# 면접 예상 질문과 답변

중앙정보 취업지원센터
Tel : 02-712-0524
Fax : 02-393-9218
Mobile :
- 홍성표 컨설턴트 : 010-5072-2311
- 심원태 컨설턴트 : 010-9276-3769
e-mail : job@choongang.co.kr

중앙정보처리학원
중앙정보기술인재개발원
SINCE 1969 www.choongang.co.kr

# – 목 차 –

# [Programming]

**1. 프로그램이 무엇인지 말해 보세요.**
- 컴퓨터가 사람 일을 할 수 있도록 해 주는 것
- 컴퓨터에 처리되는 작업의 순서를 논리적으로 명령어로 작성하는 것

**2. JAVA언어의 좋은 점에 대하여 말해 보세요.**
객체지향형 프로그래밍 언어, 플랫폼 독립적, 이식성이 좋다, 라이브러리 지향성, 보안성, 멀티 스레드, 가상머신, 바이트 코드

**3. 객체지향 언어(Object Oriented Language)의 장점과 객체지향 프로그램이 등장한 이유에 대해 설명하세요.**
- 코드의 재사용성이 높아 새로운 코드를 작성할 때 기존의 코드를 이용하여 쉽게 작성할 수 있다.
- 코드의 관리가 용이하여 코드간의 관계를 이용해서 적은 노력으로 쉽게 코드를 변경할 수 있다.
- 제어자와 메서드를 이용해서 데이터를 보호하고 올바른 값을 유지하도록 하며, 코드의 중복을 제거하여 코드의 불일치로 인한 오 동작을 방지할 수 있다.

**4. 자바의 데이터 타입인 Primitive Type 과 Reference Type 에 대해 설명하세요.**
- Primitive Type 은 변수에 값 자체를 저장하며 Reference Type 은 메모리상에 객체가 있는 위치를 저장한다.
- Primitive Type의 종류는 boolean, byte, char, short, int, float, long, double 총 8가지 이며 Reference Type 종류는 클래스타입, 인터페이스타입, 배열타입, 열거타입이 있다.

**5. 배열과 컬렉션의 차이점에 대해서 설명하세요.**
- 배열은 단 하나의 자료형만 저장이 가능하고, 컬렉션은 복수의 자료형을 저장할 수 있다.
- 배열은 고정된 크기이고, 컬렉션은 가변적 크기이다.

**6. 다형성이란 무엇인가?**
- 여러 가지 데이터를 다룰 수 있는 특성을 뜻한다.
- 조상클래스의 인스턴스를 이용하여 자손타입의 클래스를 다룬다거나, 메서드 오버로딩을 통하여 동일 이름의 메서드를 이용하여 다양한 형태의 파라미터를 다루는 것을 뜻한다.

**7. 멀티 쓰레드의 장단점은 무엇인가?**
두 가지 이상의 작업을 동시해 실행 할 수 있어 자원을 효율적으로 이용할 수 있으나 dead lock 및 동기화에 대한 철저한 검증이 필요하다.

**8. 자바에서 멀티쓰레드를 구현하는 방법에 대해서 설명하세요.**
Thread 클래스를 상속하는 법. 단일 상속만 된다는 단점이 있다. Runnable 인터페이스를 상속하는 법. 다중 상속이 된다는 장점이 있다.

**9. Java 콜렉션의 대표 인터페이스는 무엇인가?**
List
순서가 있는 데이터의 집합으로 데이터의 중복을 허용한다.
구현클래스 - ArrayList, LinkedList, Stack, Vector
Set
순서를 유지하지 않는 데이터의 집합으로 데이터의 중복을 허용하지 않는다.
구현클래스 - HashSet, TreeSet
Map
키와 값의 쌍으로 이루어진 데이터의 집합으로 순서는 유지되지 않으며, 키는 중복을 허용하지 않고,
값은 중복을 허용한다.
구현클래스 - HashMap, TreeMap, HashTable, Properties

**10. 컬렉션에서 제네릭이 추가된 이유에 대해서 설명하세요.**
컬렉션은 복수개의 데이터 타입 요소값이 저장되다 보니 원하는 자료형 타입 요소값을 추출하기 어렵다. 그러므로 지정한 자료형 타입 한가지만 저장하기 위해서 나온 것이 제네릭이다.

**11. 접근제어자의 종류와 특성에 대하여 설명하라.**
private - 같은 클래스 내에서만 접근이 가능하다.
default - 같은 패키지 내에서만 접근이 가능하다.
protected - 같은 패키지 내에서, 그리고 다른 패키지의 자손클래스에서 접근이 가능하다.
public - 접근 제한이 없다.
접근 허용 범위는 다음과 같다 public > protected > default > private

**12. Wrapper 클래스란 무엇인가?**
primitive 타입으로 표현할 수 있는 간단한 데이터를 객체로 만들어야 할 경우가 있는데 그러한 기능을 지원하는 클래스를 뜻하며 각 primitive 타입에 대응하는 Wrapper 클래스는 아래와 같다.
byte => Byte
short => Short
int => Integer
long => Long
char => Character
float => Float
double => Double
boolean => Boolean

**13. 추상클래스란 무엇인가?**
클래스를 설계도에 비유한다면 추상클래스는 미완성 설계도에 비유할 수 있다. 미완성 설계도란, 단어의 뜻 그대로 완성되지 못한 채로 남겨진 설계도를 말한다. 클래스가 미완성이라는 것은 멤버의 개수에 관계된 것이 아니라, 단지 미완성 메서드(추상메서드)들 포함하고 있다는 의미이다. 미완성 설계도로 완성된 제품을 만들 수 없듯이 추상클래스로 인스턴스는 생성활 수 없다. 추상클래스는 상속을 통해서 자손클래스에 의해서만 완성될 수 있다.

**14. 인터페이스란 무엇인가?**
인터페이스는 일종의 추상클래스이다. 인터페이스는 추상클래스처럼 추상메서드를 갖지만 추상클래스보다 추상화 정도가 높아서 추상클래스와는 달리 몸통을 갖춘 일반 메서드 또는 멤버변수를 구성원으로 가질 수 없다. 오직 추상메서드와 상수만을 멤버로 가질 수 있으며, 그 외의 어떠한 요소도 허용하지 않는다. 추상클래스를 부분적으로만 완성된 '미완성 설계도'라고 한다면, 인터페이스는 구현된 것은 아무것도 없고 밑그림만 그려져 있는 '기본 설계도'라고 할 수 있다. 인터페이스는 인터페이스로부터만 상속받을 수 있으며, 클래스와는 달리 다중상속, 즉 여러 개의 인터페이스로부터 상속을 받는 것이 가능하다.

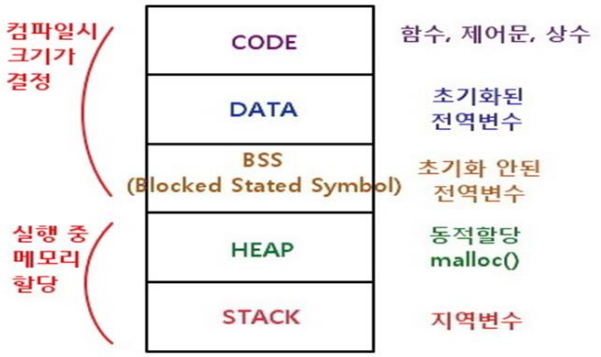
**14-1. 인터페이스를 왜 쓰는지 설명해 보세요.**
. 다형성을 구현하기 위해서 사용한다
즉 같은 인터페이스를 구현하는 클래스들은 실행하는 객체가 인터페이스를 통하여 다른 객체를 사용할 수 있도록 하여 클래스간의 결합도를 낮추고 응집도를 높힐 수 있음. 예로 스프링에서 DI(의존성 주입) 등에서 다형성을 구현하고 결합도를 낮추기 위해 사용
. 양식 역할 : 규칙에 맞게 구현하도록 표준을 제시하고 클래스들이 메세드를 빠짐없이 재정의 하도록 한다

**15. 자바의 GUI 에서 스윙과 AWT 의 차이점에 대해서 설명하세요!**
AWT 는 운영체제가 갖고 있는 각 컴포넌트를 이용한다. 즉 운영체제에 종속 적인 GUI 로서 운영체제 종류에 따라 화면에 출력되는 모양이 약간씩 다르다.
스윙은 운영체제가 갖고 있는 GUI 를 사용하지 않고 자바 가상머신의 Swing 패키지를 직접 사용해 구현한다. 따라서 운영체제가 서로 달라도 동일한 화면을 출력하게 한다. 스윙은 AWT 보다 세련되고 다양한 GUI 를 제공해 주기 때문에 자바 어플 GUI 개발에서는 AWT 보다는 스윙을 더 많이 사용한다.

**16. 가비지 콜렉터(Garbage Collection) 에 대하여 말해 보세요.**
쓰레기 수집(Garbage Collection)은 동적 할당된 메모리 영역 가운데 더 이상 사용할 수 없게 된 영역을 탐지하여 자동으로 해제하는 기법이다. 쓰레기 수집이 지원되는 환경에서는 프로그래머가 동적으로 할당한 메모리 영역의 전체를 완벽하게 관리할 필요가 없어진다. 이러한 작업은 CLR 의 '가비지 컬렉터'라는 일종의 백그라운드 서비스를 통해 자동으로 이뤄진다

**17. 컬렉션 List 인터페이스를 구현한 ArrayList 컬렉션 클래스에 대해서 말해 보세요.**
- 가변적 크기이다
- 순차적으로 요소 값을 저장한다.
- 중복 요소 값을 허용한다.

17-1. 자바(JAVA)의 메모리 구조가 어떤 식으로 이루어져 있는지?




