이터베이스 서버
- 파일 서버

***클라이언트 하드웨어 개발환경**

- 클라이언트 프로그램 : 설치-사용자와 커뮤니티
- 웹 프라우저
- 모바일 앱
- 모바일 웹

***소프트웨어 개발환경**

- 운영체제 - Windows, Unix, Linux ...
- 미들웨어 - Tomcat, Weblogic, Jeus...
- DBMS - Oracle, MySQL ..
- .

***JVM(Java Virtual Machine)**
- 시스템 메모리를 관리함녀서 자바 기반 애플리케이션을 위해 이식 가능한 실행환경을 제공하는 소프트웨어

***컨테이너**
- JSP와 서블릿을 실행시킬 수 있는 소프트웨어

***형상 관리**
- 소프트웨어 개발을 위한 전체 과정에서 발생하는 모든 항목의 변경 사항을 관리하기 위한 활동
- 절차 : 형상 식별 → 형상 통제 → 형상 감사 → 형상 기록

***JDK**
- 자바 애플리케이션을 구축하기 위한 핵심 플랫폼

***브랜치**
- 여러 개발자들이 동시에 다양한 작업을 할 수 있게 만들어 주는 기능
- 각자 독립적인 작업 영역(저장소) 안에서 마음대로 소스 코드를 변경할 수 있는 기능

***모듈**
- 크게 독립된 하나의 소프트웨어 또는 하드웨어 단위를 지칭하는 용어
- 모듈의 독립성을 높이려면 : 결합도는 낮게, 응집도는 강하게, 모듈의 크기는 작게

***모듈화 기법**
- 루틴
- 메인 루틴
- 서브 루틴
-

***응집도(Cohesion)**
- 모듈의 독립성을 나타내는 개념으로 모듈 내부 구성요소 간 연관 정도

- 정보 은닉 확장 개념 -> 하나의 모듈은 하나의 기능을 수행

***응집도의 유형 (밑으로 갈수록 좋은 품질)**
- 우연적 응집도
- 논리적 응집도
- 시간적 응집도
- 절차적 응집도
- 통신적 응집도
- 순차적 응집도
- 기능적 응집도

=> 응집도가 높을수록 품질이 좋아진다.

***결합도(Coupling)**
- 외부 모듈과의 연관도, 모듈 간의 상호 의존성
- SW 구조에서 모듈 간의 관련성을 측정하는 척도

***결합도의 유형**
- 내용 결합도
- 공통 결합도
- 외부 결합도
- 제어 결합도
- 스탬프 결합도
- 자료 결합도

***MVC(Model, View, Controller) 패턴 역할**
- 모델 : APP이 무엇을 하지 정의, 내부 비즈니스 로직 처리하기 위한 역할
- 뷰 : 화면에 무엇인가를 보여주기 위한 역할
- 컨트롤러 : 모델이 어떻게 처리할지를 알려주는 역할

***DAO(Data Access Object)**
- 특정 타입의 데이터베이스에 추상 인터페이스를 제공하는 객체로 세부내용 노출 없이 데이터를 조작하는 객체

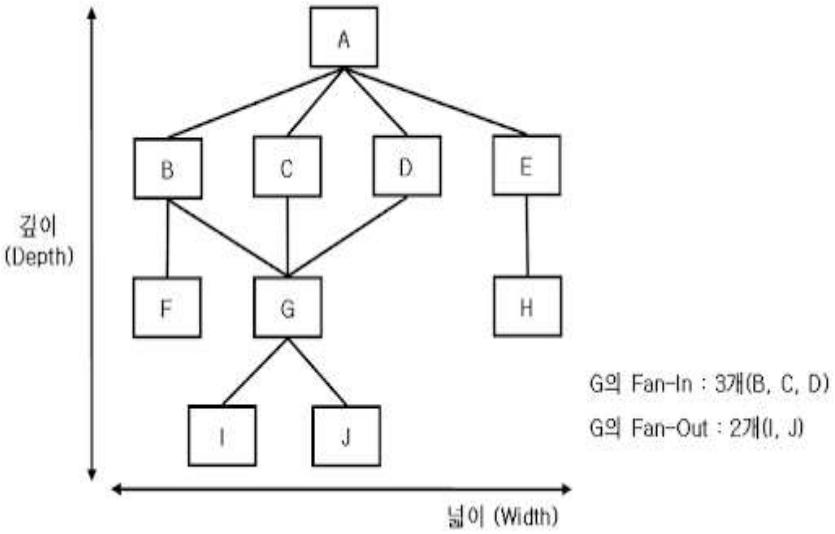
***VO(Value Object)**
- 간단한 엔티티를 의미하는 작은 객체 가변 클래스인 DTO와 달리 고정 클래스를 가지는 객

체

***DTO(Data Transfer Object)**

- 프로세스 사이에서 데이터를 전송하는 객체로 데이터 저장, 회수 외에 다른 기능이 없는 객체

*** 팬인(Fan-in) & 팬아웃(Fan-Out)** - SW 구성요소인 모듈을 계층적으로 분석하기 위해 활용. 시스템 복잡도 측정 가능	
팬인	**팬아웃**
어떤 모듈을 제어하는 모듈의 수	어떤 모듈에 의해 제어 되는 모듈의 수
- 재사용 측면 설계 잘됨/ 단일 장애점 발생 가능 / 관리 비용, 테스트 비용 비쌈	- 불필요한 모듈 호출 여부 검토 필요, 단순화 여부 검토 필요



***JUnit**

- 자바 프로그래밍 언어용 단위 테스트 도구
- 테스트 코드를 쉽게 작성하고 자동화하기 위해 사용

***루틴(Routine)**

- 소프트웨어에서 특정 동작을 수행하는 일련의 코드로서 기능을 가진 명령어들의 모임

***통합 개발환경 IDE(Integrated Development Enviroment)**

- 코딩, 디버그, 컴파일, 배포 등 프로그램 개발에 관련된 모든 작업을 하나의 프로그램 안에서 처리하는 환경을 제공하는 소프트웨어
- ex) 이클립스, Visual Studio

***화이트박스 테스트**

- 응용 프로그램의 내부 구조와 동작을 검사하는 소프트웨어 테스트 방식

***어노테이션(Annotation)**

- 자바코드에 주석을 달아 특별한 의미를 부여한 메타데이터의 일종 /보통 앞에@붙여서 사용

***프론트 앤드와 백 앤드**

프론트 앤드	백 앤드
- 사용자의 화면에 나타나는 웹 화면 영역으로 웹 페이지를 그리는 기술 - JSP, HTML, CSS ..	- 사용자와 만나지 않고 프론트앤드와 연동하여 핵심 로직을 처리하는 영역으로 DB나 인터페이스를 통해 시스템에 접근함

***MyBatis**

- 객체지향 언어인 자바의 관계형 DB프로그래밍을 좀 더 쉽게 할 수 있게 도와주는 개발 프레임 워크

***배치프로그램**

- 사용자와의 상호작용 없이 일련의 작업들을 작업 단위로 묶어 정기적으로 반복 수행하거나 정해진 규칙에 따라 일괄 처리하는 방법
- 필수 요소 : 이벤트 배치/ 온디맨드 배치/ 정기 배치

***배치 스케줄러 종류**

스프링 배치	스프링 프레임 워크의 DI, AOP, 서비스 추상화 등 스프링 프레임워크의 3대 요소를 모두 사용할 수 있는 대용량 처리를 제공하는 스케줄러
쿼츠 스케줄러	스프링 프레임워크에 플러그인되어 수행하는 작업과 실행 스케줄을 정의하는 트리거를 분리하여 유연성을 제공하는 오픈 소스 기반 스케줄러

