1. 요구사항 확인

소프트웨어 생명 주기 모델 종류
- 폭포수 모델 : 각 단계를 확실히 마무리 지은 후에 다음단계로 넘어간다.
- 프로토타이핑 모델 : 프로토타입을 구현해, 고객의 피드백을 반영하며 만들어 간다.
- 나선형 모델 : 위험을 최소화하기 위해 점진적으로 개발한다.
- 반복적 모델 : 구축 대상을 나누어 병렬적으로 개발 후 통합하거나, 반복적으로 개발한다.

▶ 소프트웨어 개발방법론 종류

- 정보공학 방법론 : 정보시스템 개발에 필요한 관리 절차와 작업 기반을 체계화
- 객체지향 방법론 : '객체'라는 기본 단위로 시스템 분석 및 설계
- 애자일 방법론 : 절차보다는 사람이 중심이 되어 변화에 유연하고 신속하게 적응하면서 효율적으로 시스템 개발 (XP, 스크럼)
- 제품 계열 방법론 : 특정 제품에 적용하고 싶은 공통된 기능을 정의해 개발, 임베디드

▶ 델파이 기법 : 전문가의 경험적 지식을 통한 문제 해결 및 미래예측을 위한 기법

▶ XP의 12가지 기본원리 (메타포어 예상, 리펙토링 기출)
메타포어(Metaphor) : 공통적인 이름 체계와 시스템 서술서를 통해 고객과 개발자간의 의사소통을 원활하게 한다.
리팩토링(Refactoring) : 프로그램의 기능은 바꾸지 않고 중복제거, 단순화 등을 위해 시스템 재구성을 한다. 프로그램을 쉽게 이해하고 수정하여 빠르게 개발할 수 있도록 하기 위함.

▶ 비용 산정 모형 종류
LOC 모형 : 원시 코드 라인 수의 낙관치, 중간치, 비관치를 측정해 예측치를 구해 비용 산정
COCOMO 모형 : Organic 5만라인 이하, Semi-Detached 30만 이하, Embedded 30만 이상

▶ 일정관리 모델 : 일정 기한 내에 적절하게 완료될 수 있도록 관리
PERT : 일의 순서를 계획적으로 정리하기 위한 수렴 기법, 비관치, 낙관치, 중간치 이용

▶ 소프트웨어 아키텍처 4+1 뷰

- 유스케이스 뷰 : 다른 뷰를 검증하는데 사용 (시스템의 요구사항이자 사용자 입장에서 본 시스템의 기능)
- 논리 뷰 : 시스템의 기능적 요구사항 설명
- 프로세스 뷰 : 시스템의 비기능적 요구사항 설명
- 구현 뷰 : 모듈의 구성, 컴포넌트 구조, 의존성
- 배포 뷰 : 어떻게 배치되는가.

▶ 미들웨어

분산 컴퓨팅 환경에서 응용 프로그램과 프로그램이 운영되는 환경 간에 원만한 통신이 이루어질 수 있도록 제어해주는 소프트웨어

▶ 정형 기술 검토(요구사항 확인 및 검증 단계의 주요 기법)

동료 검토 : 이해 관계자들이 설명을 들으면서 결함을 발견
워크 스루 : 회의 전에 검토자료를 배포, 짧게 회의를 진행하는 형태
인스펙션 : 제작자 외에 전문가 또는 팀이 검사하는 기법

디자인패턴 : 소프트웨어 설계에서 공통으로 발생하는 문제에 대해 자주 쓰이는 설계 방법을 정리한 패턴 (생성, 구조, 행위)

생성(예상문제)
팩토리 메소드 : 상위에서는 객체 생성 인터페이스 정의, 하위에서는 인스턴스 생성
Singleton : 전역변수를 사용하지 않고 객채를 하나만 생성하도록함

구조 (예상)
Composite : 객체들의 관계를 트리 구조로 구성, 부분-전체 계층 표현
Adapter : 기존에 생성된 클래스를 재사용할 수 있도록 중간에서 맞춰주는 역할
Proxy : 실체 객체에 대한 대리 객체, 실체 객체를 드러나지 않게 해 정보은닉

행위 (기출+예상)
Observer : 한 객체의 상태가 바뀌면 그 객체에 의존하는 다른 객체들에 연락
Interpreter : 언어의 다양한 해석, 구문의 해석을 맡는 클래스 가각 작성
Mediator : 중간에 통제, 중재자

2장 화면 설계

▶ UI 설계 원칙(직유학유) <UX : 사용자와 시스템사이에 유사소통 돕는 매개체>

- 직관성(Intuitiveness) : 누구나 쉽게 이해하고, 쉽게 사용할 수 있어야 한다.
- 유효성(Efficiency) : 정확하고 완벽하게 사용자의 목표가 달성 될 수 있도록 제작한다.
- 학습성(Learnability) : 모두가 쉽게 배우고 사용할 수 있어야 한다.
- 유연성(Flexibility) : 사용자의 인터랙션을 최대한 포용하고, 실수를 방지할 수 있도록 제작

UI 유형 : CLI, GUI, NUI, OUI (CGNO!)

▶ UML(Unified Modeling Language) <- 사물, 관계, 다이어그램

객체지향 소프트웨어 개발 과정에서 산출물을 명세화, 시각화, 문서화 할 때 사용되는 모델링 기술과 방법론을 통합해서 만든 표준화된 범용 모델링 언어

▶ UML 다이어그램

◇ 구조적(Structural) 다이어그램 / 정적(Static) 다이어그램

- 클래스 : 클래스의 속성 및 연산과 클래스 간 정적 관계를 표현한 다이어그램
- 객체 : 클래스에 속한 사물, 인스턴스
- 컴포넌트 : 컴포넌트와 그들 사이의 의존 관계
- 배치(Deployment) : 컴포넌트 사이의 종속성, 물리적 요소들의 위치
- 복합체 구조(Composite Structure) : 클래스나 컴포넌트가 복합 구조를 갖는 경우 그 내부 구조를 표현하는 다이어그램
- 패키지 : 유스케이스나 클래스 등의 모델 요소들을 그룹화한 패키지들의 관계, 서로 다른 패키지들 사이의 의존 관계 표현