17-2. out of memory 에러 메시지는 어느 메모리 부족으로 나타나는 것인지?
- heap 메모리 부족

18. 메서드에서 오버로딩과 오버라이딩(Overriding)에 대하여 말해 보세요.
@오버로딩 : 메소드 이름은 같지만 매개변수의 개수나 데이터 형식을 다르게 정의하는 것

@오버라이딩(Overriding) : 기존에 있는 메소드를 재정의하는 것으로 매개변수의 개수나 데이터형식이 같아야 한다

19. Exception 에 대하여 말해 보세요.
- 프로그램의 비정상적인 종료나 중단을 막을 수 있다.
- RuntimeException 과 그 외의 Exception 으로 나눌 수 있다.

20. OSI 7 계층(Layer)에 대하여 말해 보세요.
‣ OSI (Open System Interconnection) 모델의 7 개 계층구조

| 구 분 | | 설 명 |
|---|---|---|
| 네트워크 지원계층<br>(한 장치에서 다른 장치로 데이터를 이동할 때 필요한 물리적인 면(즉, 전기적인 규격, 물리적인 연결, 물리주소, 전송시간과 신뢰도 등) 처리) | 1. Physical layer<br>(물리층) | 물리적 매체를 통해 비트 흐름을 전송하기 위해 필요한 기능들을 조정하고, 인터페이스의 기계적·전기적 규격, 전송매체를 다룸. 물리적인 장치와 인터페이스가 전송을 위해 필요한 기능과 처리절차를 규정함 |
| | 2. Data link layer<br>(데이터링크층) | 가공되지 않은 내용의 전송을 담당하는 물리층을 신뢰성 있는 링크로 변환시켜 주고 노드-대-노드 전달(node-to-node delivery)함 |
| | 3. Network layer<br>(네트워크층) | 패킷을 발신지로부터 여러 네트워크(링크)를 통해 목적지까지 전달함 |
| 전송층<br>(종단-대-종단까지의 믿을 만한 데이터 전송 보장) | 4. Transport layer<br>(전송층) | 전체 메시지의 프로세스-대-프로세스 전달을 함 |
| 사용자 지원계층<br>(서로 상관없는 소프트웨어 시스템 사이의 상호연동을 가능하게 함) | 5. Session layer<br>(세션층) | 네트워크의 대화 조정자로 통신하는 시스템들 사이의 상호작용을 설정·유지하고 동기화 함 |
| | 6. Presentation layer<br>(표현층) | 두 시스템 사이에서 교환되는 정보의 구문과 의미에 관련되어 변환, 압축 및 암호화를 담당함 |
| | 7. Application layer<br>(응용층) | 사용자(사람 또는 소프트웨어)가 네트워크에 접근할 수 있도록 함. 사용자 인터페이스를 제공하고, 전자우편, 원격 파일 접근과 전송, 공유 데이터베이스 관리 및 여러 종류의 분산 정보 서비스를 제공함 |


**21. 동기화(synchronized)에 대하여 설명해 보세요.**
- 하나의 자원을 여러 태스크가 사용하려 할 때에, 한 시점에서 하나의 태스크만이 사용할 수 있도록 하는 것

**22. 프로세스와 쓰레드의 차이점에 대하여 말해 보세요.**
- 프로세스 : 실행 중인 프로그램, 자원(resources)과 쓰레드로 구성
- 쓰레드 : 프로세스 내에서 실제 작업을 수행하는 단위, 모든 프로세스는 하나 이상의 쓰레드를 가지고 있다.
- 다중 쓰레드 : 하나의 프로세스(프로그램)에 하나 이상의 쓰레드를 생성하여 실행

## 23. 코딩테스트

- for 문
- 클래스 정의
- strip, rstrip, lstrip
- split
- while 문
- list 인덱싱

---

**Q # 1) 객체지향이란?**

답변: 객체지향이란 실세계의 사물을 추상화하여 멤버 변수와 메소드를 정의하는데서 출발한다. 캡슐화를 통해 이와 같은 멤버변수와 메소드의 이용가능 범위를 적정하게 제한할 수 있고, 상속을 이용하여 부모 클래스의 기능을 자식클래스에서 물려받거나 재정의를 통해 다른 기능을 구현하는 다형성까지 포함된다.

**Q # 1-1) 객체지향 프로그래밍(OOP = Object-Oriented Programming)란 무엇인가?**

답변: 객체지향 프로그램이란 데이터를 객체로 취급하여 프로그램에 반영한 것이다. 순차적 실행이 아닌, 객체와 객체의 상호작용을 통해 동작한다.

**Q # 1-2) 객체지향의 특징은?**

답변: **다형성, 상속, 캡슐화, 추상화 의 특징을 가지고 있다.** ( 상추캡(이)다 )

**다형성** 은 하나의 메소드나 클래스가 다양한 방법으로 동작하는 것을 말한다. 상속을 통해 기능을 확장하거나 변경하는 것을 가능하게 해 주고, 같은 클래스 내의 코드 길이를 줄여준다. 오버라이딩이나 오버로딩을 통해서 한 요소에 여러 개념을 넣어 놓은 것이다.
**상속** 는 새로운 클래스가 기존의 클래스의 자료와 연산을 이용하게 해주는 것입니다. 공통적으로 필요한 성격을 가장 기본적인 클래스로 정의해두고, 상속받아 사용. 중복을 최소화한다.
**캡슐화** 는 객체가 맡은 역할을 수행하기 위한 하나의 목적을 위해 데이터와 기능들을 묶는 것을 말합니다.

public, protected, private 라는 접근 지정자를 통해 클래스에 담는 내용 중 중요한 데이터나 기능을 외부에서 접근하지 못하도록한다. 객체 외부에서는 내부 정보를 직접 접근하거나 조작할 수 없다. getter 와 setter 를 통해서만 접근 가능하다.

**추상화** 는 객체들의 공통적인 특징을 뽑아내는 것이다. 즉, 객체들의 공통적인 데이터와 기능을 도출하는 것이다. 예를 들면, Class.

- 객체지향 프로그래밍은 코드 재사용성이 높다.
- 코드의 변경과 수정이 용이하다.
- 직관적으로 코드를 분석할 수 있다.(블럭의 내부 구조가 어떤지 신경 쓸 필요가 없다 -- 캡슐화, 추상화)
- 유지보수가 쉬우므로 개발 속도를 향상한다.
- 상속을 통해 장점을 극대화한다.

**Q # 1-3) 객체 지향과 절차지향 프로그래밍과의 차이는?**

답변:

**절차지향, 구조적 프로그래밍(C)**

- 순차적 프로그래밍이다.
- 작업 순서대로 코딩한다.
- 함수 단위로 구성되며 기능별로 묶어놓은 특징이 있다.

**객체지향 프로그래밍(Java, C++, C#)**

- 주 구성요소는 클래스와 객체이다.
- 클래스를 활용해 각각의 기능별로 구성한다.

- 객체별로 개발 가능하다.
- 코드 재사용 가능하다.
- 오류 발생 가능성 적고 안정성 높다.

**Q # 2) 자바 언어에 대해 설명해보세요.**

답변: JAVA 는 네트워크상에서 쓸 수 있도록 미국의 선 마이크로 시스템즈가 개발한 객체 지향 프로그래밍 언어이다.

**Q # 2-1) 자바 언어의 특징은?**

답변: JAVA 는 자바가상머신(JVM)만 설치하면 컴퓨터의 운영체제에 상관없이 작동한다. 또한, 기본 자료형을 제외한 모든 요소들이 객체로 표현되고 객체 지향 개념의 특징인 캡슐화, 상속, 다형성이 잘 적용된 언어이다. 그 외에도 가비지 컬렉터를 통해 자동적인 메모리 관리를 하고 멀티쓰레드를 지원한다.

운영체제 없이 독립적 : JVM(Java Virtual Machine)

- 한 개 이상의 클래스로 구성
- 클래스는 한 개 이상의 필드나 메소드로 구성
- 맨 먼저 public static void main(String[ ] args) 를 찾아 실행 - 기본 자료형을 제외한 모든 요소가 객체로 표현
- 캡슐화, 상속, 다형성
- Garbage Collector 를 통한 메모리 관리기능
- 멀티스레드(Multi-thread = 다중동기화) 지원

**Q # 2-2) 자바에서 객체란 무엇인가?**

답변: 객체는 데이터(변수)와 그 데이터에 관련된 동작(함수), 절차, 방법, 기능을 모두 포함 한 것이다. 같은 성질, 같은 구조와 형태를 가지는 객체는 등급으로 정의하고 등급에 속하는 객체는 그 등급의 인스턴스이다.

예를 들면 기차역에서 승차권을 발매하는 경우, 실체인 '손님'과 동작인 '승차권 주문'은 하나의 객 체이며, 실체인 '역무원'과 동작인 '승차권 발매'도 하나의 객체이다.

**Q # 3) 자바 컴파일 과정을 설명하세요.**

답변: 컴파일러가 소스코드를 자바 바이트코드로(.class) 변환한다, JVM(자바가상머신)이 그 바이트코드를 기계어로 변환하여 인터프리터 방식으로 어플리케이션을 실행한다.

1. 전처리 : #define, #include 지시자 해석
2. 컴파일 : 고급 언어 소스 프로그램 입력받아 어셈블리 파일을 만듦
3. 어셈블 : 어셈블리 파일을 오브젝트 파일로 만듦
4. 링크 : 오브젝트 파일을 엮어 실행파일을 만들고 라이브러리 함수 연결
5. 실행

**Q # 3-1) 스크립트 언어와 컴파일 언어의 차이는?**

답변: **스크립트 언어와 컴파일 언어의 차이점은 컴파일러의 존재 여부이다.**
스크립트 언어는 실행될 때 바로 해석, 코드 수정 후 실행할 때마다 결과 바뀐다는 특징이 있다. 스크립트 언어에는 JavaScript, Python, Ruby 가 있다. 컴파일 언어는 한번 컴파일한 후, 코드 수정 후 다시 컴파일하기 전까지 결과 동일한다. 컴파일 언어에는 C, C++, C#, Java 등이 있다.

**Q # 4) JDBC (Java Database Connectivity) 란 무엇인가?**

답변: JAVA 를 통해 데이터베이스에 접근할 수 있는 프로그래밍 / DB 에 접근해(작업)
데이터를 조회/삽입/수정/삭제 가능하다.
이때, 연결해주는 응용프로그램 인터페이스인 JAVA API DBMS 종류에 따라 그에 맞는 JDBC 를 설치해야 한다.

**Q # 5) 다형성과 상속은 무엇인가?**

답변: 다형성(Polymorphism)과 상속(Inheritance)은 객체 지향 개발의 두 가지 핵심 개념이다.

**다형성** 이란 하나의 지시에 대해 여러 객체가 각자 다른 행위를 수행하는 것을 말한다. 다형성은 행동의 특정 타입에 대한 정의를 만들 수 있게 하고, 행동을 구현하는 수 많은 다른 클래스들을 갖게 한다.

**상속** 은 기존 클래스의 변수와 메소드를 모두 취하여 추가적인 기능도 가지는 클래스를 새로 만드는 것이다. 부모 클래스에서 클래스의 행동과 정의를 가져다 사용할 수 있게 해준다. 새로운 클래스를 정의할 때 이전에 정의된 클래스에서 정의와 상태를 상속할 수 있고 새로운 행동을 추가하거나 새로운 타입에 대한 행동을 오버라이드할 수 있다.

- 변수와 메소드를 그대로 쓰므로 코드중복을 줄인다.
- 객체를 사용하는 클래스 외부적인 관점으로 봤을 때 묶어서 처리(upcasting)할 수 있으므로 효율적인 코드가 된다.

### Q # 6) 인터페이스(Interface) 란?

답변: **인터페이스** 는 추상 클래스보다 추상화 정도가 높은 상태를 정의할 때 사용하며 기능의 재정의에 큰의미를 두고 있다. 껍데기만 있는 클래스로, 인터페이스를 implements 하는 클래스에게 틀을 제공한다.