***Cron 표현식**

1	초	0~59, 특수문자
2	분	0~59, 특수문자
3	시간	0~23 , 특수문자
4	일	0~31, 특수문자
5	월	0~12, JAN~DEC, 특수문자
6	요일	1~7, SUN-SAT, 특수문자
7	연도 (생략가능)	1970~2099, 특수문자

기호	의미
*	모든 수
?	해당 항목을 미사용
-	기간 설정
,	특정 기간 설정
/	시작 기간과 반복 간격 설정
L	마지막 기간에 동작
W	가장 가까운 평일에 동작
#	몇 번째 주, 요일 설정

- 0012*** : 매일 12시에 실행
- */10**** : 매 10분 마다 실행

<5과목>

***인터페이스 설계서**

- 다른 기종 시스템 및 컴포넌트 간 데이터 교환 및 처리를 위해 각 시스템의 교환되는 데이터, 업무, 송수신 주체 등이 정의된 문서 -> 인터페이스 정보를 정의한 문서

***FTP(File Transfer Protocol)**

- TCP/IP 프로토콜을 가지고 서버와 클라이언트 사이의 파일을 전송하기 위한 프로토콜

***소켓**

- 서버는 통신을 위한 소켓을 생성하여 포트를 할당하고 클라이언트 통신 요청 시 클라이언트와 연결하고 통신하는 방식

***EAI** [포허메하] - Point to Point, Hub & Spoke, Message Bus, Hybrid

***ESB**- 느슨한 결합

***컴포넌트**

- 특정한 기능을 수행하기 위해 독립적으로 개발되어 보급되는, 잘정의된 인터페이스를 가지며 다른 부품과 조립되어 응용 시스템을 구축하기 위해 사용되는 SW프로그램

***AJAX(Asynchronous Java Script and XML)**

- 자바스크립트를 사용한 비동기 통신기술로, 클라이언트와 서버 간에 XML 데이터를 주고받는 기술

***404오류**

- 웹에서 서버를 찾지 못할 때 발생하는 오류

***스니핑(Sniffing)**

- 공격 대상에게 직접 공격을 하지 않고 데이터만 몰래 들여다보는 수동적 공격기법

***시큐어 코딩 가이드 적용 대상 [입보시 에코캡아]**

- 입력데이터 검증 및 표현
- 보안 기능
- 에러 처리
- 코드 오류
- 캡슐화
- API 오용

***데이터베이스 암호화 알고리즘**

- 대칭 키 암호화 알고리즘 : 암,복호화에 같은 암호 키를 쓰는 알고리즘 / SEED ..
- 비대칭 키 암호화 알고리즘 : 공개키와 비밀키를 사용하는 알고리즘 / RSA, ECC ..
- 해시 암호화 알고리즘 : 해시값으로 원래 입력값을 찾아낼 수 없는 일방향성의 특성을 가진 알고리즘

***데이터베이스 암호화 기법**

API 방식	애플리케이션 레벨에서 암호 뮤듈을 적용하는 애플리케이션 수정방식 애플리케이션 서버에 암,복호화/ 정책 관리/ 키 관리 등의 부하 발생
Plug-in 방식	DB레벨의 확장성 프로시저 기능 이용, DBMS에 Plug-In모듈로 동작 DB서버에 암,복호화/ 정책 관리/ 키 관리 등의 부하 발생
Hybrid 방식	API+Plug-in 결합하는 방식/ DB서버와 APP서버로 부하 분산 DB서버와 애플리케이션 서버로 부하 분산

***IPSec(IP Secutity)**

- IP계층에서 무결성과 인증을 보장하는 인증헤더와 기밀성을 보장하는 암호화를 이용한 IP보안 프로토콜

***SSL/TLS**

- 응용계층과 TCP/IP 계층 사이에서 웹 데이터 암호화 및 전송 시 기밀성 보장하는 공개키 기반의 보안 프로토콜

***IKE**

- 키교환 알고리즘

***AH**

- 메시지 Checksum을 활용한 데이터 인증과 비연결형 무결성을 보장해주는 프로토콜

***ESP**

- 암호화 알고리즘을 활용한 캡슐화 기반 페이로드 기밀성을 제공하는 프로토콜

***인터페이스 구현 검증 도구**

xUnit	java, c++ 등 다양한 언어를 지원하는 단위테스트 프레임 워크 SW 함수나 클래스 같은 서로 다른 구성 원소를 테스트할 수 있게 해주는 도구
STAF	서비스 호출, 컴포넌트 재사용 등 다양한 환경을 지원하는 테스트 프레임워크
FitNesse	웹 기반 테스트케이스 설계/실행/결과 확인 등을 지원하는 테스트 프레임워크 사용자가 테스트 케이스 테이블을 작성하면 빠르고 편하게 자동으로 원하는 값에 대해 테스트를 할 수 있는 장점
Selenium	다양한 브라우저 지원 및 개발언어를 지원하는 웹 애플리케이션 테스트 프레임워크
watir	루비 기반 웹 애플리케이션 프레임 워크 모든 언어 기반의 웹 애플리케이션 테스트와 브라우저 호환성 테스팅 가능

***스카우터(SCOUTER)**

- 애플리케이션에 대한 모니터링 및 DB Agent를 통해 오픈 소스 DB모니터링 기능, 인터페이스 감시 기능 제공

<6과목 > 화면 설계

***UI(User Interface)**

- 넓은 의미에서 사용자와 시스템 사이에서 의사소통할 수 있도록 고안된 물리적, 가상의 매개체
-

***UI설계 원칙**

- 직관성 : 누구나 쉽게 이해하고, 쉽게 사용해야 한다.
- 유효성 : 정확하고 완벽하게 사용자의 목표가 달성될 수 있도록 제작
- 학습성 : 초보/숙련자 모두 쉽게 배우고 사용할 수 있게 제작
- 유연성 : 사용자의 인터션을 최대한 포용, 실수 방지

***UI 설계 지침 [사일다결 가표접명오]**

- 사용자 중심/ 일관성/ 단순성/ 경과 예측 가능/ 가시성 /표준화/ 접근성/ 명확성 /오류 발생 해결

***UX(User Experience)**

- 제품과 시스템, 서비스 등을 사용자가 직/간접적으로 경험하면서 느끼고 생각하는 총체적 경험

***레이아웃**

- 특정 공간에 여러 구성요소를 보기 좋게 효과적으로 배치하는 작업
- 구성요소 : 그리드, 버튼/컨트롤 타입, 페이지요소, 팝업 요소, 경고 요소 등 화면 구성에 필요한 요소들을 정의하는 과정

3C 분석	Customer, 경쟁 Company, 자사(Company) 비교 분석하여 자사가 어떻게 차별화해서 경재에서 이길 것인가를 분석하는 기법
SWOT 분석	기업 내/외부 환경을 분석하여 강/약점,기회 요인을 규정하고 이를 토대로 경영 전략을 수립하는 기법
시나리오 플래닝	불확실성이 높은 상황 변화를 사전에 예측하고 다양한 시나리오를 설계하는 방법으로 불확실성을 제거해 나가려는 경영 전략의 한 방법
사용성 테스트	사용자가 직접 제품 사용 -> 시나리오에 맞춰 과제수행 ->질문에 답
워크숍	소집단 인원으로 특정 문제나 과제에 대한 새로운 지식/기술/아이디어/방법들을 서로 교환하고 검토하는 연구회 및 세미나

***UI 지침**

1단계) 페르소나 정의
2단계) 콘셉트 모델 정의
3단계) 사용자 요구 사항 정의
4단계) UI컨셉션

***UI 화면 설계 분류**

구분	설명	도구
와이어 프레임	이해관계자들과의 화면구성을 협의하거나 서비스의 간략한 흐름을 공유하기 위해 화면 단위의 레이아웃을 설계하는 작업	파워포인트 키노트 스케치 일러스트
스토리보드	정책, 프로세스, 콘텐츠 구성, 와이어프레임, 기능 정의, 데이터베이스 연동 등 서비스 구축을 위한 모든 정보가 담겨 있는 설계 산출물	파워포인트 키노트 스케치