◇ 행위적(Behavioral) 다이어그램/ 동적(Dynamic) 다이어그램

- 유스케이스 : 시스템이 제공하고 있는 기능 관련된 외부 요소를 사용자의 관점에서 표현
- 시퀀스(Sequence) : 객체 간 상호작용을 시간적 개념을 중심으로 메시지 흐름으로 표현
- 커뮤니케이션 : 동작에 참여하는 객체들이 주고받는 메시지, 객체 간의 연관
- 상태(State) : 상호작용에 따라 상태가 어떻게 변화하는지
- 활동(Activity) : 어떤 기능을 수행하는지, 객체의 처리 로직, 조건에 따른 처리의 흐름
- 타이밍 : 객체 상태 변화와 시간 제약을 명시적으로 표현

3장 데이터 입출력 구현

▶ 데이터 모델 (개념적 모델 -> 논리적 모델 -> 물리적 모델, 개논물)
현실 세계의 정보를 인간과 컴퓨터가 이해할 수 있도록 추상화하여 표현한 모델

▶ 관계 대수 : 원하는 정보와 그 정보를 어떻게 유도하는가를 기술하는 절차적 정형 언어
- 일반 집합 연산자 : 합집합, 교집합 ∩, 차집합 - , 카티션 프로덕트 x
- 순수 관계 연산자 : SELECT, PROJECT, JOIN, DIVISION

▶ 정규화(Normalization)
데이터의 중복성을 제거해 이상현상을 방지하고, 데이터의 일관성과 정확성을 유지하기 위해 무손실 분해하는 과정

▶ 정규화 단계 (도부이걸다조? -> 도메인, 부분함수,이행함수 종속,결정자, 다치함수, 조인)
1NF : 도메인이 원자값
2NF : 부분함수 종속 제거
3NF : 이행함수 종속 제거
BCNF : 결정자 함수이면서 후보키 아닌 것 제거
4NF : 다치 종속 제거
5NF : 조인 종속 제거

▶ 이상 현상(Anomaly) -> 삽삭갱
데이터의 중복성으로 인해 릴레이션을 조작할 때 발생하는 비합리적인 현상
- 삽입 이상, 삭제 이상, 갱신 이상

▶ 반 정규화(De-Normalization) -> 정규화의 반대개념
정규화 된 엔티티, 속성, 관계에 대해 성능 향상과 개발 운영의 단순화를 위해 중복, 통합, 분리 등을 수행하는 과정

▶ 맵 리듀스(Map Reduce) : 구글에서 대용량 데이터 처리를 분산 병렬 컴퓨팅 처리하기 위한 목적으로 제작해 2004년에 발표한 소프트 프레임 워크

▶ 온톨로지(Ontology)
실세계에 존재하는 모든 개념들과 개념들의 속성, 개념들 간의 관계 정보를 컴퓨터가 이해할 수 있도록 서술해 놓은 지식베이스

▶ 데이터 마이닝(Data Minning)
대규모로 저장된 데이터 안에서 체계적이고 자동적으로 통계적 규칙이나 패턴을 찾아내는 기술

4장 통합 구현 (2021년 2회차에 0문제, 1회차에 3문제 출제)

• 연계 솔루션(EAI) : 기업에서 운영되는 서로 다른 플랫폼 및 애플리케이션들 간의 정보 전달, 연계, 통합을 가능하게 해주는 솔루션, 어댑터 이용
- Point-to-Point : 기본적인 애플리케이션 통합 방식 1:1 연결
- Message Bus : 애플리케이션 사이에 미들웨어를 두어 처리하는 방식
- Hub & Spoke : 허브 시스템을 통해 데이터를 전송하는 중앙 집중형 방식
- Hybrid : Hub & Spoke 와 Message bus 방식을 결합해서 사용

• Socket : 네트워크를 경유하는 프로세스 간 통신의 접속점, 소켓을 통해 클라이언트와 서버 프로그램 사이에 데이터를 송수신 할 수 있음 (출제 예상)

• Web Service/ESB : WSDL과 SOAP프로토콜을 이용한 시스템 간 연계

▶ 웹 서비스 유형

• SOAP(Simple Object Access Protocol) : HTTP, HTTPS, SMTP 등을 사용하여 XML 기반의 메시지를 네트워크 상태에서 교환하는 프로토콜

• WSDL(Web Service Description Language) : 웹 서비스 명, 제공 위치, 메시지 포맷, 프로토콜 정보 등 웹 서비스에 대한 상세 정보가 기술된 XML 형식의 언어

• UDDI(Universal Description, Discovery and Integration) : WSDL을 등록하고 검색하기 위한 저장소로 공개적으로 접근, 검색이 가능한 레지스트리이자 표준

5장 인터페이스 구현 (20/3~21/2 출제 X)

JSON : 개방형 표준 포맷, AJAX에서 많이 사용, XML을 대체하는 주요 데이터 포맷
XML : HTML의 단점을 보완한 특수한 목적을 갖는 마크업 언어
AJAX : 자바 스크립트를 사용하여 웹서버와 클라이언트간 비동기 웹기술, 전체 페이지를 새로 로드 하지 않고 필요한 부분만 로드함
REST : 웹과 같은 분산 하이퍼미디어 환경에서 자원의 존재/상태 정보를 표준화된 HTTP 메서드로 주고받는 웹 아키텍처

▶ 인터페이스 구현 검증 도구 (출제예상)

- xUnit : 자바, C++, .Net 등 다양한 언어를 지원하는 단위테스트 프레임워크
- STAF : 서비스 호출, 컴포넌트 재사용 등 다양한 환경 지원하는 테스트 프레임 워크
- FitNess : 웹 기반 테스트 케이스 설계/실행/결과 확인 등을 지원
- NTAF : FitNess + STAF
- watir : 루비 기반 웹 애플리케이션 테스트 프레임워크