- 일종의 추상 클래스로, 오직 상수와 추상 메소드만을 선언할 수 있다.
- Implements 키워드를 사용해서 상속을 진행하며 자식 클래스에서 반드시 메소드를 재정의해야한다.
- 자신을 implement 하는 클래스에게 메소드 구현을 강제 = 상속의 관계가 없는 클래스 간에 서로 같은 로직을 구현하게 한다.
- 예) List 인터페이스를 implement 하는 클래스들인 ArrayList, LinkedList 는 모두 add(), clear(), indexOf(), get() 등의 메소드를 가진다.
- 다중 상속 가능하다.

### Q # 7) 추상클래스(Abstract)란?

답변: **추상클래스** 는 abstract 을 이용한 미완성 메소드로 형태만 정의해두는 클래스를 말합니다. 추상 메소드를 하나 이상 가진 클래스로, 하위 클래스를 제어하기 위해 사용한다. 추상클래스는 상속을 진행하고 상속받은 자식클래스에서 반드시 재정의를 해야한다. 추상클래스는 상속을 통해 기능을 확장하는 데에 의미가 있다.

### Q # 7-1) 추상클래스와 인터페이스의 공통점과 차이점은?

**공통점**

- 선언만 있고 구현 내용이 없다.
- New 연산자로 새로운 인스턴스 생성 불가능하다.
- 프로토타입만 있는 메소드를 가진다.
- 사용하기 위해서는 하위클래스(자식클래스)에서 확장/구현해야 한다.

**차이점**

- 인터페이스는 메소드 선언만 가능하다. 클래스가 아니기 때문에 다중상속이 가능하다.
- 추상클래스는 일반 메소드 선언을 사용한다. 다중상속이 불가능하다.

### Q # 8) Static 이란?

답변: 클래스가 로딩될 때, 메모리 공간을 할당하는데 처음 설정된 메모리 공간이 변하지 않음을 의미 / 객체를 많이 만들어도 해당 변수는 하나만 존재(객체와 무관한 키워드)

Q # 9) Generic 이란?

답변: 클래스에서 사용할 타입을 클래스 외부에서 설정하는 것이다. 만들어져 있는 클래스를 원하는 형태로 사용할 수 있다.

- < > 안에 들어갈 수 있는 것은 참조자료형(클래스, 인터페이스, 배열)이다.
- 기본자료형을 사용하기 위해선 wrapper 클래스 이용한다.

**Q # 10) 자바 데이터 타입에 대해 설명하세요.**

답변:

Primitive type 기본형 : 변수에 값 자체를 저장(정수형, 실수형, 문자형, 논리형 등 8 가지) 한다. stack 메모리 영역에 실제 값을 저장한다.

Reference type 참조형 : 기본형 외의 모든 타입(String, 클래스, 인터페이스, 배열 등)이다. new 로 정의하고, 실제 값은 heap 에 저장하고 stack 에는 메모리 주소만 저장한다.

Q # 11) Thread 란?

답변: 프로세스(운영체제에서 실행 중인 하나의 프로그램으로 하나 이상의 쓰레드 포함) 내에서 동시에 실행되는 독립적인 실행 단위이다. 자원을 많이 사용하지 않고 구현이 쉬우며 범용성이 높다.

- 장점 :

- 빠른 프로세스 생성
- 적은 메모리 사용
- 쉬운 정보 공유

- 단점 :

- 자원을 공유하기때문에 교착상태(Deadlock = 다중 프로그래밍 체제에서 하나 또는 그 이상의 프로세스가 수행할 수 없는 어떤 특정 시간 을 기다리고 있는 상태) 에 빠질 수 있다.

Q # 12) Thread 와 Process 의 차이?

답변: 여러 개의 스레드가 하나의 프로세스이다.

**스레드** 는

각자의 스택 메모리 영역을 가지고 있다. 동일한 프로세스 내의 다른 스레드들과 전역 메모리를 공유한다. = CPU 로부터 새로운 자원을 할당받지 않아 도 되기 때문에 프로세스보다 실행 속도가 빠르다.

**프로세스** 는 실행 중인 프로그램을 말한다. CPU 가 메모리를 잘 분배
해줘야하며 운영체제의 성능에 따라 성능이 결정된다.

**Q # 13) 접근 제한자 (접근지정자)의 종류에 대해 설명하세요.**

답변:

Public : 같은 프로젝트 내에 어디서든 사용 가능하다.

Protected : 같은 패키지 내, 다른 패키지에서 상속받아 자손 클래스에서 접근 가능하다.

Default : 같은 패키지 내에서만 접근 가능

Private : 같은 클래스 내에서만 접근 가능

Public > protected > default > private

**Q # 14) Call by value(값에 의한 호출), Call by reference(레퍼런스에 의한 호출) 에 대해 설명하세요.**

**답변: Call by value** 는 값을 복사해서 새로운 함수로 넘기는 호출 방식이다. 원본값 변경하지 않는다.

**Call by reference** 는 주솟값을 인자로 전달하는 호출 방식이다. 원본값 변경이 가능하다.

자바언어는 Call by value 이다.

[참고] https://brunch.co.kr/@kd4/2

**Q # 15) String , StringBuffer, StringBuilder 란?**

답변: String

String 는 값 변경 불가능하다. 문자를 수정하려면 지우고 다시 새로 생성해야 한다.

수십번 String 이 더해지는 경우, 각 String 의 주솟값이 stack 에 쌓이고 클래스들은 Garbage Collector 가 호출되기 전까지 heap 에 지속적으로 쌓이게 된다. 메모리 관리적인 측면에서는 치명적이라고 볼 수 있다.
String 에서 저장되는 문자열은 알고보면 char 의 배열형태로 저장되며 이 값들은 외부에서 접 근할 수 없도록 private 으로 보호된다. 또한 final 형이기 때문에 초기값으로 주어진 String 의 값은 불변으로 바뀔 수가 없게 되는 것이다.

StringBuffer 는 값변경이 가능하며, 한번 만들고 필요할 때 크기를 변경하여 문자 변경한다. 멀티쓰레드 환경에서 동기화를 보장한다.

스트링 버퍼는 char 타입의 배열로 되어 있어서 한글자 한글자를 append 할 수 있다
name = name +
“홍”; 구문이 실행될 때 실제로는 스트링 버퍼를 새로 생성해서 name 이 가리키는 “길동”을 만들어주고 스트링 버퍼의 append 함수를 이용하여 “홍” 를 붙여준다. 그렇게 완성된 스트링 버퍼값을 메모리에 올리고 name 은 다시 이 값을 참조하게 된다. 그럼 그와중에 생겨난 메모리 안의 “길동” 이라는 값과 “홍”라는 값은 가비지 컬렉터가 가지고 있 다가 필요없으니 버린다. 그 짧은 순간에 이런 일처리가 진행이 되어서 스트링 버퍼를 사용 하는 것이 스트링 객체를 사용하는 것보다 빠르다.

StringBuilder 는 멀티스레드 상태에서 동기화 지원하지 않는다. 멀티스레드 환경에 부적합하나 싱글스레드에서 stringBuffer 보다 좋다.
[참고] https://jeong-pro.tistory.com/85

**Q # 16) 자바 메모리 영역에 대해 설명하세요.**

답변:
- 메소드 영역 : 바이트코드, static 변수, 전역변수, 코드에서 사용되는 Class 정보 등이 올

라감 / 코드에서 사용되는 class 들을 로더로 읽어 클래스별로 런타임 필드데이터, 메소드 데 이터 등을 분류해 저장
- 스택(Stack) : 매개변수, 지역변수, 함수() 등이 할당되는 LIFO(Last In First Out) 방식의 메모리 / 사용이 끝나면 바로 소멸, 컴파일 시 메모리를 할당

- 힙(Heap) : new 연산자를 통한 동적 할당된 객체들이 저장됨,

Garbage 컬렉션에 의해 메모리가 관리 / 호출이 끝나도 사라지지 않으며 프로그램 실행 시 동적으로 할당

**Q # 17) DAO (Data Access Object) 란?**

답변: 데이터베이스의 데이터에 접근을 위한 객체(DB 를 사용해 데이터를 조회하거나 조작하는 기능을 전담)이다.

데이터베이스에 접근을 하기위한 로직과 비즈니스 로직을 분리하기 위해서 사용한다.

**Q # 18) DTO (Data Transfer Object = VO(Value Object)) 란?**

답변: 각 계층간 데이터 교환을 위한 JavaBean(여기서 계층이란 Controller, View, Business Layer, Persistent Layer)

이다. VO 는 동일한 개념이지만 read only 속성을 가진다.

**Q # 19) 변수명 표기하는 방법?**

- 헝가리언 표기법 : 자료형을 식별자에 같이 포함 ex) inum; int int_num; 인터페이스명.

- 파스칼 표기법 : 식별자가 한 단어나 혹은 여러 단어로 조합(언더바 X), 각 단어의 첫 문 자만 대문자로 예)

KorScore

- 캐멀 표기법 : 모든 단어를 공백없이 조합(언더바 X), 첫 단어의 첫 문자는 소문자로 예) korScore

- 스네이크 표기법 : 예) eng_score 배열, 연결리스트의 차이점

**Q # 20) 배열과 연결리스트의 차이점에 대해 설명하라.**

답변:

**배열**

- 인덱스를 가진다.
- 데이터를 한번에 접근하여 접근 속도가 빠르다.
- 크기 변경 불가하다.
- 데이터를 삽입/삭제하면 다음 위치부터 모든 데이터의 위치를 변경해야 한다.

**연결리스트**

- 인덱스 대신 현재 위치의 이전/ 다음 위치를 기억한다.
- 한 번에 접근 불가, 연결된 링크를 따라가야만 접근 가능하여 배열보다 속도가 떨어진다.

- 크기가 가변적이다.
- 데이터 삽입/삭제는 논리적 주소만 바꿔준다.

**Q # 20-1) 어떨 때 배열을 사용하고, 인접리스트를 사용해야할까?**

답변: 배열은 데이터 양이 많지만 삽입/삭제 없으며, 데이터 접근이 빈번할 때 사용한다. 연결리스트는 데이터 양이 적고, 삽입/삭제가 빈번할 때 사용한다.

**Q # 21) ArrayList<> 란?**

답변: 배열의 확장판이다. 배열의 크기를 임의로 바꿀 수 있다. list 에 들어갈 데이터 타입 설정할 수있다. (add, remove, isEmpty, size, get, indexOf 등의 메소드가 있다.)

**Q # 22) Hash 란?**

답변: 내부적으로 배열을 사용(HashTable)하여 데이터를 저장하고, 검색 속도가 빠르다. 데이터 삽입과 삭제 시 기존 데이터를 밀어내거나 채우지 않고, 데이터와 연관된 고유한 숫자를 생성해 이를 인덱스로 사용한다.

**Q # 23) equals 와 HashCode 란?**

답변: **equals** 는 동일한 내용을 가진 객체인지를 비교한다. **hashcode** 는 동일한 객체인지 구별하기 위해 고유한 정숫값으로 출력한다.