***스토리보드 작성 절차**

1) 전체 개요 작성
2) 서비스 흐름 작성
3) 스타일 확정
4) 메뉴별 화면 설계도 작성 및 상세 설계
5) 추가 관련 정보 작성

***프로토타입**

- SW의 설계 또는 성능, 구현 가능성, 운용 가능성을 평가하거나 요구사항을 좀 더 잘 이해하고 결정하기 위하여 전체적인 기능을 간략한 형태로 구현한 시제품

<7과목> 애플리케이션 테스트 관리

***소프트웨어 테스트의 개념**

개발된 응용 애플리케이션이나 시스템이 사용자가 요구하는 기능과 성능, 사용성, 안정성 등을 만족하는지 확인하고, 노출되지 않은 숨어있는 소프트웨어의 결함을 찾아내는 활동

***소프트웨어 테스트의 필요성**

- 오류 발견 관점 / 오류 예방 관점 / 품질 향상 관점

***소프트웨어 테스트 프로세스**

- 테스트 계획 → 테스트 분석 및 디자인 → 테스트 케이스 및 시나리오 작성 → 테스트 수행 → 테스트 결과 평가 및 리포팅

* **소프트웨어 테스트 원리** [결완초집 살정오]

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, 무한 입력 값으로 인한 테스트 어려움
개발 초기에 테스팅 시작	조기 테스트 설계 시 장점 : 테스팅 결과를 단시간에 알 수 있고, 테스팅 기간 단축, 재작업을 줄여 개발 기간 단축 및 결함 예방
결함 집중	적은 수의 모듈에서 대다수의 결함이 발견됨 20%모듈에서 80%의 결함이 발견됨
살충제 패러독스	동일한 테스트 케이스에 의한 반복적 테스는 새로운 버그를 찾지 못함 테스트케이스의 정기적 리뷰와 개선 및 다른 시각에서의 접근이 필요
테스팅은 정황에 의존적	소프트웨어의 성격에 맞게 테스트 실시 정황과 비즈니스 도메인에 따라 테스트를 다르게 수행
오류-부재의 궤변	요구사항을 충족시켜주지 못한다면, 결함이 없다고 해도 품질이 높다고 볼 수 없음

***소프트웨어 테스트 산출물**

테스트 계획서	- 테스트의 수행을 계획한 문서
테스트 케이스	- 테스트를 위한 설계 산출물로 응용 sw가 사용자의 요구사항을 준수하는지 확인하기 위해 설계된 입력값, 실행 조건, 기대 결과로 구성된 테스트 항목의 명세서
테스트 시나리오	- 테스트 수행을 위한 여러 개의 테스트 케이스의 집합 - 테스트 절차를 명세한 문서
테스트 결과서	- 테스트 결과럴 정리한 문서, 결과 평가하고 리포팅

***소프트웨어 테스트 - 프로그램 실행 여부에 따른 분류**

분류	설명	유형
정적 테스트	프로그램의 실행 없이 구조를 분석하여 논리성을 검증하는 테스트	동료검토, 워크스루, 인스펙션
동적테스트	프로그램 실행을 요구하는 테스트	화이트박스 테스트 블랙박스 테스트

***화이트박스 테스트**

- 프로그램 내부 로직을 보면서 수행하는 테스트
- 유형 : 제어구조 테스트/ 루프 테스트

***블랙박스 테스트**

- 프로그램 외부 사용자의 요구사항 명세를 보면서 수행하는 테스트
- 유형 : [동경결상 유분페] (사례 수제비 7-6)

유형	설명
동등 분할 테스트	입력 데이터의 영역을 유사한 도메인별로 유효 값/ 무효 값을 그룹핑하여 대푯값 테스트 케이스를 도출하여 테스트하는 기법
경계 값 분석 테스트	동기분할 후 경계 값 부분에서 오류 발생 확률이 높기에 경계 값을 포함하여 테스트 케이스를 설계하여 테스트하는 기법
결정 테이블 테스트	요구사항의 논리와 발생조건을 테이블 형태로 나열하여, 조건과 행위를 모두 조합하여 테스트하는 기법
상태전에 테스트	테스트 대상/시스템이나 객체의 상태를 구분하고, 이벤트에 의해 어느 한 상태에서 다른 상태로 전이되는 경우의 수를 수행하는 테스트 기법
유스케이스 테스트	시스템이 실제 사용되는 유스케이스로 모델링 되어 있을 때 프로세스 흐름을 기반으로 테스트 케이스를 명세화하여 수행하는 테스트 기법
분류트리 테스트	SW의 일부 또는 전체를 트리 구조로 분석 및 표현하여 테스트 케이스를 설계하여 테스트하는 기법
페어와이즈 테스트	Test Data 값들 간에 최소한 한 번 씩을 조합하는 방식이며, 이는 커버해야 할 기능적 범위를 모든 조합에 비해 상대적으로 적은 양의 테스트 세트를 구성하기 위한 테스트 기법

***테스트 시각에 따른 분류**

검증 (Verification)	소프트웨어 과정을 테스트 올바른 제품을 생산하고 있는지 검증 전단계에서 설정된 개발 규격과 욕구를 충족시키는지 판단
확인 (Validation)	소프트웨어 결과를 테스트 만들어진 제품이 제대로 동작하는지 확인 최종 사용자 요구 또는 소프트웨어 요구에 적합한지 판단

***테스트 목적에 따른 분류**

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

***테스트 종류에 따른 분류**

- 명세 기반 테스트 / 구조 기반 테스트/ 경험 기반 테스트

***애플리케이션 성능지표**

처리량 Troughput	애플리케이션이 주어진 시간에 처리할 수 있는 트랜잭션의 수
응답 시간 Response Time	사용자 입력이 끝난 후, 애플리케이션의 응답 출력이 개시될 때 까지의 시간
경과 시간 Turnaround Time	애플리케이션에 사용자가 요구를 입력한 시점부터 트랜잭션을 처리 후 그 결과의 출력이 완료할 때까지 걸리는 시간
자원 사용률 Resourse Usage	애플리케이션이 트랜잭션을 처리하는 동안 사용하는 CPU 사용량, 메모리 사용량, 네트워크 사용량

***LDAP(Lightweight Directory Access Protocol)**

- TCP/IP 위에서 조직화되고 비슷한 특성을 가진 객체들의 모임인 디렉터리 서비스를 조회하고 수정하는 응용 프로토콜

***TPS(Transaction Per Second)**

- 초당 처리 건수를 의미, 초당 몇 개의 트랜잭션을 처리할 수 있는지 나타내는 서비스 성능 지표

***Ramp-up load**

- 한계점의 측정을 목적으로 낮은 수준의 부하부터 높은 수준의 부하까지 예상 트래픽을 꾸준히 증가시키며 진행하는 부하 테스트

***클린코드**

- 가독성이 높고, 단순, 의존성 낮고, 중복을 최소화하여 깔끔하게 잘 만들어진 코드
- 유형 : 의미 있는 이름/ 간결하고 명확한 주석/ 보기 좋은 배치/ 작은 함수 / 읽기 쉬운 제어 흐름 / 오류 처리