6장 프로그래밍 언어 활용 (3회차 필기 문제) 꾸준히 3~5문제 출제 기출 확인

Python

Strcat(str, p2); -> str과 p2를 연결해라 (string 문자열로 저장 돼 있으면 문자열 연결)
char str[50] = “nation”;
char *p2 = “alter”; -> 포인터 변수 100번지에 문자열 alter 저장
strcat(str, p2); -> “nationalter” -> nation과 alter 두 개 연결
print(“%s”, str); -> nationalter 출력

출력 결과는 print 함수를 본다.
for num in range(n+1)
s += num -> s = s + num 누적함수
range(12) -> 0~11까지 출력 range에 언급이 없으면 0부터 시작함

range(1,3) -> range(시작수, 끝나는수) range(1,5) -> 1,2,3,4 까지 출력
단, range(5) 언급이 없으면 0부터 ~ 5까지임

C언어 (++, -- 값을 1 더한다, 1 뺀다 , ++a 전치연산 , ++b 연산 후에 b에 1더함)

int a = 1, b = 0;

b = a++; // b = 1, a = 2
b = a--; // b = 1, a = 0
b = ++a; // b = 2, a = 2
b = --a; // b = 0, a = 0;

a = 3, b = 4, c = 2; // ! NOT 참이면 거짓, 거짓이면 참
r1 = b <= 4 || c == 2; -> || 은 or 둘 중 하나라도 1이면 1 출력
r2 = (a>0) && (b<5); -> &&은 and 둘 다 1이면 1 출력
r3 = !c; -> 숫자에 not은 !참, 참의 반대는 거짓이므로 r3 0이 저장됨 거짓이 저장!

포인터
&a -> 주소 연산자, *p -> 포인터 연산자

atoi : 문자열을 숫자열로 바꿔준다 int num = atoi(str); -> str에 저장된 문자 숫자 변경

7장 SQL 응용 (꾸준히 3문제정도 출제)

트랜잭션 : 하나의 논리적 기능을 정상적으로 수행하기 위한 작업의 기본 단위
- 원자성(Atomicity) : 분해가 불가능한 작업의 최소단위, 연산 전체가 성공 또는 실패
- 일관성(Consistency) : 트랜잭션이 실행 성공 후 항상 일관된 데이터베이스 상태를 보존
- 격리성(Isolation) : 트랜잭션 실행 중 연산의 중간 결과를 다른 트랜잭션이 접근 불가
- 영속성(Durability) : 성공 완료된 트랜잭션의 결과는 영속적으로 데이터베이스에 저장

COMMIT : 트랜잭션 메모리에 영구적으로 저장
ROLLBACK : 오류가 발생했을 때, 오류 이전 특정시점으로 돌려주는 제어어
CHECKPOINT : 롤백을 위한 시점 지정

병행제어 : 다수 사용자 환경에서 여러 트랜잭션을 수행할 때, 데이터베이스의 일관성 유지를 위해 상호작용을 제어하는 기법
로킹 : 접근한 데이터에 대한 연산을 모두 마칠때까지, 상호 배제 하는 것 LOCK을 요청
(로킹 단위가 크면, 관리 쉽지만 병행성 낮음, 로킹 단위가 작으면 관리 어렵 병행성 높음)
타임스탬프 : 타임스탬프를 부여해, 부여된 시간에 따라 트랜잭션 수행

▶ 회복(Recovery) 기법 - 영속성 주요 기법
트랜잭션 수행 도중 장애로 인해 손상된 데이터베이스를 손상 이전의 정상적인 상태로 복구시키는 작업

- 로그 기반 회복 기법
- 지연 갱신 회복 기법(Deferred) : 트랜잭션이 완료된 후에 데이터베이스에 기록
- 즉각 갱신 회복 기법(Immediate) : 트랜잭션 수행 중 갱신 결과를 바로 DB에 반영
- 체크 포인트 회복 기법 : 장애 발생 이전의 상태로 복원
- 그림자 페이징 회복 기법 : DB 트랜잭션 수행 시 복제본을 생성하여 데이터베이스 장애 시 이를 이용해 복구하는 기법

▶ DDL의 대상
- 도메인 : 하나의 속성이 가질 수 있는 원자 값들의 집합
- 스키마 : 데이터베이스의 구조, 제약조건 등의 정보를 담고 있는 기본적인 구조
  - 외부(전체 데이터의 한 물리적인 부분), 개념(조직 전체의 입장), 내부 스키마 (DB의 물리적 구조)
- 테이블 : 데이터 저장 공간
- 뷰 : 가상의 테이블
- 인덱스 : 검색을 빠르게 하기 위한 데이터 구조

▶트리거(Trigger)
데이터베이스 시스템에서 삽입, 삭제, 갱신 등의 이벤트가 발생할 때마다 관련 작업이 자동으로 수행되는 절차형 SQL

8장 서버 프로그램 구현 (매 시험 1~2문제 출제) 보통 결합도 응집도

▶ 형상 관리(Configuration Management)
소프트웨어 개발을 위한 전체 과정에서 발생하는 모든 항목의 변경 사항을 관리하기 위한 활동

▶ 형상 관리의 절차 (식통감기)
- 형상 식별 : 형상 관리 대상 정의 및 식별
- 형상 통제 : 형상 항목 버전 관리를 위해 변경 여부와 변경 활동 통제
- 형상 감사 : 소프트웨어 베이스라인의 무결성 평가, 베이스라인 변경 시 요구사항과 일치하는지 검토

* 베이스 라인 : 개발과정의 각 단계별 산출물에 대한 변화를 통제하는 시점의 기준

- 형상 기록 : 형상 및 변경관리에 대한 각종 수행결과 기록

▶ 응집도(Cohesion) -> 높을수록 좋음 기능적 응집도가 제일 좋음
모듈의 독립성을 나타내는 정도, 모듈 내부 구성요소 간 연관 정도

▶ 결합도(Coupling) -> 낮을수록 좋음 자료 결합도가 제일 좋음
모듈 내부가 아닌 외부의 모듈과의 연관도, 모듈 간의 상호의존성, 모듈 간의 관련성