**Q # 24) 스택과 큐의 차이는?**

답변: **Stack** 은 (LIFO = Last in First out) 함수를 호출 할 때, 현재 함수에서 사용되는 값을 스택에 넣고, 작업이 끝나면 함수를 리턴하고 스택에 넣었던 값을 꺼내는 방식으로 동작한다. **Queue** 는 (FIFO = First in First out) 이고, 프로세스 처리, CPU 관리, 프린터 큐 등에 사용한다.

**Q # 24-1) 스택과 힙의 차이는?**

답변: 메모리는 스택과 힙의 두 가지 주요 영역으로 구분된다.

스택(Stack) 은 지역변수, 함수(메서드) 등이 할당되는 LIFO(Last In First Out) 방식의 메모리이다.

힙(Heap) 은 new 연산자를 통한 동적 할당된 객체들이 저장되며, Garbage 컬렉션에 의해 메모리가 관리되어 진다.

[참고] 메서드 영역은 static 변수, 전역변수, 코드에서 사용되는 Class 정보 등이 올라간다. 코드에서 사용되는 class 들을 로더로 읽어 클래스별로 런타임 필드데이터, 메서드 데이터 등을 분류해 저장한다. 스택은 기본값, 객체의 참조, 메서드가 저장되는 위치다. 따라서 스택에 있는 변수의 생애주기는 코드의 스코프에 영향을 받는다. 스코프는 일반적으로 메서드 호출이나 for 문, while 문 같은 코드에서 괄호로 구분해서 정의한다. 일단 해당 스코프의 실행이 종료되면 스코프 안에 선언된 변수들은 스택에서 제거된다. 메서드를 호출했을 때 선언된 변수들은 스택의 상단에 위치한다. 스택에서 다른 메서드를 호출하면 스택은 새로운 메서드의 변수를 스택의 상단에 둔다.

**Q # 25) Overloading VS Overriding**

답변:

**Overloading (오버로딩)** 은 같은 이름의 메소드를 여러 개 정의하는 것이다. 매개변수의 타 입이 다르거나 개수가 달라야 한다. Return type 과 접근 지정자는 영향을 주지 않는다.
**Overriding (오버라이딩)** 은 상속에서 나온 개념이다. 상위 클래스(부모)의 메소드를 하위 클래스(자식 클래스)에서 재정의하는 것이다.

**Q # 26) 가비지 컬렉터란?**

답변: 가비지 컬렉터는 힙 내의 객체 중 가비지를 찾아내어 처리해서 힙의 메모리를 회수하는 작업을 하는 것이다.

JVM 이 메모리를 할당받고 프로그램들을 실행하다가 메모리가 부족해지면 OS 에게 추가로 메모리를 요청하는데, 이때 가비지 컬렉터가 실행된다.

**Q # 26-1) 가비지 컬렉션이란?**

답변: 정리되지 않은 메모리, 유효하지 않은 메모리 주소인 가비지를 정리해주는 프로그램이다.

시스템에서 더 이상 사용하지 않는 동적 할당된 메모리 블록을 찾아 자동으로 다시 사용 가능한 자원으로 회수하는 것이다.

[참고] 가비지 컬렉션 은 기존에 할당된 메모리를 재사용하는 메커니즘으로 나중에 메모리를 할당할 때 재사용할 수 있다. 가비지 컬렉션을 사용하면 메모리를 직접 해제할 필요가 없다. 자바에서는 new 키워드로 새로운 객체를 생성한다면 JVM 은 해당 객체에 저장된 데이터에 적절한 양의 메모리를 할당한다. 객체가 더 이상 필요하지 않으면 JVM 은 메모리 공간을 재배치해야 한다. 그래서 생성된 다른 객체가 해당 메모리 공간을 사용하게 된다. c 나 c++같은 프로그래밍 언어에서는 malloc 와 free 함수를 호출해서 이런 메모리 할당을 수동으로 직접 관리해야 했다. 자바와 c#같은 대부분의 모던 프로그래밍 언어에는 효율을 높이기 위해 메모리를 자동으로 관리하는 시스템이 있다. 덕분에 프로그래머는 잠재적으로 발생할 수 있는 모든 실수를 줄일 수 있다.

**Q # 26-2) 가비지란?**

답변: 정리되지 않은 메모리, 유효하지 않은 메모리 주소로 참조하던 관계가 끊어지면서 어떠한 것도 참조하지 않는 값이 되버리는 것을 말한다.

**Q # 26-3) 가비지 컬렉션이 일어나는 장소는?**

답변: 실행 중인 자바 가상 머신(JVM) 내부에서 일어난다. JVM 은 OS 아래에서 발생한다.프로그램 실행시 OS 에서 할당받은 메모리를 JVM 이 사용한다.

**Q # 26-4) 가비지 컬렉터가 가비지 객체를 어떻게 판단하나요? (= 사용하지 않거나 사용할 수 없는 메모리를 어떻게 판단할까요?)**

답변: 가비지 컬렉터는 가비지 객체를 판단하기 위해 reachability 개념을 사용한다. 어떤 객체에 유효한 참조가 있으면 reachable, 없으면 unreachable 로 구별하고 unreachable 객체를 가비지로 간주한다.

**Q # 27) 프레임워크에 대해 설명해주세요.**

답변: 소프트웨어를 만들 때 뼈대가 되는 부분을 미리 구현한 클래스, 인터페이스, 메서드 등의 모음이다.

**Q # 27-1) 프레임워크를 사용했을 때의 장점과 단점은?**

답변: 장점은 미리 구현해 둔 코드를 쓰기 때문에 빨리 만들 수 있고, 품질이 보장되어 있고, 사용하기 쉽습니다. 단점은익숙해지는 데에 시간이 걸릴 수 있고, 유연성이 부족합니다. 또한, 언어가 아닌 프레임워크를 배우게 된다는 부작용이 있습니다.

Q # 27-2) Spring 프레임워크를 사용하는 이유?

답변:

EJB(Enterprise JavaBeans = 기업환경의 시스템을 구현하기 위한 서버측 클라이언트 모델) 에 비해 가볍기 때문에 엔터프라이즈급의 시스템을 더 빠른 시간에 작성 가능하다.
기존의 프레임워크들은 웹, 또는 하드웨어, 데이터베이스등 전문적인 영역만 지원하는 경우가 많은데, 스프링은 어느 한 분야에 집중하기 보다, 전체를 설계하는 용도로 사용한다. 스프링은 전체 구조에 집중했기에 특정영역의 프레임워크와 공존하는 방식으로 사용 가능하다.

[참고] Spring 프레임워크의 특징

- POJO(Plain Old Java Object)의 구성만으로 가능하도록 제작(따로 프레임워크의 사용을 위해 공부할 필요가 없다.)
- 의존성 주입(DI)를 통한 객체간의 관계구성
- AOP(Aspect oriented programming)을 지원한다. 공통된 부분등, 특정 부분들(핵심 관심사) 로 나누어 관리함으로써,프로그램을 모듈화 하여 개발 및 유지보수가 편하다.
- 트랜잭션을 지원한다. 스프링은 이를 어노테이션이나 xml 로 설정 가능하다.

Q # 28) Class 와 Instance 의 차이점은 무엇인가?

답변: Class(클래스) 는 어떤 특정 종류의 모든 객체들에 대해 일반적으로 적용할 수 있는 변수와 메소드를 정의하고 있는 소프트웨어적인 설계도라 할 수 있다. 실세계에 존재하는 객체들이 가질 수 있는 상태와 행동 들에 대해 소프트웨어적으로 추상화(abstraction) 해 놓은 것이다. Instance(인스턴스) 는 클래스에 대해 선언, 생성되는 변수이며 메모리 공간을 차지하게 된다. 인스턴스의 메소드를 이용하여 변수들의 값을 설정 및 변경할 수 있다.

**Object(객체)는 소프트웨어 세계에 구현할 대상이고, 이를 구현하기 위한 설계도가 Class(클래스)이며 이 설계도에 따라 소프트웨어 세계에 구현된 실체가 Instance(인스턴스)이다.**

Q # 29) Lambda Expressions 에 대해 설명하시오.

답변: Lambda Expressions(람다식) 은 식별자 없이 실행 가능한 함수 표현식이다. 자바 8 의 가장 특징적인 기능으로 기존의 불필요한 코드를 줄이고 가독성을 향상시키는 것에 목적을 두고 있다. 대표적으로 반복문, 비교문이 있다. 기본적으로 람다식을 위한 인터페이스에서 추상 메소드는 단 하나여야 하며, 이 인터페이스는 람다식을 위한 것이다라는 표현을 위해 @FuctionalInterface 를 사용한다. Stream API 의 존재를 알아두면 좋다.

**Q # 30) Vector 와 ArrayList 에 대해 설명하시오.**

답변:

Vector

- 필요에 따라 크기를 동적으로 조절할 수 있는 동적배열 구현한다.
- 배열과 마찬가지로 정수 인덱스를 이용해 배열에 액세스 가능하다.
- 동기화 되어 있으면 한 번에 하나의 스레드만 벡터의 메소드를 호출 가능하다.

ArrayList

- Collections 프레임워크의 일부, java.util 패키지 내에 존재한다.
- Object 배열을 이용해서 데이터를 순차적으로 저장한다.
- 배열에 더 이상 저장할 공간이 없으면 보다 큰 새로운 배열을 생성해서 기존의 배열에 저장된 내용을 새로운 배열로 복사한 다음에 저장한다.
- 선언된 배열의 타입이 모든 객체의 최고 조상인 Object 이기 때문에 모든 종류의 객체를 담을 수 있다.

**Q # 30-1) Vector 와 ArrayList 의 공통점과 차이점은?**

답변: 공통점은 크기가 동적인 배열을 사용할 때 주로 사용한다. 차이점은 Vector 는 동기식(한 스레드가 벡터 작업 중이면 다른 스레드가 벡터를 보유할 수 없음) 방식이고, ArrayList 는 비동기식(여러 스레드가 ArrayList 에서 동시에 작업할 수 있음, Vector 보다 빠름) 이다.

**1) 운영 체제 란 무엇입니까?**

운영 체제는 컴퓨터 하드웨어가 컴퓨터 소프트웨어와 통신하고 작동하도록하는 소프트웨어 프로그램이다.

**2) 운영 체제의 주요 목적은 무엇입니까?**

운영 체제에는 두 가지 주요 목적이 있다.

1. 컴퓨터 시스템의 계산 활동을 관리하여 컴퓨터 시스템이 제대로 작동하도록 한다.

2. 프로그램 개발 및 실행을 위한 환경을 제공한다.

**3) 프로세스란 무엇을 의미합니까?**

컴퓨터에서 실행되고 있는 프로그램을 프로세스라고 한다.

두 가지 유형의 프로세스가 있다. 운영 체제 프로세스, 사용자 프로세스

### 3-1) 프로세스의 특징을 설명하세요.

- 프로세스는 각각 독립된 메모리 영역(Code, Data, Stack, Heap 의 구조)을 할당받는다.