***테스트 시나리오**

- 테스트 수행을 위한 여러 개의 테스트 케이스의 집합으로 테스트 케이스의 동작 순서를 기술한 문서이며, 테스트 절차를 명세한 문서

<8과목> : SQL 응용

구분	프로시저	사용자 정의함수 User-Difined Function	트리거 Trigger
개념	절차형 SQL을 활용하여 특정 기능을 수행할 수 있는 트랜잭션 언어	절차형 SQL을 활용하여 일련의 SQL처리를 수행하고, 수행 결과를 단일 값으로 반환할 수 있는 절차형 SQL	특정 테이블에 삽입, 수정, 삭제 등의 데이터 변경 이벤트가 발생하면 DBMS에서 자동적으로 실행되도록 구현하는 프로그램
구성	선언부[DECLARE] 시작/종료부[BEGIN/END 제어부[CONTROL] SQL 예외부[EXCEPTION] **실행부[TRASACTION]**	선언부[DECLARE] 시작/종료부[BEGIN/END 제어부[CONTROL] SQL 예외부[EXCEPTION] **반환부[RETURN]**	선언부[DECLARE] **이벤트부[EVENT]** 시작/종료부[BEGIN/END 제어부[CONTROL] SQL 예외부[EXCEPTION]
암기	디비컨 SET	디비컨 SER	디이비건 SE

***프로시저 SQL**

- SELECT : 데이터 조회 : 검색 명령
- INSERT : 데이터 생성 : 삽입
- UPDATE : 데이터 변경 : 수정/변경
- DELETE : 데이터 삭제 : 삭제
-

***프로시저 실행부**

- COMMIT : 성공적으로 하나의 트랜잭션이 끝나고 DB가 일관적인 상태에 있을 때 하나의 트랜잭션이 끝났을 때 사용하는 연산
- ROLLBACK : 트랜잭션이 비정상적으로 종료되어 트랜잭션 원자성이 깨질 경우 처음부터 다시 시작하거나 부분적으로 연산을 취소하는 연산

***트리거의 목적**

- 이벤트와 관련된 테이블의 데이터 삽입, 추가, 삭제 작업을 DBMS가 자동적으로 실행시키는 데 활용
- 데이터 무결성 유지/ 로그메세지 출력 등의 별도 처리를 위해 트리거 사용

***트리거 이벤트 순서**

BEFORE	- 이벤트부의 테이블에 대한 INSERT/UPDATE/DELETE를 수행하기 전에 트리거가 실행하도록 지정하는 명령
AFTER	- 이벤트부의 테이블에 대한 INSERT/UPDATE/DELETE가 성공적으로 실행되었을 때만 트리거가 실행하도록 지정하는 명령

***TRUNCATE**

- 데이터가 하나도 없이 테이블 구조만 남은 최초 테ㅣ블이 만들어진 상태로 되돌아가도록 하는 명령

***집계함수**

- 여러 행 또는 테이블 전체 행으로부터 하나의 결괏값을 반환하는 함수
- 집계함수 구문 : GROUP BY / HAVING 구문

집계 함수	내용
COUNT	줄 수
SUM	합계
AVG	평균
MAX	최댓값
MIN	최솟값
STDDEV	표준 편차
VARIAN	분산

***그룹함수**

- 테이블 전체 행을 하나 이상의 컬럼을 기준으로 컬럼 값에 따라 그룹화하여 그룹별로 결과를 출력하는 함수
- ROLLUP 함수 / CUBU / GROUPING SETS -> 8-29 한번 보기

***MyBaits**

- SQL Mapping 기반 오픈 소스 Access Framework로, DBMS에 질의하기 위한 SQL쿼리를 별도의 XML파일로 분리하고 Mapping을 통해서 SQL을 실행한다
- 장점 : JDBC 코드 단순화/ SQL그대로 사용 / Spring frame work랑 통합가능 / 우수한 성능

***데이터 제어어 DCL**

- 데이터베이스 관리자가 데이터 보안, 무결성 유지, 회복을 위해 관리자(DBA)가 사용하는 제어용 언어

***GRANT(권한 부여) 명령문**

- DBA가 사용자에게 DB에 대한 권한을 부여하는 명령어
- **GRANT** 권한 **ON** 테이블 **TO** 사용자 [WITH 권한 옵션];

***REVOKE(권한 취소) 명령어**

- DBA가 사용자에게 DB에 대한 권한을 회수하는 명령어
- REVOKE 권한 ON 테이블 FROM 사용자 [CASCADE CONSTRAINTS];

***CASCADE CONSTRAINTS** : 연쇄적인 권한 해제 명령어

<9과목> :소프트웨어 개발 보안 구축

*** SW 개발 보안**

- 소스 코드 등에 존재하느 보안 취약점을 제거하고, 보안을 고려하여 기능을 설계 및 구현하는 등 소프트웨어 개발 과정에서 지켜야 할 일련의 보안 활동

*** SW 개발 보안의 3대 요소**

기밀성 Confidentiality	- 인가되지 않은 개인 혹은 시스템 접근에 따른 정보 공개 및 노출을 차단하는 특성
가용성 Availability	- 권한을 가진 사용자나 애플리케이션이 원하는 서비스를 지속 사용할 수 있도록 보장하는 특성
무결성 Integrity	- 정당한 방법을 따르지 않고선 데이터가 변경될 수 없으며, 데이터의 정확성 및 완전성과 고의/악의로 변경되거나 훼손 또는 파괴되지 않음을 보장하는 특성

*** SW 개발 보안 용어**

자산 Assets	조직의 데이터 또는 조직의 소유자가 가치를 부여한 대상
위협 Threat	조직이나 기업이 자산에 악영향을 끼칠 수 있는 사건이나 행위
취약점 Vulnerability	위협이 발생하기 위한 사전 조건에 따른 상황
위험 Risk	위협이 취약점을 이용하여 조직의 자산 손실 피해를 가져올 가능성

*** DoS (Denial of Service) 공격**

- 시스템을 악의적으로 공격해 해당 시스템의 자원을 부족하게 하여 원래 의도된 용도로 사용하지 못하게 하는 공격
- 특정 서버에게 수많은 접속 시도를 만들어 다른 이용자가 정상적으로 서비스 이용을 하지 못하게 하거나, 서버의 TCP 연결을 소진시키는 등의 공격

지역 시스템 공격	실제 대상 시스템에 접근하여 서버하드웨어에 직접 과부하를 주는 공격
원격 네트워크 공격	- 공격자가 목표 시스템에 접근하지 않고 원격지에서 인터넷 등을 이용한 공격 - 서비스를 제공받지 못하거나 실제 시스템에 영향

*** DDoS(Distributes Dos) 공격의 개념**

- DoS의 또 다른 형태로 여러 대의 공격자를 분산 배치하여 동시에 동작하게 함으로써 특정 사이트를 공격하는 기법
- 해커들이 취약한 인터넷 시스템에 대한 액세스가 이뤄지면, 침입한 시스템에 소프트웨어를 설치하고 이를 실행시켜 원격에서 공격을 개시

*** DDos 공격 요소**

핸들러(Handler)	마스터 시스템의 역할을 수행하는 프로그램
에이전트 (Agent)	공격 대상에게 직접 공격을 가하는 시스템
마스터 (Master)	- 공격자에게서 직접 명령을 받는 시스템 - 여러 대의 에이전트를 관리하는 역할
공격자 (Attacker)	공격을 주도하는 해커의 컴퓨터
데몬(Demon) 프로그램	에이전트 시스템의 역할을 수행하는 프로그램