▶ 응집도 유형 (응 우논시 절교 순기)
- Coincidental Cohesion(우연적 응집도) : 모듈 내부의 구성요소가 각 연관이 없을 경우
- Logical Cohesion(논리적 응집도) : 유사한 성격, 특정 형태로 분류되는 처리 요소들이 한 모듈에서 처리되는 경우
- Temporal Cohesion(시간적 응집도) : 특정 시간에 처리 되어야 하는 활동들
- Procedural Cohesion(절차적 응집도) : 모듈이 다수의 관련 기능을 갖고, 모듈 안의 구성요소들이 그 기능을 순차적으로 수행할 경우
- Communication Cohesion(통신적, 교환적 응집도) : 동일한 입력과 출력을 사용해 다른 기능을 수행하는 활동들이 모임
- Sequential Cohesion(순차적 응집도) : 모듈 내 한 활동으로부터 나온 출력값을 다른 활동이 사용할 경우
- Functional Cohesion(기능적 응집도) : 모듈 내부의 모든 기능이 단일한 목적을 위해 수행되는 경우

▶ 결합도 유형 (결 자스제 외공내 -> 약한 순서대로 정리)

- Content Coupling(내용 결합도) : 다른 모듈 내부에 있는 변수나 기능을 다른 모듈에서 사용하는 경우 (제일 강한 결합도, 다모내)
- Common Coupling(공통 결합도) : 파라미터가 아닌 모듈 밖에 선언되어 있는 전역 변수를 참조하고 갱신하는 식으로 상호작용하는 경우 (모밖선)

• External Coupling(외부 결합도) : 두 개의 모듈이 외부에서 도입된 데이터 포맷, 통신 프로토콜, 또는 디바이스 인터페이스를 공유할 경우 (외도)
• Control Coupling(제어 결합도) : 어떻게 처리를 해야 한다는 제어요소가 전달되는 경우
• Stamp Coupling(스탬프 결합도) : 모듈 간의 인터페이스로 배열이나 객체, 구조 등이 전달되는 경우 (배객구)
• Data Coupling(자료 결합도) : 모듈 간의 인터페이스로 전달되는 파라미터를 통해서만 모듈 간의 상호 작용이 일어나는 경우 (제일 약한 결합도)

▶ 팬인(Fan-In) / 팬 아웃(Fan-Out) (예전 기출 그림 보면 이해 바로 됨)
• 팬인 : 모듈 자신을 기준으로 들어오면 팬인
• 팬 아웃 : 모듈 자신을 기준으로 나가면 팬 아웃

▶ JUnit : 자바 프로그래밍 언어용 단위 테스트 도구

9장 소프트웨어 개발 보안 구축 (매년 1~2개)

- 기밀성(Confidentiality) : 인가되지 않은 개인 혹은 시스템 접근에 따른 정보 공개 및 노출을 차단하는 특성 인가되지 않은 사용자 접근 X
- 무결성(Integrity) : 인가되지 않은 사용자는 데이터를 수정 할 수 없음.
- 가용성(Availability) : 인가된 사용자는 언제든 사용 가능.

임의적 접근 통제 (DAC):정보의 소유자가 보안 수준을 결정, 이에 대한 정보의 접근통제 설정
강제적 접근 통제 (MAC) : 사용자가 갖는 접근 권한에 근거해 시스템 접근 제한
네트워크 접근 제어 (NAC) : 내부 네트워크 접속 통제기능 제공
가상 사설 통신망 (VPN) : 가상 사설 네트워크, 전용회선을 이용하는거처럼 해주는 보안 솔루션

▶ DoS(Denial of Service)
시스템을 악의적으로 공격해 해당 시스템의 자원을 부족하게 해 사용하지 못하게 하는 공격

▶ DoS 공격의 종류

- SYN 플러딩(Flooding) : TCP 프로토콜의 구조적 문제를 이용한 공격, SYN 패킷만 보냄
- UDP 플러딩(Flooding) : 대량의 UDP 패킷을 만들어 임의의 포트 번호로 전송
- 스머프(Smurf)/스머핑(Smurfing) : 출발지 주소를 공격 대상의 IP로 설정하여, 네트워크 전체에게 ICMP Echo 패킷을 직접 브로드 캐스팅해 마비시킴
- 죽음의 핑(PoD) : ICMP 패킷(핑)을 정상적인 크기보다 아주 크게 만들어 전송
- 랜드 어택 : 출발지 IP와 목적지 IP를 같은 패킷 주소로 만들어 보내, 수신자가 자기 자신에게 응답
- 티어 드롭 : IP 패킷의 재조합 과정에서 잘못된 Fragment Offset 정보로 인해 수신 시스템이 단편화된 패킷의 재조합 과정에서 문제를 발생하도록 만드는 Dos 공격, IP 헤더가 조작된 일련의 IP 패킷 조각들을 전송

▶ DDoS(Distrivuted DoS)
여러 대의 공격자를 분산 배치해 동시에 동작하게 함으로써 특정 사이트 공격

▶ DDoS 공격 도구

- Trinoo : 많은 소스로부터 통합된 UDP flood 서비스 거부 공격을 유발하는데 사용
- TFN(Tribe Flood Network) : Trinoo와 비슷한 분산 도구, 많은 소스에서 하나 혹은 여러 개의 목표 시스템에 대해 서비스 거부 공격