- 기본적으로 프로세스당 최소 1 개의 스레드(메인 스레드)를 가지고 있다.

- 각 프로세스는 별도의 주소 공간에서 실행되며, 한 프로세스는 다른 프로세스의 변수나 자료구조에 접근할 수 없다.

- 한 프로세스가 다른 프로세스의 자원에 접근하려면 프로세스 간의 통신(IPC, inter-process communication)을 사용해야 한다. (Ex. 파이프, 파일, 소켓 등을 이용한 통신 방법 이용)

## 4) 스레드(Thread)는 무엇입니까?

스레드는 CPU 사용의 기본 단위이다. 프로세스 내에서 실행되는 여러 흐름의 단위를 말한다.

스레드는 스레드 ID, 프로그램 카운터, 레지스터 세트 및 스택으로 구성된다.

[참고] 운영체제는 자원을 효율적으로 사용하려고 한다. 스레드를 사용하면 프로세스보다 생성할 때 오버헤드도 적고 공유된 자원에 대해서도 오버헤드가 적다. 그리고 스레드를 이용하면 병렬성을 높일 수 있다.

### 4-1) 스레드의 특징을 설명하세요.

- 스레드는 프로세스 내에서 각각 Stack 만 따로 할당받고 Code, Data, Heap 영역은 공유한다.

- 프로세스 내의 주소 공간이나 자원들(힙 공간 등)을 같은 프로세스 내에 스레드끼리 공유하면서 실행된다.

- 같은 프로세스 안에 있는 여러 스레드들은 같은 힙 공간을 공유한다. 반면에 프로세스는 다른 프로세스의 메모리에 직접 접근할 수 없다.

- 각각의 스레드는 별도의 레지스터와 스택을 갖고 있지만, 힙 메모리는 서로 읽고 쓸 수 있다.

- 한 스레드가 프로세스 자원을 변경하면, 다른 이웃 스레드(sibling thread)도 그 변경 결과를 즉시 볼 수 있다.

### 4-2) 프로세스와 스레드의 차이는 무엇입니까?

프로세스는 운영체제로부터 자원을 할당받는 작업의 단위이고, 스레드는 프로세스가 할당받은 자원을 이용하는 실행의 단위 이다. 프로세스는 운영체제로부터 메모리, 주소 공간 등을 할당받고 쓰레드는 할당받은 자원들을 내부 스레드끼리 공유하면서 실행된다.

[정리]

- 프로세스는 실행 중인 프로그램으로 다른 프로세스와 상관없이 독립적으로 자원을 할당 받는다.

- 스레드는 경량화된 프로세스로 프로세스 안에 존재한다. 각 스레드는 별도의 레지스터와 스택을 갖고, 힙 영역은 공유한다.

스레드를 사용하는 이유는 운영체제에서 더 효율적으로 시스템 자원을 관리하기 위해 사용된다고 할 수 있다. 멀티 프로세스로 진행되는 작업을 멀티 쓰레드로 수행하게 되면 시스템 콜이 줄어들기 때문에, 자원을 효율적으로 관리할 수 있고 프로세스의 통신비용보다 쓰레드간의 통신 비용이 적다는 이점도 있다.

단 스레드간의 자원공유는 전역변수를 이용하므로 동기화 문제에 신경을 써야하며 멀티스레드 프로그래밍은 프로그래머의 주의를 요구한다.

**4-3) 스레드의 장점?**

- 스레드는 프로세스보다 생성 및 종료시간, 스레드간 전환시간이 짧다.

- 스레드는 프로세스의 메모리, 자원등을 공유하므로 커널의 도움없이 상호간의 통신이 가능하다.

**4-4) 멀티 스레딩(Multi-threading) 의 장점과 단점은?**

- 멀티 스레딩이란 하나의 프로세스를 다수의 스레드로 만들어 실행하는 것

장점)

- 하나의 프로세스 내에 다수의 실행 단위들이 존재하여 작업의 수행에 필요한 자원들을 공유하기 때문에 자원의 생성과 관리가 중복되는 것을 줄일 수 있다.

단점)

- 교착상태를 발생시킬 수 있다.

- 동기화에 주의해야한다.

**4-5) 멀티 프로세스 대신 멀티 스레드를 사용하는 이유는 무엇입니까?**

- 쉽게 설명하면, 프로그램을 여러 개 키는 것보다 하나의 프로그램 안에서 여러 작업을 해결하는 것이다.

- 자원의 효율성 증대

: 멀티 프로세스로 실행되는 작업을 멀티 스레드로 실행할 경우, 프로세스를 생성하여 자원을 할당하는 시스템 콜이 줄어들어 자원을 효율적으로 관리할 수 있다. (프로세스 간의 Context Switching 시 단순히 CPU 레지스터 교체 뿐만 아니라 RAM 과 CPU 사이의 캐시 메모리에 대한 데이터까지 초기화되므로 오버헤드가 크기 때문)

: 스레드는 프로세스 내의 메모리를 공유하기 때문에 독립적인 프로세스와 달리 스레드 간 데이터를 주고받는 것이 간단해지고 시스템 자원 소모가 줄어들게 된다.

- 처리 비용 감소 및 응답 시간 단축

: 프로세스 간의 통신(IPC)보다 스레드 간의 통신의 비용이 적으므로 작업들 간의 통신의 부담이 줄어든다. (스레드는 Stack 영역을 제외한 모든 메모리를 공유하기 때문)

: 프로세스 간의 전환 속도보다 스레드 간의 전환 속도가 빠르다. (Context Switching 시 스레드는 Stack 영역만 처리하기 때문)

**4-6) 멀티 프로세싱과 멀티프로그래밍의 차이는?**

멀티 프로세싱은 여러개의 처리장치(CPU)를 장착하여 동시에 여러 작업을 병렬로 실행하는 방법.

멀티 프로그래밍은 다수 개의 프로그램의 같이 주기억장치에 있도록 한 방식.

**5) 소켓이란 무엇입니까?**

소켓은 두 응용 프로그램을 연결하는 데 사용된다. 연결의 끝점을 소켓이라고 한다.

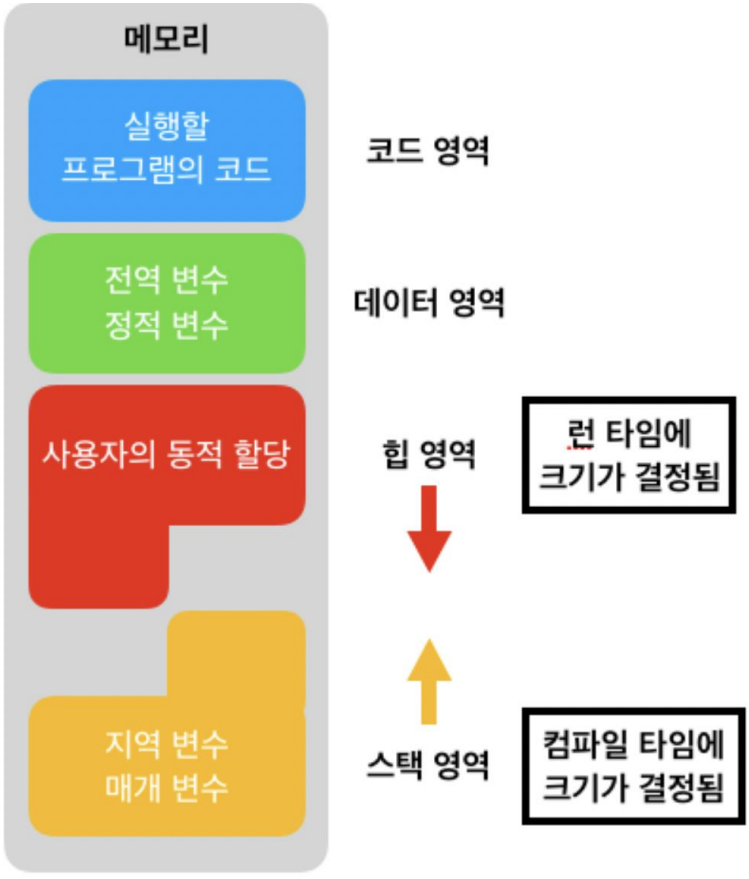
### 6) 커널이란 무엇입니까?

커널은 OS 의 모든 부분에 대한 기본 서비스를 제공하는 컴퓨터 운영 체제의 핵심이자 가장 중요한 부분이다.

### 7) 힙 영역과 스택영역의 차이점을 설명하시오.

프로그램이 실행되기 위해 프로그램이 메모리에 로드가 되어야한다. 따라서 운영체제에서 프로그램의 실행을 위해 다향한 메모리 공간을 제공한다. 코드, 데이터, 스택, 힙 영역이 할당되고 각 역할은 다음과 같다.

- 코드 : 실행할 프로그램의 코드가 저장되는 텍스트 영역이다. CPU 는 코드영역에서 저장된 명령어를 하나씩 가져가서 처리한다.
- 데이터 : 전역변수와 정적변수가 이해 해당된다. 프로그램의 시작과 함께 할당되며 프로그램이 종료되면 소멸된다.
- 스택 : 스택영역은 함수의 호출과 관계되는 지역변수와 매개변수가 저장되는 영역이다. 함수의 호출과 함께 할당되며, 함수의 호출이 종료될때 해제된다.
- 힙 : 힙 영역은 사용자가 직접 관리할 수 있는 메모리 영역이다. 힙 영역은 사용자에 의해 메모리공간이 동적으로 할당되고 해제된다.




### 8) 운영 체제에서 페이징을 사용하는 것은 무엇입니까?

페이징은 운영 체제에서 외부 조각화 문제를 해결하는 데 사용된다. 이 기술을 사용하면 필요한 데이터를 최대한 빨리 사용할 수 있다.

**9) 멀티 프로세서 시스템의 장점은 무엇입니까?**

프로세서가 많을수록 처리량이 크게 증가한다. 리소스를 공유 할 수 있기 때문에 비용 효과적이다. 따라서 전반적인 안정성이 향상된다.

**10) 가상 메모리(Virtual Memory)란 무엇입니까?**

가상 메모리는 프로세스가 메모리 외부에서 실행될 수 있도록하는 매우 유용한 메모리 관리 기술이다. 이 기술은 실행 프로그램이 실제 메모리에 맞지 않을 때 특히 사용된다.

**11) 교착 상태(DeadLock)이란?**

두 개 이상의 프로세스나 스레드가 서로 자원을 기다리면서 무한히 기다리게 되는 상태를 말한다.

교착 상태는 두 개 이상의 작업이 서로 상대방의 작업이 끝나기 만을 기다리고 있기 때문에, 다음 단계로 진행하지 못하는 상태이다. 배치 처리 시스템에서는 일어나지 않는 문제이다. 프로세스, 스레드 둘다 이와 같은 상태가 일어날 수 있다.