*** DDoS 공격 대응 방안**

- 차단 정책 업데이트
- 좀비 PC IP확보
- 보안 솔루션 운영
- 홈페이지 보안 관리
- 시스템 패치

*** DoS vs DDoS**

- DoS 는 직접 공격하고, DDoS는 공격하도록 지시

*** 자원 고갈 공격**

- 서버 간 핸드셰이크를 통해 통신이 연결되는 정상 트래픽과 달리 DoS 공격은 정상 접속을 시도하는 오픈된 소켓에 트래픽을 집중시킴

- 공격이 임계치에 도달하면 사용자들은 네트워크에 전혀 접근할 수 없게 됨

공격기법	설명
SYN 플러딩 (SUN Flooding)	- TCP 프로토콜의 구조적인 문제를 이용한 ㄱㅇ격 - 서버의 동시 가용 사용자 수를 SYN 패킷만 보내 점유하여 다른 사용자가 서버를 사용 불가능하게 하는 공격 - 공격자는 ACK를 발송하지 않고 계속 새로운 연결 요청을 하게 되어 서버는 자원할당을 해지하지 않고 자원만 소비하여 자원이 고갈
UDP 플러딩	- 대량의 UDP패킷을 만들어 임의의 포트 번호로 전송하여 응답 메시지를 생성하게 하여 지속해서 자원을 고갈시키는 공격
스머프	- 출발지 주소를 공격 대상의 IP로 설정하여 네트워크 전체에게 ICMP Eech 패킷을 직접 브로드캐스팅하여 마비시키는 공격
PoD	- 큰 사이즈의 패킷을 의도적으로 목표시스템으로 발생시켜 시스템이 서비스할 수 없는 상태로 만드는 공격

***애플리케이션 공격**

공격 기법	설명
HTTP GET 플러딩	- Cache Control Attack 공격 - Http 캐시 옵션을 조작하여 캐싱 서버가 아닌 웹서버가 직접 처리하도록 유도, 웹서버 자원을 소진시키는 서비스 거부 공격
Slowloris	- HTTP GET 메소드를 사용하여 헤더의 최종 끝을 알리는 개행 문자열인 \r \n \r \n을 전송하지 않고, \r \n만 전송하여 대상 웹서버와 연결상태를 장시간 지속시키고 연결 자원을 모두 소진시키는 서비스 거부 공격
RUDY	- 요청 헤더의 Content-length를 비정상적으로 크게 설정하여 메시지 바디 부분을 매우 소량으로 보내 계속 연결 상태를 유지시키는 공격

***네트워크 서비스 공격**

공격기법	설명
네트워크 스캐너, 스니퍼	네트워크 하드웨어 및 소프트웨어 구성의 취약점 파악을 위해 공격자가 사용하는 공격 도구
패스워드 크래킹	사전 크래킹과 무차별 크래킹 방법을 사용해 네트워크 패스워드를 탐색
IP 스푸핑	서버에 대한 인증되지 않은 액세스 권한을 입수하는데 사용하는 기법 - 침임자가 패킷 헤더 수정을 통해 인증된 호스트의 IP 어드레스를 위조 - 타깃 서버로 메시지를 발송한 이후 패킷은 해당 포트에서 유입되는 것처럼 표시
트로이 목마	악성 루틴이 숨어 있는 프로그램으로서 겉보기에는 정상적인 프로그램으로 보이지만 실행하면 악성 코드를 실행

***취약점 공격**

공격기법	설명
랜드 어택	- 출발지 IP와 목적지 IP를 같은 패킷 주소로 만들어 보냄으로써 수신자가 자기 자신에게 응답을 보내게 하여 시스템의 가용성을 침해하는 공격기법
봉크 / 보잉크	- 프로토콜의 오류 제어를 이용한 공격 기법으로서 시트메의 패킷 재전송과 재조립이 과부화를 유발 - 봉크 : 같은 시퀀스 번호를 계속 보냄 - 보잉크 : 일정한 간격으로 시퀀스 번호에 빈 공간을 생성
잉크	- 프로토콜의 오류 제어를 이용한 공격 기법으로서 시트메의 패킷 재전송과 재조립이 과부화를 유발 - 봉크 : 같은 시퀀스 번호를 계속 보냄 - 보잉크 : 일정한 간격으로 시퀀스 번호에 빈 공간을 생성지를 발송한 이후 패킷은 해당 포트에서 유입되는 것처럼 표시
티어 드롭	- IP 패킷의 재조합 과정에서 잘못된 Fragment Offset 정보로 인해 수신시스템이 문제를 발생하도록 만드는 DDoS공격 - 공격자는 IP Fragment Offset 값을 서로 중첩되도록 조작하여 전송하고, 이를 수신한 시스템이 재조합하는 과정에서 오류가 발생, 시스템의 기능을 마비시키는 공격 방식

*암호 알고리즘
 - 데이터의 무결성 및 기밀성 확보를 위해 정보를 쉽게 해독할 수 없는 형태로 반환하는 기법

*대칭 키 암호화 방식
 - 암호화 알고리즘의 한 종류로, 암호화와 복호화에 같은 암호키를 쓴느 알고리즘
 - 블록 암호화 / 스트림 암호화 알고리즘

 *비 대칭키 암호화 방식 (공개 키 암호화 방식)
  - 공개 키를 이용해 암호화하고 공개 키에 해당하는 개인 키를 이용해 복호화하는 암호 방식
  - 비대칭 키 암호 방식에서는 공개 키와 개인 키가 존재하며, 공개 키는 누구나 알 수 있지만 그에 대응하는 개인 키는 키의 소유자만이 알 수 있어야 함
 - 비밀 키는 키의 소유자만이 알 수 있어야 한다. 공개 키는 보안 타협 없이공개적으로 배포가 가능
 - RSA, 디피-헬만

*해시 방식
 - 단방향 알고리즘으로서 임의의 데이터를 고정된 길이의 데이터로 매핑하는 함수
 - 해시 함수의 결과로 원본 데이터를 유추하기 어려운 것을 이용
 - 연산에 걸리는 시간이 빠른 것이 장점이지만, 동일한 결과를 갖는 값이 발생하는 해시 충돌 문제가 발생 가능
 - SHA, MD5

*고유 식별 정보
 - 주민번호, 여권번호, 운전면허번호, 외국인등록번호

*시큐어 코딩
 - 설계 및 구현 단계에서 해킹 등의 공격을 유발할 가능성이 있는 잠재적인 보안 취약점을 사전에 제거하고, 외부 공격으로부터 안전한 소프트웨어를 개발하는 기법

*시큐어 코딩 가이드
 - 입력 데이터 검증 및 표현
 - 보안 기능
 - 시간 및 상태

- 에러 처리
- 코드 오류
- 캡슐화
- API 오용

*SQL 삽입 공격
- 웹 애플리케이션에서 입력데이터에 대한 유효성 검증을 하지 않을 경우, 공격자가 입력 창 및 URL에 SQL문을 삽임하여 DB로부터 정보를 열람, 조작할 수 있는 취약점 공격 기법
- 시큐어 코딩 : PreparedStatement객체를 상수 문자열로 생성하고 파라미터 부분을 setString등 메서드로 설정해 외부의 입력이 쿼리문의 구조를 바꾸는 것을 방지함

*XSS 크로스 사이트 스크립트 공격
- 웹 페이지에 악의적인 스크립트를 포함해 사용자 측에서 실행되게 유도할 수 있는 공격 기법
- 시큐어 코딩 : 외부 입,출력 값에 스크립트가 삽입되지 못하도록 & <> "' () 등에 대해 문자열 치환 함수를 구현

< 11과목>

* Windows 운영체제 특징
- GUI 제공
- 선점형 멀티태스킹 방식 제공
- 자동감지 기능 제공
- OLE 사용 (작성 중 문서에 개체 자유롭게 연결 또는 삽입)

* 유닉스 운영체제 특징
- 대화식 운영체제 기능
- 다중 작업 기능
- 다중 사용자 기능
- 이식성 제공
- 계층적 트리 구조 파일 시스템 제공