▶ 네트워크 공격

• 스니핑(Sniffing) : 공격대상에게 직접 공격 하지 않고 데이터만 몰래 들여다보는 수동적 공격

• 네트워크 스캐너(Scanner), 스니퍼(Sniffer) : 공격자가 HW, SW 구성의 취약점을 탐색하는 공격 도구
• IP 스푸핑(Spoofing) : 침입자가 인증된 컴퓨팅 시스템인 것처럼 속여 정보를 빼내기 위해 본인의 패킷 헤더를 인증된 호스트의 IP 어드레스로 위조해 타깃에 전송
• ARP 스푸핑(Spoofing) : 공격자가 특정 호스트의 MAC 주소를 자신의 MAC 주소로 위조한 ARP Reply를 만들어 희생자에게 지속적으로 전송
• ICMP Redirect 공격 : 스니핑 시스템을 네트워크에 존재하는 또 다른 라우터라고 알림으로써 패킷의 흐름을 바꾸는 공격 기법
• 트로이 목마(Trojan Horses) : 악성 루틴이 숨어 있는 프로그램, 겉보기에는 정상적인 프로그램처럼 보이지만 실행하면 악성 코드를 실행하는 프로그램

▶버퍼 오버플로우(Buffer Overflow) 공격

메모리에 할당된 버퍼 크기를 초과하는 양의 데이터를 입력해 공격

• 랜섬웨어(Ransomware) : 감염된 시스템의 파일들을 암호화해 인질처럼 잡고 몸값을 요구하는 악성 소프트웨어 (출제 예상)

▶ 대칭키 암호화 알고리즘

• DES : 1975년 미국 연방 표준국(NIST)에서 발표 54Bits의 키와 64Bits의 블록
• SEED : 1999년 한국인터넷진흥원(KISA) 개발, 128Bits
• AES : 2001년 미국 표준 기술 연구소(NIST)에서 발표, DES 상위호환, DES 대체
• ARIA : 2004년 국가정보원과 산학연구협회가 개발
• IDEA : DES를 대체하기 위해 스위스 연방기술 기관에서 개발
• LFSR : 선형 되먹임 시프트 레지스터, 선형 함수로 계산

▶ 비대칭키 암호화 알고리즘

• RSA : 1977년 3명의 MIT 수학 교수가 고안, 소인수 분해하는 수학적 알고리즘
• ECC : RSA 암호 방식에 대한 대안, 타원 곡선 암호
• 디피-헬만 : 최초 공개키 알고리즘, 이산대수

▶ 해시 암호화 알고리즘

- MD5 : MD4를 개선한 알고리즘 , 프로그램이나 파일의 무결성 검사에 사용
- SHA : NSA에서 미 정부 표준으로 지정, DSA에서 사용, 해시 값 생성
- SHA-256/384/512
- HAS-160 : 국내 표준 서명 알고리즘을 위해 개발된 해시 함수, MD5장점+SHA-1장점

▶ IPSec(Internet Protocol Security)

IP계층에서 무결성과 인증을 보장하는 인증 헤더와 기밀성을 보장하는 암호화를 이용한 IP 보안 프로토콜

- 인증, 암호화, 키 관리 프로토콜로 구성

▶ SSL/TLS

전송계층과 응용계층 사이에서 클라이언트와 서버 간의 웹 데이터 암호화, 상호 인증 및 전송 시 데이터 무결성을 보장하는 보안 프로토콜

방화벽 : 기업 내부, 외부 간 트래픽을 모니터링 하여 시스템의 접근을 허용,차단하는 시스템

- 침입 탐지 시스템(IDS; Intrusion Detection System) : 네트워크에 발생하는 이벤트를 모니터링하고, 침입을 실시간으로 탐지하는 시스템
- 침입 방지 시스템(IPS; Intrusion Prevention System) : 네트워크에 대한 공격이나 침입을 실시간적으로 차단하는 시스템

디지털 저작권 관리(DRM) : 디지털 저작물에 대한 보호와 관리 솔루션

BIA(Business Impact Analysis) : 장애나 재해로 인한 운영상의 주요 손실을 볼 것을 가정하여 비즈니스 영향 분석

RTO : 업무중단 시점부터 업무가 복구되어 다시 가동될 때까지의 시간

RPO: 업무중단 시점부터 데이터가 복구되어 다시 정상 가동될 때 데이터의 손실 허용 시점

DRP(Disaster Recovery Plan) : 재난으로 장기간에 걸쳐 시설의 운영이 불가능한 경우를 대비한 재난 복구 계획

DRS(Disaster Recovery System) : 재해 복구 센터

XSS : 검증되지 않은 외부 입력 데이터가 포함된 웹페이지를 사용자가 열람할 때, 부적절한 스크립트가 실행되는 공격

SQL Injection : 악의적인 SQL 구문 삽입, DB의 정보를 열람, 조작하는 공격법

사이트 간 요청 위조(CSRF) : 사용자가 자신의 의지와는 무관하게 공격자가 의도한 행위를 특정 웹사이트에 요청하게 하는 공격

솔트 (Salt) : 일방향 해시 함수에서 다이제스트를 생성할 경우 추가되는 임의의 문자열

-> 소금이 기본 양념이듯 원문에 가미하여 암호문을 다른값으로 만드는 것

10장 애플리케이션 테스트 관리 (매회 2문제 출제)

• 개발 초기에 테스팅 시작 > 요르돈의 법칙(Snowball Effect, 눈덩이 법칙) : 개발 초기에 테스팅 하지 않으면 비용이 커진다.
• 결함 집중 > 파레토 법칙 : 소프트웨어 테스트에서 오류의 80%는 전체 모듈의 20% 내에서 발견된다.
• 살충제 패러독스 : 동일한 테스트 케이스로 반복해서 테스트하면 새로운 버그를 찾지 못한다.
• 오류-부재의 궤변 : 요구사항을 충족시키지 못한다면, 결함이 없다고 해도 품질이 높다고 볼 수 없다.