**11-1) 교착 상태의 4 가지 필요 조건은 무엇입니까?**

다음과 같은 4 가지 조건이 있다.

1. 상호 배제(Mutual Exclusion) : 한 자원에 대한 여러 프로세스의 동시 접근은 불가능하다. 즉, 하나의 자원을 특정 시기에 하나의 프로세스나 스레드만 소유할 수 있는 형태.

2. 점유와 대기(Hold and Wait) : 하나의 자원을 소유하고 다른 프로세스 혹은 스레드의 자원을 요청하는 상태이다.

3. 비선점(Non preemptive) : 하나의 프로세스나 스레드에게 주어진 자원은 해당 프로세스나 스레드가 스스로 놓기 전에는 놓게 만들 수 없는 상태. 즉, 다른 프로세스에서 자원을 사용하는 동안 자원을 강제로 가져올 수 없다.

4. 환형 대기(Circle wait) : 각 프로세스가 다음 프로세스가 요구하는 자원을 가지고 있는 것을 말한다. 두 개의 프로세스나 스레드의 경우, A -> B, B -> C, C -> A 에게 서로 자원을 요청하고 기다리는 상황

**11-2) 교착 상태 해결 방법을 말하시오.**

위의 4 가지 조건들 중 하나라도 제거하면 된다. 예방, 회피, 탐지, 복구방법이 있다.

공유 자원 중 많은 경우가 한 번에 한 프로세스만 사용할 수 있기 때문에 상호배제 조건은 제거하기 어렵다. 대부분의 교착상태 방지 알고리즘은 순환대기가 발생하는 일을 막는데 초점이 맞춰져 있다.

1. 예방(Prevention) : 교착상태가 발생하지 않도록 하는 것

2. 회피(Avoidance) : 교착상태를 피하는 것. Ex) Dijkstra 의 Banker's Algorithm

3. 탐지(Detection) : 교착상태가 발생하면 탐지 하는 것, 복구를 수반

4. 복구(Recovery) : 프로세스 중지, 자원 선점(희생자의 선택-> 기아, 복귀)

**11-3) Banker's algorithm 은 무엇입니까?**

교착 상태를 피하기 위해 뱅커 알고리즘이 사용된다. E,J, Dijkstra 가 제안한 방법으로, 은행에서 모든 고객의 요구가 충족되도록 현금을 할당하는데서 유래한 기법이다. 프로세스가 자원을 요구할 때 시스템은 자원을 할당한 후에도 안정 상태로 남아있게 되는지를 사전에 검사하여 교착 상태를 회피하는 기법이다. 안정 상태에 있으면 자원을 할당하고, 그렇지 않으면 다른 프로세스들이 자원을 해지할 때까지 대기한다. 은행은 더 이상 모든 고객의 요구 사항을 충족시킬 수없는 방식으로 가용 현금을 할당하지 않는 뱅킹 시스템에서 뱅커 알고리즘이라고 한다.

**12) RAID 란 무엇입니까? RAID 의 종류에 대해 말하시오.**

RAID 는 Redundant Array of Independent Disks 의 약자이다. 전체 성능을 향상시키기 위해 동일한 데이터를 중복 저장하는 데 사용된다.

- RAID 0-내결함성이없는 스트립 디스크 어레이
- RAID 1-미러링 및 이중화
- RAID 2-메모리 스타일 오류 수정 코드
- RAID 3-비트 인터리브 패리
- RAID 4-블록 인터리브 패리티
- RAID 5-블록 인터리브 분산 패리티
- RAID 6-P + Q 이중화

**13) Sync 와 Async 의 차이는 무엇입니까?**

일반적으로 동기와 비동기의 차이는 메소드를 실행시킴과 동시에 반환 값이 기대되는 경우를 동기 라고 표현하고 그렇지 않은 경우에 대해서 비동기 라고 표현한다. 동시에라는 말은 실행되었을 때 값이 반환되기 전까지는

blocking 되어 있다는 것을 의미한다. 비동기의 경우, blocking 되지 않고 이벤트 큐에 넣거나 백그라운드 스레드에게 해당 task 를 위임하고 바로 다음 코드를 실행하기 때문에 기대되는 값이 바로 반환되지 않는다.

### 14) 스풀링이란 무엇입니까?

스풀링은 장치, 프로그램 또는 시스템에서 데이터를 사용하고 실행하기 위해 일시적으로 데이터를 수집하는 프로세스이다. 인쇄와 관련이 있다. 다른 응용 프로그램이 동시에 출력을 프린터로 보내면 스풀링은 이러한 모든 작업을 디스크 파일에 보관하고 프린터에 따라 대기열에 넣는다.

### 15) 뮤텍스란 무엇입니까?

프로세스 혹은 스레드 간의 통신 시에 shared memory 등을 쓰는 경우 하나의 자원에 두 개 이상의 프로세스 혹은 스레드가 접근하는 경우에 문제가 발생한다. 이를 제어하기위해 스레드는 뮤텍스를 사용하고, 프로세스에서는 세마포어를 사용한다.

- 뮤텍스 : 상호배제라고도 하며, Critical Section 을 가진 스레드의 Running Time 이 서로 겹치지 않도록 각각 단독으로 실행하게 하는 기술이다. 뮤텍스는 상태가 0, 1 두개 뿐인 이진 세마포어. synchronized 또는 lock 을 통해 해결한다.

#### 15-1) 세마포어란 무엇입니까?

세마포어는 사용중인 리소스를 잠그는 데 사용되는 보호 된 변수 또는 추상 데이터 유형이다. 공유된 자원의 데이터를 여러 '프로세스'에서 접근하는 것을 막는다. 세마포어의 값은 공통 자원의 상태를 나타냅니다. 리소스 상태를 나타내는 간단한 카운터이다. 공유 리소스에 접근할 수 있는 프로세스의 최대 허용치만큼 동시에 사용자가 접근하여 사용할 수 있다.

#### 15-2) 뮤텍스와 세마포어의 차이?

가장 큰 차이는 동기화 대상의 갯수이다. 뮤텍스는 동기화 대상이 하나뿐이고, 세마포어는 동기화 대상이 하나 이상일 때.

### 16) 운영 체제에서 기아(Starvation) 란 무엇입니까?

특정 프로세스의 우선순위가 낮아서 원하는 자원을 계속 할당 받지 못하는 상태이다.

기아상태는 자원 관리 문제이다. 이 문제에서 대기중인 프로세스는 리소스가 다른 프로세스에 할당되어 있기 때문에 오랫동안 필요한 리소스를 얻지 못한다.

### 17) 운영 체제에서 에이징(Aging)는 무엇입니까?

에이징은 자원 스케줄링 시스템에서 기아를 방지하기 위해 사용되는 기술이다. 특정 프로세스의 우선순위가 낮아 무한정 기다리게되는 경우, 한번 양보하거나 기다린 시간에 비례하여 일정 시간이 지나면 우선순위를 한 단계씩 높여 가까운 시간 안에 자원을 할당받도록 하는 기법을 말한다.

### 18) 운영 체제에서 페이징(Paging)은 무엇입니까?

세그먼테이션과 가상 메모리를 고정된 크기로 나누어 메모리(가상메모리)를 관리하는 기법을 말한다. 자세하게 말하자면 커다란 크기의 작업을 일정한 크기로 나누어 잘게 쪼개어 처리하는 것이다. 따라서 불연속적인 메모리 요청 등에 유연하게 처리할 수 있다.

세그먼테이션은 논리적 블록을 필요에 따라 다른 크기로 할당한 것이라면, 페이징은 고정된 크기로 나누는 것이다. 외부단편화는 해결하지만, 내부단편화가 발생할 수있다.

#### 18-1) 페이징의 장점과 단점은?

장점: 메모리를 페이지단위로 가져와서, 프로세스의 효율적인 운영이 가능하다.

단점: 페이지 크기별, 단위별로 페이지 폴트 현상이 발생할 수 있다.

### 19) 세그먼테이션(Segmentation) 이란?

메모리를 서로 크기가 다른 논리적인 블록 단위인 세그먼트(segment)로 분할하고 메모리를 할당하여 물리 주소를 논리 주소로 변환하는 것을 말한다. 미리 분할하는 것이 아니라 메모리를 사용할 시점에 할당된다. 내부단편화는 없지만 외부단편화가 발생할 수 있다.

### 20) 멀티 스레드 프로그래밍의 장점은 무엇입니까?

프로세스를 이용하여 동시에 처리하던 일을 스레드로 구현할 경우 메모리 공간과 시스템 자원 소모가 줄어들게 된다.

스레드 간의 통신이 필요한 경우에도 별도의 자원을 이용하는 것이 아니라 전역 변수의 공간 또는 동적으로 할당된 공간인 Heap(힙) 영역을 이용하여 데이터를 주고받을 수 있다. 그렇기 때문에 프로세스 간 통신 방법에 비해 스레드 간의 통신 방법이 훨씬 간단하다. 심지어 스레드의 context switch(문맥교환)는 프로세스 context switch

와는 달리 캐시 메모리를 비울 필요가 없기 때문에 더 빠르다. 따라서 시스템의 성능이 향상되고 자원 소모가 줄어들며 자연스럽게 프로그램의 응답 시간이 단축된다. 이러한 장점 때문에 여러 프로세스로 할 수 있는 작업들을 하나의 프로세스에서 스레드로 나눠 수행하는 것이다.

- 사용자의 반응을 향상시킨다.(= 응답성이 좋다 : 하나의 프로세스에 여러 스레드를 생성하여 스레드에 각기 다른 작업을 하게 함으로써 특정 작업을 하면서도 사용자로부터 명령을 입력받게 할 수 있다.)

- 프로세스 내 리소스 공유하여 경제적이다. (= 자원공유를 효율적으로 할 수 있다.)

- 작업이 분리되어 코드가 간결해진다.

**20-1) 멀티 스레드 프로그래밍의 단점은 무엇입니까?**

멀티 프로세스 기반으로 프로그래밍할 때는 프로세스 간 공유하는 자원이 없기 때문에 동일한 자원에 동시에 접근하는 일이 없었지만 멀티 스레딩을 기반으로 프로그래밍할 때는 이 부분을 신경써줘야 한다. 서로 다른 스레드가 데이터와 힙 영역을 공유하기 때문에 어떤 스레드가 다른 스레드에서 사용중인 변수나 자료구조에 접근하여 엉뚱한 값을 읽어오거나 수정할 수 있다.

그렇기 때문에 멀티스레딩 환경에서는 동기화 작업이 필요하다. 동기화를 통해 작업 처리 순서를 컨트롤 하고 공유 자원에 대한 접근을 컨트롤 하는 것이다. 하지만 이로 인해 병목현상이 발생하여 성능이 저하될 가능성이 높다. 그러므로 과도한 락으로 인한 병목현상을 줄여야 한다.

- 구현하기 어렵고, 테스트와 디버깅이 어렵다.