* 운영체제 제어 방법
- CLI : 사용자가 직접 명령어 입력하여 명령을 내림
- GUI : 마우스로 화면 클릭, 그래픽 위주 제어

* Windows 운영체제 기본 명령어

CALL	다른 일괄 프로그램 호출
CD	현재 디렉터리 이름 보여주거나 바꿈
CLS	화면 지움
CMD	명령 프롬프트 창 열어줌
COMP	두개 이상의 파일 비교
ECHO	메시지 표시 or ECHO 사용/사용하지 않음
EXIT	명령행 인터프리터 마침

* 리눅스/유닉스 기본 명령어

시스템 관련	uname -a : 시스템의 모든 정보 확인 uname -r : 운영체제 배포버전 출력 cat : 파일 내용 출력 uptime : 시스템 가동 시간, 현재 사용자 수, 평균 부하량
사용자	id : 사용자의 로그인 명, id 그룹 등 출력 last : 시스템의 부팅부터 현재까지 모든 사용자의 로그인/아웃 정보 표시 who : 현재 접속자 정보
- 파일 처리	ls : 자식히 속해있는 폴더 내 파일 및 폴더 표시 pwd : 현재 작업중인 디렉토리3) rm : 파일 삭제 cp : 파일 복사 mv : 파일 이동
프로세스	ps : 현재 실행 중인 프로세스 목록 pmap : 프로세스 ID 기준 메모리 맵 정보 출력 kill : 특정 PID 프로세스 종료
파일 권한	chmd : 특정 파일 , 디렉토리 퍼미션 수정 chown : 파일, 디렉토리 소유자 또는 그룹 수정
네트워크	ipconfig : 네트워크 인터페이스 설정, 확인 host : 도메인은 아는데 ip 주소를 모르거나 반대의 경우
압축	tar : 여러 개 파일을 하나의 파일로 묶거나 풀기 gzip : 압축 담당
검색	grep : 특정 문자열 찾기 find : 특정 파일 찾기
파일 이동	rysnc : 로컬 또는 원격에 파일 또는 디렉토리 복사, 동기화
디스크 사용	df : 마운트된 하드디스크 남은 용량 확인 du : 파일 사이즈를 킬로 바이트 단위로 보여줌
디렉토리 이동	cd

*** Windows 운영체제 특징**

- GUI 제공
- 선점형 멀티태스킹 방식 제공
- 자동감지 기능 제공
- OLE 사용 (작성 중 문서에 개체 자유롭게 연결 또는 삽입)

*** 유닉스 운영체제 특징**

- 대화식 운영체제 기능
- 다중 작업 기능
- 다중 사용자 기능
- 이식성 제공

- 계층적 트리 구조 파일 시스템 제공

*** 운영체제 제어 방법**

- CLI : 사용자가 직접 명령어 입력하여 명령을 내림
- GUI : 마우스로 화면 클릭 , 그래픽 위주 제어

*** 리눅스/유닉스 기본 명령어**

시스템 관련	uname -a : 시스템의 모든 정보 확인 uname -r : 운영체제 배포 버전 출력 cat : 파일 내용 출력 uptime : 시스템 가동 시간 , 현재 사용자 수 , 평균 부하량
사용자	Id : 사용자의 로그인 명 , id 그룹 등 출력 last : 시스템의 부팅부터 현재까지 모든 사용자의 로그인/아웃 정보 표시 who : 현재 접속자 정보
파일처리	ls : 자신이 속해있는 폴더 내 파일 및 폴더 표시 pwd : 현재 작업 중인 디렉터리 3) rm : 파일 삭제 cp : 파일 복사 mv : 파일 이동
프로세스	ps : 현재 실행 중인 프로세스 목록 pmap : 프로세스 ID 기준 메모리 맵 정보 출력 kill : 특정 PID 프로세스 종료
파일권한	chmd : 특정 파일 , 디렉터리 퍼미션 수정 chown : 파일 , 디렉토리 소유자 또는 그룹 수정
네트워크	ipconfig : 네트워크 인터페이스 설정 , 확인 host : 도메인은 아는데 ip 주소를 모르거나 반대의 경우
압축	tar : 여러 개 파일을 하나의 파일로 묶거나 풀기 gzip : 압축 담당
검색	grep : 특정 문자열 찾기 find : 특정 파일 찾기
파일 이동	rysnc : 로컬 또는 원격에 파일 또는 디렉터리 복사 , 동기화
디스크 사용	df : 마운트된 하드디스크 남은 용량 확인 du : 파일 사이즈를 킬로 바이트 단위로 보여줌
디렉터리 이동	cd

*** Windows 운영체제 기본 명령어**

CALL	다른 일괄 프로그램 호출
CD	현재 디렉터리 이름 보여주거나 바꿈
CLS	화면 지움
CMD	명령 프롬프트 창 열어줌
COMP	두 개 이상의 파일 비교
ECHO	메시지 표시 or ECHO 사용 /사용하지 않음
EXIT	명령행 인터프리터 마침

*** 운영체제 핵심 기능**

- 메모리 관리
- 프로세스 관리

*** 메모리 관리 기법**

- 반입 기법 : 다음 프로세스 반입 시기 결정 (요구반입, 호출반입)
- 배치 기법 : 어디에 적재할지 결정 (최초 적합 , 최적 적합 , 최악 적합)
- 할당 기법 : 메모리 적재 방법 (연속 할당 , 분산 할당)
- 교체 기법 : 메모리 교체 대상 결정 (FIFO, LRU, LFU 등 ..)

*** 프로세스 상태**

- 생성 상태 : 프로세스 생성 상태
- 준비 상태 : CPU 할당 받을 수 있는 상태
- 실행 상태 : CPu 할당 받아 동작 중
- 대기 상태 : 입출력 처리 등 발생 -> 대기 리스트에서 기다림
- 완료 상태 : 수행 종료 상태

*** 프로세스 상태 전이**

- 디스패치 : 실행될 프로세스 선정하여 CPU 할당 -> 문맥 교환 발생
- 타이머 런 아웃 : 실행 상태에서 준비 상태로 전이
- 블록 : 입출력 등 발생 -> 대기 상태로 전이
- 웨이크업 : 대기 상태에서 준비 상태로 전이
- Swap-in :다시 기억장치가 할당
- Swap-out : 기억장치를 잃은 경우

*** 프로세스 스케줄링**

- CPU를 사용하려고 하는 프로세스들 사이의 우선순위를 관리하는 작업
- 스케줄링 => 처리율과 CPU 이용률을 증가시키고 오버헤드, 응답시간, 반환시간, 대기시간을 최소화 시키기 위한 기법

*** 프로세스 스케줄링 주요 용어**

- 서비스 시간 : 결과 산출까지 소요되는 시간
- 응답 시간 : 입력되고 수행 결과를 산출하기까지 소요되는 시간 (대기시간+수행시간) => Trunaround Time
- 대기시간 : 프로세스가 프로세서에 할당 대기까지 큐에 대기하는 시간
- 종료 시간
- 시간 할당량 : 프로세서 독점을 막기 위해 서비스되는 시간 할당량
- 응답률 : (대기시간 + 서비스 시간) / 서비스 시간 => 요게 HRN 공식