정적(정형기술검토) 테스트 :프로그램 실행 없이 구조를 분석 (동료 검토, 인스펙션, 워크스루)
동적 테스트 : 프로그램 실행을 요구하는 테스트 (화이트 박스, 블랙박스)
워크스루 : 회의 전, 동료검토 : 이해관계자 설명, 인스펙션 : 다른 전문가 결함 발견
검증 : 소프트웨어 과정을 테스트 (개발자 관점)
확인 : 소프트웨어 결과를 테스트 (사용자 관점)

▶ 화이트박스 테스트 유형 (중요한 커버리지들)
각 응용 프로그램의 내부 구조와 동작을 검사하는 소프트웨어 테스트 (구조 기반 테스트)
• 구문(Statement) 커버리지 : 프로그램 내의 모든 명령문을 적어도 한 번 수행
• 결정(선택, 분기)(Decision) 커버리지 : 프로그램 내 전체 결정문이 적어도 한번은 참과 거짓의 결과를 수행
• 조건(Condition) 커버리지 : 결정 명령문 내 각 조건이 적어도 한번은 참과 거짓이 출력
• 조건/결정 커버리지 : 전체 조건식 + 개별 조건식 모두 참 거짓 한번씩 출력
• 변경 조건/결정 커버리지 : 개별 조건식이 다른 개별 조건식에 영향을 받지 않고 전체 조건식에 독립적으로 영향을 주도록 함
• 다중 조건(Multiple Condition) 커버리지 : 모든 개별식 조건의 모든 조합을 고려

블랙박스 테스트(명세 기반 테스트) : 외부 사용자의 요구사항 명세를 보면서 수행하는 테스트
• 동등 분할(Equivalence Partitioning) 테스트 : 입력 자료에 초점을 맞춰 테스트 케이스를 만들고 검사하는 방법 (동등 분할 검사, 동등 분할 기법 등)

• 경곗값 분석(Boundary Value Analysis) 테스트 : 최솟값 바로 위, 최대치 바로 아래 등 입력값의 극한 한계를 테스트 하는 기법 -> 80, 100, 60, 79, 0, 59 경계값 분석

평가점수	성적
80~100	A
60~79	B
0~59	C

명세 기반 테스트 : 동등 분할, 경계값 분석, 결정 테이블, 유스케이스, 상태 전이
경험 기반 테스트 : 블랙박스 테스트

단위 테스트 : 모듈이나 컴포넌트에 초점을 맞춰 테스트 (구조 기반, 명세 기반)
통합 테스트 : 모듈간 상호 작용 테스트
시스템 테스트 : 시스템 기능 테스트
인수 테스트 : 사용자 입장에서 테스트, 알파테스트(함께), 베타 테스트(따로) (인알베)

하향식 통합 : 가장 상부부터 하부로 내려가면서 통합 (Stub 사용, 다른 프로그래밍 기능을 대리하는 코드, 기존 코드를 흉내내거나 아직 개발되지 않은 코드를 임시 대치)
상향식 통합 : 가장 하부부터 올라가면서 통합 (테스트 드라이버 사용)
빅뱅 통합 : 실제 모듈 테스트 진행

Mock 객체 : 스텁의 객체지향 버전, 독립적인 컴포넌트 테스트에 사용

테스트 오라클 : 테스트의 결과가 참인지 거짓인지 판단하기 위해서 사전에 정의한 참 값을 입력하여 비교

참 오라클 : 모든 입력값에 대해 기대하는 결과 생성, 모든 오류 검사
샘플링 오라클 : 특정 몇 개의 입력 값에 대해 결과값을 제공
휴라스틱 오라클 : 샘플 오라클 개선, 특정 몇 개의 입력값에 대한 결과를 제공
일관성 검사 오라클 : 애플리케이션 변경이 있을 때 전,후의 결과값 확인

테스트 하네스 : 애플리케이션 컴포넌트 및 모듈을 테스트하는 환경의 일부분, 테스트를 지원하기 위한 코드와 데이터

▶ 애플리케이션 성능 측정 지표 (처응경자, 처리량,응답,경과,자원)
• 처리량(Throughput) : 일정 시간 내 애플리케이션이 처리하는 작업의 양
• 응답 시간(Response Time) : 작업을 요청해서 응답시간까지 걸린 시간
• 경과 시간(Turnaround Time) : 작업을 의뢰한 시간부터 처리가 완료될 때 까지 걸린 시간
• 자원 사용률(Resource Usage) : 애플리케이션이 트랜잭션을 처리하는 동안 사용하는 CPU 사용량, 메모리 사용량, 네트워크 사용량

외계인 코드 : 아주 오래되거나, 개발자가 없어 유지보수가 어려운 코드

11장 응용 SW 기초 기술 활용 (매 회 3~4문제씩 꾸준히 출제)

운영체제 종류 : UNIX, Widows, Linux, MAC, Android

▶ 리눅스/유닉스 계열의 기본 명령어

• chmod : 특정 파일 또는 디렉토리의 퍼미션 수정 명령어

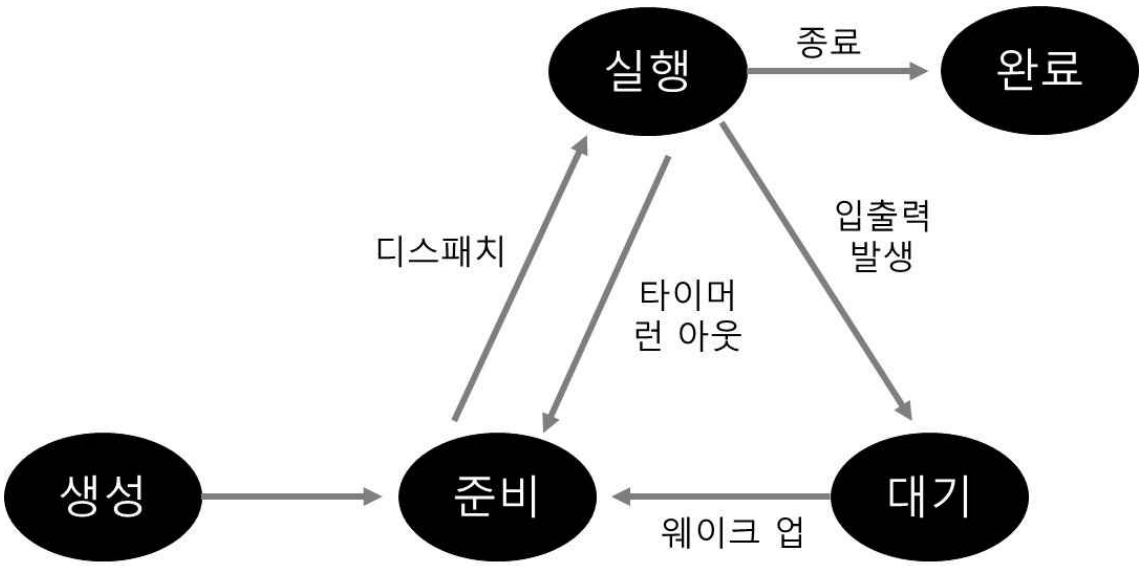
ex) chmod 751 a.txt (사용자, 그룹, 그 외 2진수로 000~111 까지 출력 권한 부여)