- 전체 프로세스에 영향을 줄 수 있다.

- 성능 저하가 동반됩니다.

- 동기화 작업이 필요하다.

- 교착상태가 발생하지 않도록 주의해야한다 .

### 21) Scheduling 이란(Process Scheduling)?

여러 프로세스가 있고, 이 프로세스들이 자원(CPU 등)을 동시에 요구하는데 자원이 제한되어 있다. 그러면 제한된 자원들을 어떻게(순서를 할당하는 등) 나눠줄 것인지에 대한 정책을 말한다.

### 22) CPU Scheduling?

CPU 하나는 동시에 여러개의 프로세스를 처리할 수 없기 때문에, 한 순간에 어떤 프로세스가 CPU 를 사용할 수 있게 하는지 결정하는 정책이다.

#### 22-1) CPU 스케줄링은 언제 발생하는가?

- 실행상태에서 대기상태로 전환될 때 (예, 입출력 요청) - Non preemptive(비선점)
- 실행상태에서 준비상태로 전환될 때 (예, 인터럽트 발생) - preemptive(선점)
- 대기상태에서 준비상태로 전환될 때(예, 입출력이 종료될 때)
- 종료될 때(Terminated)

#### 22-2) CPU 스케줄링의 종류를 설명하시오.

비선점(Non-preemptive) 스케줄링

- 이미 할당된 CPU 를 다른 프로세스가 강제로 빼앗아 사용할 수 없는 스케줄링 기법이다.

- 프로세스가 CPU 를 할당받으면 해당 프로세스가 완료될 때까지 CPU 를 사용한다.

- 일괄 처리 방식의 스케줄링(공정하지만 긴급 응답을 요하는 작업에 좋지 않다.)

- FCFS(FIFO) : 준비상태 큐에 도착한 순서에 따라 CPU 를 할당하는 기법. 공평성은 유지되지만 짧은 작업이 긴 작업을, 중요한 작업이 중요하지 않은 작업을 기다리게 됨.
  - 장점 : 평균 응답시간이 길다. (대화식 시스템에 부적합)
  - 단점 : 도착 순서에 따라 공평하다.
- SJF(Shortest Job First) : 실행시간이 가장 짧은 프로세스에 먼저 CPU 를 할당하는 기법. 가장 적은 평균 대기시간을 제공하는 최적 알고리즘
  - 장점 : 평균 응답 시간을 최소화 할 수 있다.
  - 단점 : 실행시간이 긴 프로세스는 CPU 를 할당받지 못하고 무한히 대기하는 현상 발생(starvation)

- HRN(Highest Response ratio) : 실행 시간이 긴 프로세스에 불리한 SJF 기법을 보완하기 위한 것으로 우선순위 계산 결과 값이 높은 것부터 우선순위가 부여된다. 대기 시간이 길수록 계산 결과가 높다. 우선순위 = (대기시간 + 서비스시간 / 서비스시간) 큰 프로세스일수록 우선순위가 낮으므로 평균 응답시간도 단축
- **기한부**(DeadLine) : 프로세스에게 일정한 시간을 주어 그 시간 안에 프로세스를 완료하도록 하는 기법
- **우선순위**(Priority) : 준비상태 큐에서 기다리는 각 프로세스마다 우선순위를 부여하여 그 중 가장 높은 프로세스에게 먼저 CPU 를 할당하는 기법. 정적, 동적 우선순위 방법 존재

선점(Preemptive) 스케줄링

- 하나의 프로세스가 CPU 를 할당받아 실행 하고 있을 때 우선순위가 높은 프로세스가 CPU 를 강제로 빼앗아 사용할 수 있는 스케줄링 기법

- 선점으로 인한 많은 오버헤드가 발생한다.

- 시분할 시스템에 사용하는 스케줄링이다. (긴급을 요하는 우선순위를 갖는 시분할 처리, 실시간 처리에 유용)

- 선점을 위해 시간 배당을 위한 인터럽트용 타이머 클럭(Clock)이 필요한다.

SRT(Shortest Remaining Time) : 현재 실행 중인 프로세스의 남은 시간과 대기 큐에 프로세스의 실행시간이 가장 짧은 프로세스에게 CPU 를 할당하는 기법 (비선점 기법인 SJF 알고리즘의 선점 형태로 변경한 기법)

  - 단점 : 잦은 선점으로 인한 문맥교환의 부담, starvation 의 위험
- **선점 우선순위** : 준비상태 큐의 프로세스들 중에서 우선순위가 가장 높은 프로세스에게 먼저 CPU 를 할당하는 기법
- RR(Round Robin) : 시분할 시스템을 위해 고안된 방법으로, FCFS 알고리즘을 선점 형태로 변형한 기법. 대기 큐를 사용하여 먼저 대기한 작업이 먼저 CPU 를 사용한다.
  - 단점 : CPU 를 사용할 수 있는 시간(Quantum)동안 CPU 를 사용한 후에 다시 대기 큐의 가장되로 배치된다. 할당되는 시간이 클 경우 FCFS 기법과 같아지고, 시간이 작을 경우 문맥교환 및 오버헤드가 자주 발생됨
- MLQ**(다단계 큐)** : 프로세스를 특정 그룹으로 분류할 수 있는 경우 그룹에 따라 각기 다른 준비상태 큐를 사용한다. 작업들을 여러 종류의 그룹으로 분할. 큐들간에 프로세스 이동이 불가능하다. 각 큐는 자신만의 독자적인 스케줄링을 가진다. 상위 우선 순위의 큐가 Empty 이면 하위 우선순위의 큐의 프로세스가 수행된다.
- MLFQ**(다단계 피드백 큐)** : 특정 그룹의 준비상태 큐에 들어간 프로세스가 다른 준비상태 큐로 이동할 수 없는 다단계 큐 기법을 준비상태 큐 사이를 이동할 수 있도록 개선한 기법. 새로운 프로세스는 높은 우선순위, 프로세스의 실행이 길어질 수록 점점 낮은 우선순위 큐로 이동. 제일 마지막 단계에서는 RR/FCFS 처리. 우선순위가 높은 단계의 큐일수록 시간 할당량을 작게 설정한다. 기아 상태를 예방하는 Aging 방법. 현대 OS 에서 RR 방식과 함께 가장 많이 사용되는 스케줄링 기법.
- RM(Rate Monotonic, 주기단조) 알고리즘 : 수행 주기가 가장 짧은 프로세스에 가장 높은 우선순위를 부여하는 실시간 스케줄링 알고리즘. 정적 스케줄링방식. 마감 시간과 주기가 일치.
  - 장점 : 간단, 사용률이 0.69 이하일때 항상 스케줄링 가능
  - 단점 : 주기가 긴 태스크들의 우선순위가 낮아서 장시간 대기

- **EDF**(Earliest Deadline First, 최단 마감시간 우선)알고리즘 : 프로세스의 마감시한이 가까울수록 우선순위를 높게 부여하는 선점방식의 동적 스케줄링
    - 장점 : 이론적으로 총 이용률이 1 이하면 스케줄링 가능
    - 단점 : 태스크들의 수행 시간, 마감시간, 주기 등을 정확히 예측하는 것이 현실적으로 어려움.

[정리]

비선점형 : FCFS, 비선점형 SJF

선점형 : RR, MLQ, MLFQ, 선점형 SJF(SRF), RM(Rate Monotonic), EDF

**22-3) 선점 스케줄링과 비선점 스케줄링의 차이점?**

선점 : CPU 를 할당받아 실행 중인 프로세스로부터 CPU 를 선점(빼앗는 것)하여 다른 프로세스를 할당 할 수 있는 방식

Ex) RR, SRT, MLQ, MFQ

비선점 : CPU 를 할당받은 프로세스는 스스로 CPU 를 반납할 때까지 CPU 를 독점하여 사용

Ex) FCFS, SJR, HRN

**23) 메모리 단편화 란 무엇인가?**

메모리의 빈 공간 또는 자료가 여러 개의 조각으로 나뉘는 현상을 말한다. 할당한 메모리를 해제를 하게 되면 그 메모리 공간이 빈 공간(사용하지 않는 공간)이 되고 그 빈공간의 크기보다 큰 메모리는 사용할 수 없다. 그리하여 이 공간들이 하나 둘 쌓이게 되면 수치상으로는 많은 메모리 공간이 남았음에도 불구하고, 실제로 사용할 수 없는 메모리가 발생한다.

**23-1) 내부단편화와 내부단편화란?**

- 내부단편화

: 분할된 영역이 할당된 프로그램의 크기보다 커서 사용되지 않고 남아 있는 빈 공간을 말한다.

: 내부 단편화는 페이징에서 발생한다.

- 외부단편화

: 분할된 영역이 할당될 프로그램의 크기보다 작아서 모두 빈 공간으로 남아 있는 전체 영역을 말한다.

: 외부 단편화는 세그먼테이션에서 발생한다.

**23-2) 메모리 단편화 해결방법은?**

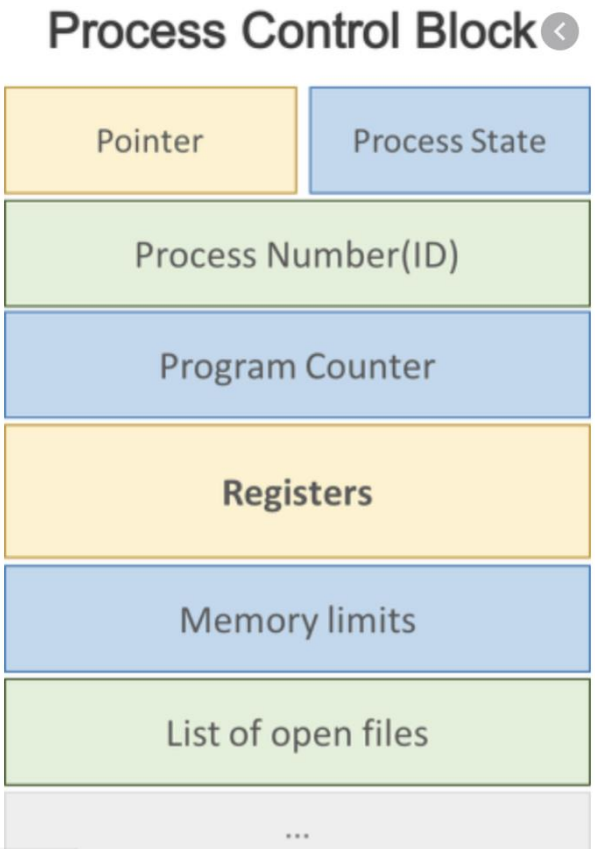
메모리 압축(디스크 조각 모음), 메모리 통합(단편화가 발생된 공간들을 하나로 통합시켜 큰 공간으로 만드는 기법)

**24) 문맥교환(Context Switching)이란?**

프로세스 상태를 변경하는 것을 말한다. 하나의 프로세스가 CPU 를 사용 중인 상태에서 다른 프로세스가 CPU 를 사용하도록 하기 위해 이전 프로세스의 상태를 보관하고 새로운 프로세스의 상태를 적재하는 작업이다. 즉, 스케줄링에 의해 실행 중인 코드, 자원 등을 저장하고 현재 상태를 대기 상태로 만들고, 다른 프로세스를 실행시키는 과정

［참고］ CPU 가 현재 처리중인 프로세스의 PCB(Process Controll Block)을 따로 저장하고 다른 PCB 를 가져오는 것

PCB? 특정 프로세스에 대한 중요한 정보를 저장하고 있는 운영체제의 자료구조이다. 운영체제는 프로세스를 관리하기 위해 프로세스의 생성과 동시에 고유한 PCB 를 생성한다.