*** 선점형 스케줄링 알고리즘**

: 하나의 프로세스가 cpu를 차지하고 있을때, 우선순위가 높은 다른 프로세스가 현재 프로세스를 중단시키고 cpu를 점유하는 스케줄링 방식

알고리즘 유형	동작 방식
라운드 로빈	- 프로세스는 같은 크기의 CPU시간을 할당, 할당 시간 내에 처리 완료를 못하면 준비 큐 리스트의 가장 뒤로 보내지고, CPU는 대기중인 다음 프로세스로 넘어감
SRT (Shortest Remaining Time First)	-가장 짧은 시간이 소요되는 프로세스를 먼저 수행하고, 남은 처리 시간이 더 짧다고 판단되는 프로세스가 준비 큐에 생기면 언제라도 프로세스가 선점됨
다단계 큐(Mulit Level Queue)	- 작업들을 여러 종류 그룹으로 분할, 여러 개의 큐를 이용하여 상위단계 작업에 의한 하위단계 작업이 선점 당함 - 각 큐는 자신만의 독자적인 스케줄링을 가짐
다단계 피드백 큐(Multi Level Feedback Queue)	- 입출력 위주와 CPU위주인 프로세스의 특성에 따라 큐마다 서로 다른 CPU시간 할당량을 부여 - FCFS와 라운드 로빈 스케줄링 기법을 혼합한 것으로, 새로운 프로세스는 높은 우선순위, 프로세ㅡ의 실행 시간이 길어질수록 점점 낮은 우선순위 큐로 이동하고 마지막 단계에는 라운드로 로빈 방식을 적용

*** 비선점형 스케줄링**

: 한 프로세스가 CPU를 할당받으면 작업 종료 후 CPU 반환 시까지 다른 프로세스는 CPU 점유가 불가능한 프로세스 방식

알고리즘 유형	동작 방식
우선순위 (Priority)	- 프로세스 별로 우선순위가 주어지고, 우선순위에 따라CPU를 할당함 - 동일 순위는 FCFS
기한부 (Deadline)	-작업들이 명시된 시간이나 기한 내에 완료되도록 계획
FCFS (First Come FIrst Servie)	- 프로세스가 대기 큐에 도착한 순서에 따라 CPU를 할당 - FIFO 알고리즘이라고도 함
SJF (Shortest Job First)	- 프로세스가 도착하는 시점에 따라 그 당시 가장 작은 서비스시간을 갖는 프로세스가 종료 시까지 자원 점유 - 준비 큐 작업 중 가장 짧은 작업부터 수행, 평균 대기시간 감소 - CPU 요구 시간이 긴 작업과 짧은 작업 간의 불평등이 심하여, CPU 요구 시간이 긴 프로세스는 기아 현상 발생
HRN (Highest Response Ratio Next)	- 대기 중인 프로세스 중 현재 응답률이 가장 높은 것을 선택 - SJF의 약점인 기아 현상을 보완한 기법으로 긴 작업과 짧은 작업 간의 불평등 완화 - HRN의 우선순위 =(대기시간+서비스시간) / 서비스시간

*** 가상화** : 물리적 리소스를 하나로 보이게 하거나 , 하나를 여러개로 보이게 하는 기술

- 종류 : 플랫폼 가상화 / 리소스 가상화

*** 가상화 기술요소**

- 컴퓨팅 가상화 : ex) 하이퍼바이저
- 스토리지 가상화 : ex) 분산 파일 시스템
- I/O 가상화
- 컨테이너 ex) 도커
- 분산처리 기술
- 네트워크 가상화 기술 ex) SDN, NFV

*** 클라우드 컴퓨팅 분류**

- 사설 클라우드 : 기업 내부 구축 , 보안성 높음 , 비용 문제
- 공용 클라우드 : 서비스 제공 업체에서 다중 사용자를 위함 , 확장성 , 유연성 높고 보안성 문제
- 하이브리드 클라우드 : 사설 클라우드 , 공용 클라우드 모두 사용

*** 클라우드 서비스 유형**

- IaaS (인프라형)
- PaaS (플랫폼형)
- SaaS (소프트웨어형)

*** 관계형 데이터베이스 종류**

- 오라클
- SQL Server : 윈도우 서버에서만 구동
- MySQL : 오픈 소스
- Maria DB

*** DBMS 유형**

- 키 -값 DBMS
- 컬럼 기반 데이터 저장 DBMS
- 문서 저장 DBMS
- 그래프 DBMS

*** DBMS 특징**

- 데이터 무결성 , 일관성 , 회복성 , 보안성 , 효율성

*** E-R 다이어그램**

- 현실 세계의 개체와 개체간의 관계를 도식화한 다이어그램
- 구성요소 : 개체 , 관계 , 속성

*** 트랜잭션 특성**

- 원자성 : 연산 전체가 성공 또는 실패 (커밋 , 롤백과 관련)
- 일관성 : 실행 성공 후 항상 일관된 DB 상태 유지

- 격리성 : 트랜잭션 실행 중 연산의 중간 결과를 다른 트랜잭션이 접근 불가
- 영속성 : 성공 완료된 트랜잭션의 결과는 영속적으로 DB 에 저장

*** 트랜잭션 상태**
- 활동 상태 (초기상태)
- 부분 완료 상태 (마지막 명령문 실행된 후 상태)
- 완료 상태 (트랜잭션 성공 완료 후)
- 실패 상태
- 철회 상태 (트랜잭션 취소로 시작전 상태로 환원된 상태)

*** 트랜잭션 제어 (TCL 명령어)**
- 커밋 COMMIT
- 롤백 ROLLBACK
- 체크포인트 CHECK POINT (롤백을 위한 시점 설정)

*** 빅데이터의 특성 (3V)**
- Volume (데이터의 양)
- Variety (데이터의 다양성)
- Velocity (데ㅣ터의 속도)

*** NoSQL 개념**
- 데이터 저장에 고정된 테이블 스키마가 필요 없고 , 조인 연산 사용할 수 없고 , 수평적으로 확장이 가능한 DBMS

*** NoSQL 특성**
- Bascially Availabe : 분산 시스템이기 때문에 가용성 중시
- Soft-State
- Eventually Consistency : 일정 시간이 지나면 일관성 유지

*** NoSQL 유형**
- Key-Value Store
- Column Family Data Store
- Graph Store

*** 데이터 마이닝**

- 대규모 데이터에서 의미있는 패턴을 파악하거나 예측하여 의사결정에 활용하는 기법

*** 데이터 마이닝 절차**

- 목적 설정 -> 데이터준비 -> 가공 -> 마이닝 기법 적용 -> 정보 검증

*** 데이터 마이닝 주요 기법**

- 분류 규칙 : 과거 데이터로 부터 분류 모형을 만들어 새로운 레코드값 예측
- 연관 규칙 : 데이터 안에 존재하는 항목 간 종속 관계
- 연속 규칙 : 시간에 관련한 정보가 포함된 형태의 기법
- 데이터 군집화 : 유사한 특성을 지닌 소그룹 분할

*** 네트워크 장비**

1계층	2계층	3계층
1) 허브 : 여러대 컴 연결 또는 하나에서 여러대 컴으로 송신 2) 리피터 : 신호 증폭	1) 브리지 : 두개의 LAN 연결 2) L2 스위치 3) NIC : 가장 빠른 속도로 데이터를 주고 받게 컴퓨터 내에 설치 4) 스위칭 허브	1) 라우터 2) 게이트웨이 3) L3 스위치 4) 인터넷 공유기 5) 망 스위칭 허브

*** 프로토콜**

개념	- 서로 다른 시스템이나 기기 간 데이터 교환을 원활히 하기 위한 표준화된 통신 규약 - 기본 요소 : 구문 , 의미 , 타이밍
특징	- 단편화 - 재조립 - 캡슐화 - 연결제어 : 데이터 전송량 , 속도 제어 - 오류 제어 - 동기화 : 송수신 시점을 맞추는 기법 - 다중화 : 하나의 통신 회선에 여러 기기 접속 - 주소지정