사용자에게 읽기/쓰기/실행 권한을 부여하고 그룹에게는 읽기/실행을 부여 그 외에는 실행권한을 a.txt 에 부여하는 명령어

• chown : 파일이나 디렉토리의 소유자, 소유 그룹 명령어

메모리 관리 기법 : 반 배 교 할 (반입, 배치, 교체, 할당 : 적재 방법 결정)

메모리 배치 기법 : 최초 적합, 최적 적합, 최악 적합



준비 -> 실행 : 디스패치 , 실행 -> 준비 : 타이머 런 아웃, 대기 -> 준비 : Wake up

▶ 프로세스 스케줄링 유형

◇ 선점형 스케줄링 : 우선순위가 높은 다른 프로세스가 현재 프로세스 중단 후 CPU 점유

• 라운드 로빈(Round Robin) : 같은 크기의 CPU 시간 할당

• SRT(Shortest Remaining Time First) : 가장 짧은 시간이 소요되는 프로세스를 우선 교체

• LRU(Least Recently Used), NUR(Not Used Recently) : 최근에 사용하지 않은 페이지 교체

◇ 비선점형 스케줄링 : 우선순위, 응답률이 높은 프로세스 먼저 수행

- FCFS(First Come Frist Service) : 프로세스가 대기 큐에 도착한 순서에 따라 CPU 할당
- SJF(Shortest Job First) : 프로세스가 도착하는 시점에 따라 그 당시 가장 작은 서비스 시간을 갖는 프로세스가 종료 시까지 자원 점유, 기아 현상 발생
- HRN(Highest Response Ratio Next) : 대기 중인 프로세스 중 현재 응답률이 가장 높은 것을 선택, 기아 현상 최소화 기법 (응답률 순 서비스시간+대기시간/서비스시간)

▶ 클라우드 컴퓨팅(Cloud Computing)

인터넷의 서버를 통해 IT 관련 서비스를 한 번에 사용할 수 있는 컴퓨팅 환경

▶ 클라우드 컴퓨팅 유형
- 인프라형 서비스(laaS) : 서버, 스토리지 같은 시스템 자원을 클라우드로 제공하는 서비스
- 플랫폼형 서비스(PaaS) : 애플리케이션을 개발, 실행, 관리할 수 있게 하는 플랫폼을 제공하는 서비스
- 소프트웨어형 서비스(SaaS) : 클라이언트를 통해 접속하여 소프트웨어를 서비스 형태로 이용하는 서비스

▶ 프로토콜(Procotcol) 구문,의미,타이밍
서로 다른 시스템에 있는 두 개체 간의 데이터 교환을 원활히 하기 위한 일련의 통신규약

▶ 프로토콜의 기본 3요소
- 구문(Syntax) : 시스템 간의 정보 전송을 위한 데이터 형식, 코딩, 신호레벨 등의 규정
- 의미(Semantic) : 시스템 간의 정보 전송을 위한 제어 정보로 조정과 에러 처리를 위한 규정
- 타이밍(Timing) : 시스템 간의 정보 전송을 위한 속도 조절과 순서 관리 규정

▶ OSI 7계층 : 국제표준화기구(ISO)에서 개발, 네트워크 프로토콜 디자인과 통신을 계층으로 나눠 설명한 것 (아파서 티내다, 피나다, APS TND P)
1. Physical Layer : 전송에 필요한 두 장치간의 실제 접속, 전기적 신호 변환 , 리피터(입력 신호 증폭), 허브(가까운 컴퓨터 연결)
2. DataLink Layer : 각종 흐름, 순서 제어 - 프레임 - 브리지, 스위치 - PPP,Ethernet
3. Network Layer : 라우팅, 혼잡 제어 - 패킷 - 라우터(경로 설정) -> IP,IGMP,BGP,ARP (IP주소를 MAC주소로 변환), RARP (MAC주소를 IP주소로 변환), ICMP(IP패킷을 처리할 때 발생되는 문제를 알려주는 메시지형 프로토콜)
4. Transport Layer : 종단 시스템 간 투명한 데이터 전송 - 세그먼트
(TCP : 신뢰성 보장, 연결 지향적 특징, 흐름,혼잡 제어) (UDP : 비신뢰성, 비연결성)
5. Session Layer : 동기점 (데이터의 회복) - RPC (원격제어)

6. Presentation Layer : 통신에 알맞은 형태 - 데이터 - JPEG,MPEG
7. Application Layer : 사용자와 네트워크간 응용 서비스 연결 - 데이터 - HTTP, FTP, SMTP, TELNET (인터넷이나 로컬 영역에서 사용)

▼ 라우팅 프로토콜의 구성

<내부라우팅 프로토콜(IGP)>

• RIP(Routing Information Protocol) : AS 내에서 사용하는 거리 벡터 알고리즘에 기초해 개발된 내부 라우팅 프로토콜, 벨만-포드 알고리즘, 15홉 제한, IGRP
• OSPF(Open Shortest Path First) : 규모가 크고 복잡한 TCP/IP네트워크에서 RIP의 단점 개선위한 링크 상태 알고리즘 적용해 최단 경로를 찾는 프로토콜, 다익스트라 알고리즘, 홉 제한 없음, ELGRP

<외부 라우팅 프로토콜(EGP)>
• BGP : AS 상호 간에 경로 정보를 교환하기 위한 라우팅 프로토콜

▼ 라우팅 알고리즘의 유형
• 거리 벡터(Distance Vector) 알고리즘 : 인접 라우터와 정보를 공유해 목적지까지의 거리와 방향을 결정하는 알고리즘, 벨만-포드 알고리즘 사용

• 링크 상태 알고리즘 : 링크 상태 정보를 모든 라우터에게 전달해 최단 경로 트리 구성, 다익스트라(Dijkstra) 알고리즘 사용

IPv4 : 32bit 주소 8bits 4개, Unicast, Multicast, BroadCast
IPv6 : 128bits 주소, 32bits 4개, Unicast, Multicast, Anycast

DNS(Domain Name System) : 도메인 네임을 ip주소로 매핑
SAN(Storage Area Network) : 여러 OS 기종이 동일 저장장치의 데이터 공유
NAT(Network Address Translation) : 사설 ip를 공인 ip 로 바꾸는 네트워크 주소 변환
MQTT : 제한된 대역폭의 푸시기술 기반 경량 메시지 전송 규약

12장 제품 소프트웨어 패키징 (안나오는 추세 PASS)

<신기술 동향 & 기타>

- Map reduce: 구글에서 대용량 데이터를 분산 병렬 컴퓨팅에서 처리하기 위한 목적으로 제작(Hadoop)
- RTO(Recovery Time Objective): 복구 목표 시간, 서비스를 복구하는 데까지 걸리는 최대 허용 시간
- RPO(Recovery Point Objective): 복구 목표 시점, 감내 가능한 손실 허용 시점, 어느 시점으로 백업할지
- BCP(Business Continuity Planning): 장애 및 재해로부터 조직의 생존을 보장하기 위한 예방/복구 계획
- DRP(Disaster Recovery Planning): 재해 복구 계획, 재해에 대비하여 빠른 복구를 위한 제반 계획
- DRS(Disaster Recovery System): 재해 복구 시스템, 재해 복구 계획의 원활한 수행을 지원하기 위해 평상
시에 확보하여 두는 인적/물적 자원 및 이들에 대한 지속적인 관리 체계가 통합된 것
- Mirror Site: 주 센터와 데이터복구센터 모두 운영 상태로 실시간 동시 서비스가 가능한 재해 복구 센터
- Hot Site(동일한 수준 Stanby), Warm Site(중요한 자원만, 수 시간) Cold Site(최소한, 수일 주기 백업)
- 디지털 포렌식(디지털 법의학): 범죄 사실에 대한 디지털 증거자료를 수집/복사/제출하는 일련의 과정
- 애자일 개발 방법론: 변화에 유연하고 절차보다는 사람 중심인 신속 적응적 경량 개발 방법론
- 럼바우 방법: 분석 활동을 객체 모델링/동적 모델링/기능 모델링 으로 나누어 수행하는 방법
- Tailoring: 프로젝트의 필요에 따라 SW 개발 프로세스/기법 등을 비즈니스/기술적 요구에 최적화
- NAS(Network Attached Storage): 서버와 저장 장치를 네트워크로 연결하는 방식
- MVC 패턴: 모델, 뷰, 컨트롤러로 구성되어 비즈니스 로직을 서로의 영향 없이 쉽게 고칠 수 있는 패턴
- 멀티 프로그래밍: 2개 이상의 프로그램을 CPU가 번갈아 사용해가며 자원 활용률을 극대화하는 기법
- ETL(Extraction Transformation Loading): 데이터를 추출하고 변환하고 전송 및 로딩하는 이동 프로세스
- XP: 비즈니스 요구 변동이 심할 때 적합(원리- 짝프로그래밍, 리팩토링, 테스트 기반 개발, 작은 릴리즈)
- XP의 특징: 의사소통, 단순성, 피드백, 용기, 존중
- Secure SDLC(Secure Sofrware Development Life Cycle): 보안 소프트웨어 개발 생명주기

- HA(고가용성): 서버와 네트워크, 프로그램 등 정보시스템이 오랜 기간 동안 계속 정상 운영 가능한 성질
- TELNET: 원격의 컴퓨터에 접속하여 자신의 컴퓨터처럼 사용할 수 있도록 해주는 프로토콜
- 하이퍼바이저: 하나의 컴퓨터에서 동시에 다수의 OS를 구동시킬 수 있는 SW 가상화 플랫폼
- 데이터 웨어하우스: 다량의 데이터를 효과적으로 분석하여 정보화하고 효율적으로 사용할 수 있게 된 DB
- 에이징 기법: 기다린 시간에 비례하여 프로세스의 우선 순위를 높여주는 기법
- 리팩토링: 기능은 수정하지 않고 복잡한 소스코드를 수정/보완하여 가독성/유지보수성 높이는 기법
- NAC(Network Access Control): 내부 PC의 MAC 주소를 IP관리 시스템에 등록 후 보안관리 기능 제공
- 쿠버네티스: 컨테이너화된 APP의 자동 배포, 스케일링 등 제공하는 오픈소스 관리시스템(리눅스 재단)
- CLASP(Comprehensive, Lightweight Application Security Process): 이미 운영중인 시스템에 적용하기 좋음
- 척와: 비정형 데이터 수집기술, 분산된 각 서버에서 에이전트를 실행하고, 컬렉터가 에이전트로부터 데이
터를 받아 HDFS(하둡 File system)에 저장하는 기술
- 스쿱: 정형 데이터 수집기술, 커넥터를 이용하여 RDBMS에서 HDFS로 데이터를 수집하는 기술
- MEMS: 초정밀 반도체 제조기술 바탕, 초미세 장치
- Stack Guard: 메모리 상에서 프로그램 복귀 주소와 변수 사이에 특정 값 저장->변경되면 실행 중단
- HIPO: 하향식 소프트웨어 개발을 위한 문서화 도구
- 하둡 :대량의 자료를 처리할 수 있는 대형 컴퓨터에서 동작하는 자바 오픈소스 프레임워크
- 형상관리 : 모든 항목의 변경사항을 관리하기 위한 활동