*** TCP 특징**
- 신뢰성 보장
- 연결 지향적 특징
- 흐름 제어
- 혼잡 제어

*** UDP 특징**
- 비 신뢰성
- 순서화되지 않은 데이터 그램 서비스 제공
- 실시간 응용 및 멀티캐스팅 가능
- 단순 헤더

*** IPv4 주소고갈 문제** -> 해결책 IPv6

*** 패킷 스위칭** ex) X.25 , 프레임 릴레이 , ATM (비동기 전송 모드, AAL-ATM-물리 계층 구조)
*** 서킷 스위칭** : 네트워크 리소스를 특정 사용층이 독점

*** 라우팅 알고리즘**
- 거리벡터 알고리즘
- 링크 상태 알고리즘

*** 라우팅 프로토콜 종류**
- RIP (최초 라우팅 프로토콜)
- IGRP (RIP 개선)
- OSPF (링크 상태 알고리즘 사용)
- BGP (대형 사업자 간 라우팅)

<12과목>

*** 제품 소프트웨어 패키징 적용 시 특성**

- 전체 내용 포함
- 버전 관리/릴리즈 노트
- 고객 중심
- 모듈화

*** 모듈화 장점**

- 개발 편의성 -> 유지보수 용이 등
- 복잡성 감소 -> 성능향상

*** 사용자 중심 패키징 고려 사항**

- 시스템 환경 : OS, CPU, 메모리 등
- 직관적 UI
- 관리 서비스
- 안정적 배포

*** 사용자 중심의 모듈 패키징 작업 수행 (순서) // 기모빌 사적변**

- 기능 식별
- 모듈화
- 빌드 진행
- 사용자 환경 분석
- 패키징 적용 시험
- 패키징 변경 개선

*** 릴리즈 노트 개념**

- 고객과 잘 정리된 릴리즈 정보를 공유하는 문서

*** 릴리즈 노트의 중요성**

- 정보 제공
- 관리의 용이성

*** 릴리즈 노트 작성 항목**

- 헤더
- 개요

- 목적
- 이슈 요약 : 버그의 간단 설명 또는 릴리즈 추가항목 요약
- 재현 항목
- 수정/개선 내용
- 사용자 영향도
- 소프트웨어 지원 영향도
- 노트
- 면책 조항
- 연락정보

*** 릴리즈 노트 예외 케이스**
- 테스트 단계에서의 베타 버전 출시
- 긴급 버그 수정 시
- 자체 기능 향상을 포함한 모든 추가 기능의 향상
- 사용자 요청에 따른 특이 케이스 발생

*** 릴리즈 노트 작성 프로세스 // 모정개 영정추**
- 모듈 식별
- 릴리즈 정보 확인
- 릴리즈 노트 개요 작성
- 영향도 체크
- 정식 릴리즈 노트 작성
- 추가 개선 항목 식별

*** 패키징 도구 활용 시 고려 사항**
- 암호화/보안 골
- 이기종 연동 고려
- 사용자 편의성 고려
- 적합한 암호화 알고리즘 적용

*** 저작권**
- 저작물에 대한 배타적 독점적 권리로 타인의 침해를 받지 않을 고유 권한

*** 클리어링 하우스**
- 디지털 저작권 라이선싱을 중개하고 라이선스 발급을 수행하는 정산소

*** 저작권 관리 구성요소**
- 콘텐츠 제공자

- 콘텐츠 분배자 (암호화된 콘텐츠 제공 , 쇼핑몰 등)
- 패키저 (콘텐츠를 메타 데이터와 함께 배포 가능 단위로 묶는 기능)
- 보안 컨테이너 (전자적 보안 장치)
- DRM 컨트롤러 (이용 권한 통제)
- 클리어링 하우스

*** 패키징 도구 구성 // 자세한건 12-10, 11 쪽**
- 암호화
-키 관리
- 식별 기술 (DOI, URI)
- 저작권 표현 (XrML, MPEG-21)
- 암호화 파일 생성
- 정책 관리 (XML, 콘텐츠 관리 시스템)
- 크랙 방지 (난독화, Secure DB)
- 인증

*** 애플리케이션 배포 도구를 활용한 배포 프로세스 // 빌식수 설배치**
- 빌드 내용 식별
- 패키징 도구 식별
- DRM 흐름을 확인하여 패키징 수행
- 패키징 도구 설치
- 배포 작업
- 정상 배포 확인

*** 제품 소프트웨어 설치 매뉴얼 개념**
- 최초 설치 시 참조하는 매뉴얼

*** 제품 소프트웨어 설치 매뉴얼 기본 작성 항목**
- 목차 및 개요
- 문서 이력 정보
- 설치 매뉴얼 주석
- 설치 도구 구성

*** 제품 소프트웨어 설치 매뉴얼 구성요소 // 위에꺼랑 헷갈려잉**
- 제품 소프트웨어 개요
- 설치 관련 파일

- 설치 절차
- 설치 아이콘
- 삭제 방법
- 설치 버전 및 작성자
- 고객 지원 방법 및 FAQ
- 준수 정보 & 제한 보증

*** 설치 매뉴얼 작성 프로세스**
- 개요 및 기능 식별
- UI 분류
- 설치 파일 / 백업 파일 확인
- 삭제 절차 확인
- 이상 유형 확인
- 최종 매뉴얼 적용

*** 제품 소프트웨어 사용자 매뉴얼 개념**
- 설치와 사용에 필요한 제반 절차 및 환경 등 전체 내용을 포함하는 메뉴

*** 제품 소프트웨어 사용자 매뉴얼 작성 항목**
- 목차 및 개요
- 문서 이력 정보
- 사용자 매뉴얼 주석
- 기록 항목
- 기본 사항
- 고객 지원 방법 및 FAQ
- 준수 정보 & 제한 보증

*** 제품 소프트웨어 사용자 매뉴얼 작성 프로세스**
- 작성지침 정의
- 사용자 매뉴얼 구성요소 정의
- 구성요소별 내용 작성
- 사용자 매뉴얼 검토

*** 제품 소프트웨어 형상 관리 개념**
- 변경 사항을 체계적으로 추적하고 통제하는 관리 기법 (베이스라인 설정, 버전 체계 관리)

*** 제품 소프트웨어의 형상 관리 역할**

- 관리 유용
- 동시 개발
- 빠른 복구
- SW 적시 공급

*** 버전 관리 도구의 버전 관리 항목**

- 가져오기 : 디렉토리 파일을 처음으로 저장소에 복사
- 추가 : 신규로 파일을 저장소에 추가
- 체크아웃 : 저장소 파일 받기
- 체크 인 : 저장소에 새로운 버전으로 갱신
- 업데이트 : 커밋 이후 새로운 개발자와 자기 작업 공간 동기화
- 커밋 : 체크인 시 갱신 사항이 있는 경우 충돌 알림
- 저장소 : 변경 이력 정보 저장 저장소
- 차이 : 기존 개발자가 추가한 파일과 이후 변경된 파일의 차이 확인

*** 소프트웨어 버전 관리 도구 개념**

- 형상 관리 지침을 활용하여 소프트웨어의 신규 개발, 변경, 개선과 같은 수정 사항 관리

*** 소프트웨어 버전관리 도구 유형**

- 공유 폴더 방식 (RCS, SCCS)
- 클라이언트/서버 방식 (CVS, SVN)
- 분산 저장소 방식 (Git)

*** 소프트웨어 버전관리 도구별 특징**

- CVS : 다수인원 동시에 운영체제 접근 가능
- SVN : 하나의 서버에서 소스를 쉽고 유용하게 관리하도록 도움
- RCS : 파일 수정을 한 사람으로 제한
- Git : 빠른 속도, 대형 프로젝트
- Clear Case : 복수 서버, 복수 클라이언트 구조

*** 소프트웨어 버전관리 도구 사용 시 유의사항**

- 버전에 대한 쉬운 정보 접근성
- 불필요한 사용자에 대한 접근 제어
- 동일 프로젝트에 대한 동시 사용성
- 빠른 오류 복구