PCB 의 구성

**25) 커널 수준 스레드와 사용자 수준 스레드의 각각 장단점?**

- 커널 수준 스레드

: 장점

- 프로세스의 스레드들을 몇몇 프로세서에 한꺼번에 디스패치 할 수 있기 때문에 멀티프로세서 환경에서 매우 빠르게 동작한다.
- 다른 스레드가 입출력 작업이 다 끝날 때까지 다른 스레드를 사용해 다른 작업을 진행할 수 있다.
- 커널이 각 스레드를 개별적으로 관리할 수 있다.
- 커널이 직접 스레드를 제공해 주기 때문에 안정성과 다양한 기능이 제공된다.

: 단점

- 스케줄링 동기화를 위해 커널을 호출하는데 무겁고 오래걸린다. (저장한 내용을 다시 불러오는 과정이 필요)
- 즉, 사용자 모드에서 커널 모드로의  전환이 빈번하게 이뤄져 성능 저하가 발생한다.
- 사용자가 프로그래밍할 때 구현하기 어렵고 자원을 더 많이 소비하는 경향이 있다.

- 사용자 수준 스레드

: 장점

- 운영체제에서 스레드를 지원할 필요가 없다
- 스케줄링 결정이나 동기화를 위해 커널을 호출하지 않았기 때문에 인터럽트가 발생할 때 커널 레벨 스레드보다 오버헤드가 적다
- 즉, 사용자 영역 스레드에서 행동을 하기에 OS Scheduler 의 context switch 가 없다. (사용자 레벨 스레드 스케줄러 이용)
- 커널은 스레드의 존재조차 모르기 때문에 모드 간의 전환이 없고 성능 이득이 발생한다.

: 단점

- 시스템 전반에 걸친 스케줄링 우선순위를 지원하지 않는다. (무슨 스레드가 먼저 동작할 지 모른다)
- 프로세스에 속한 스레드 중 I/O 작업 등에의해 하나라도 블록이 걸린다면 전체 스레드가 블록된다.

**25-1) 사용자 수준 스레드와 커널 수준 스레드 차이는?**

코드가 실행되는 모드의 차이.

커널 수준의 스레드는 커널 모드기 때문에 write()같은 함수를 사용할 수 있다. 사용자 모드에서 커널 스레드를 사용하면 문맥 교환이 일어나서 오버헤드가 발생할수 있다. 사용자 스레드가 여럿 있을때 하나라도 커널 모드가 되면 다른 쓰레드가 중지된다. 그래서 요즘에는 혼합해서 쓸수 있는 스레드를 사용하는 것으로 알고 있다.

**26) 모드 스위치와 프로세스 스위치 간의 차이점은?**

모드 스위치는 사용자 모드에서 커널모드로 변경할 때 발생하며 완전문맥교환이 필요하지 않고 시스템 스택을 사용한다. 프로세스 스위치는 보통 문맥교환이라고 부르는 것이며 실행중인 프로세스를 멈추고 새 프로세스를 실행하는 것

# [Database]

**1. 데이터베이스 종류에 대하여 말해 보세요.**
1) Hierarchical DataBase : IMS/DB, HDB
2) Network  DataBase
3) Relational DB, OR(Relational, Object) DB
  - 대형 Oracle, DB2, Sysbase, Informix
  - PostgresSQL
  - 기타 MongoDB
4) OODB(Object Oriented DataBase)

**※Oracle과 MySQL의 차이를 말해 보세요.**
- Oracle : 대용량처리에 적합, UNIX, Linux, 메인프레임 등에서 사용, DB관리자 별도
- MySQL : 5000만건 미만의 데이터 주로 PC, UNIX시스템에서 사용, 보통 개발자가 DB관리
- SQL의 80% - 90%가 비슷 하지만
  MySQL에 없는 Oraclr 명령어 : varchar2, nvl, nvl2, sequence, decode, outer join에서 + 등
  Oracle에 없는 MySQL 명령어 : autoincrement, show, ifnull 등

**2. 데이터베이스 언어(Database Language)에 대하여 말해 보세요.**
- DDL(Data Definition Language) : CREATE, ALTER, DROP
- DML(Data Manipulation Language) : SELECT, INSERT, DELETE, UPDATE
- DCL(Data Control Language) : COMMIT, ROLLBACK, GRANT, REVOKE

**3. 각종 제약사항(Constraint)에 대하여 말해 보세요.**
- Not Null
- Primary key
- Foreign Key
- Unique
- Default
- Check

**4. 검색성능을 향상 시키는 색인(index)에 대하여 말해 보세요.**
색인(index)은 컴퓨터에서 내용을 미리 목록으로 만들어 놓고 찾고자 하는 내용을 검색하는데 시간을 줄이기 위한 것이다.
색인은 검색성능을 향상시키기 위한 것으로 정보 요구자가 보다 빨리 정보에 접근할 수 있도록 그 정보의 소재를 표시해 주고, 원하는 자료의 유무를 확인시켜 주며 자료의 신속한 이용을 가능하게 하는 기능을 가지고 있다.
인덱스(색인)이 많으면 조회 속도는 빠르지만 입력/수정/삭제의 속도는 떨어진다

**5. 트랜잭션(transaction)에 대하여 말해 보세요.**
데이터베이스 트랜잭션(Database Transaction)은 데이터베이스 관리 시스템 또는 유사한 시스템에서 상호작용의 단위로 논리적 작업 단위(LUW, Logical Units of Work) 이다. 여기서 유사한 시스템이란 트랜잭션이 성공과 실패가 분명하고 상호 독립적이며, 일관되고 믿을 수 있는 시스템을 의미한다.
데이터베이스 시스템은 각각의 트랜잭션에 대해 원자성(Atomicity), 일관성(Consistency), 고립성(Isolation), 영구성(Durability)을 보증한다. 이 성질을 첫 글자를 따 ACID라 부른다.

Begin transaction
Commit transaction (트랜잭션이 성공적이며, 갱신이 실제 적용됨)

**6. 커밋(Commit)과 롤백(Roll back)에 대하여 설명해 보세요**

- Commit - 처리결과의 영구적 반영을 시행한다.
- Rollback - 결과를 취소, 트랜잭션의 처음 시점으로 되돌린다

7. CRUD 란 무엇인가?

| 이름 | 조작 | SQL |
|---|---|---|
| Create | 생성 | INSERT |
| Read(또는 Retrieve) | 읽기(또는 인출) | SELECT |
| Update | 갱신 | UPDATE |
| Delete(또는 Destory) | 삭제(또는 파괴) | DELETE |

8. 커서(Cursor)에 대하여 말해 보세요.

커서(Cursor)는 일련의 데이터에 순차적으로 액세스할 때 검색 및 "현재 위치"를 포함하는 데이터 요소이다.


---

Q # 1) DBMS 를 정의하십시오.

**답변** : DBMS 는 데이터베이스 관리 시스템을 나타냅니다. 사용자가 데이터에 관한 정보를 가능한 한 효율적이고 효과적으로 구성, 복원 및 검색 할 수 있도록하는 응용 프로그램 모음입니다.

널리 사용되는 DBMS 중 일부는 MySql, Oracle 등입니다.

Q # 1-1) RDBMS 를 정의하십시오.

**답변** : 관계형 데이터베이스 관리 시스템 (RDBMS)은 데이터베이스에 별도의 테이블에 저장된 관계형 데이터 모델을 기반으로하며 공통 열의 사용과 관련이 있습니다. SQL (Structured Query Language)을 사용하여 관계형 데이터베이스에서 데이터에 쉽게 액세스 할 수 있습니다.

Q # 1-2) DBMS 의 장점에 대해 설명하세요.

**답변** : DBMS 의 장점은 다음과 같습니다.

- 데이터는 구조적으로 저장되므로 중복성이 제어됩니다.
- 입력 한 데이터의 유효성을 검사하고 데이터베이스에 대한 무단 액세스에 대한 제한을 제공합니다.
- 필요한 경우 데이터 백업 및 복구를 제공합니다.
- 여러 사용자 인터페이스를 제공합니다.

**Q # 2) 데이터베이스에서 다양한 유형의 관계는 무엇입니까?**

**답변 :** 데이터베이스에는 3 가지 유형의 관계가 있습니다.

- **일대일 :** 한 테이블은 비슷한 종류의 열을 가진 다른 테이블과 관계가 있습니다. 각 기본 키는 관련 테이블에서 하나의 레코드 또는 하나의 레코드와 관련이 없습니다.
- **일대다 :** 한 테이블은 기본 및 외래 키 관계가있는 다른 테이블과 관계가 있습니다. 기본 키 테이블에는 관련 테이블에없는 하나 또는 여러개의 레코드와 관련된 하나의 레코드 만 포함됩니다.
- **다대다 :** 두 테이블의 각 레코드는 다른 테이블의 여러 레코드와 관련 될 수 있습니다.

**Q # 3) SQL 을 설명하세요.**

**답변 :** SQL 문은 기본적으로 DDL, DML 및 DCL 의 세 가지 범주로 나뉩니다.

다음과 같이 정의 할 수 있습니다.

**DDL (데이터 정의 언어)** 명령은 데이터를 보유하는 구조를 정의하는 데 사용됩니다. 이 명령은 자동 커밋됩니다. 즉, 데이터베이스의 DDL 명령에 의해 수행 된 변경 사항이 영구적으로 저장됩니다.

**DML (데이터 조작 언어)** 명령은 데이터베이스의 데이터를 조작하는 데 사용됩니다. 이 명령은 자동 커밋되지 않으며 롤백 할 수 있습니다.

**DCL (데이터 제어 언어)** 명령은 데이터베이스에서 데이터를 사용하기위한 액세스 권한 취소와 같이 데이터베이스에서 데이터의 가시성을 제어하는 데 사용됩니다.

**Q # 4) 정규화와 비정규화를 설명하십시오.**

**답변 :**

**정규화** 는 데이터 무결성을 유지하기 위해 잘 정의 된 방식으로 테이블을 분할하여 데이터베이스에서 중복 데이터를 제거하는 프로세스입니다. 즉, 관계형 데이터베이스에서 중복을 최소화하기 위해 데이터를 구조화하는 작업입니다. 이 프로세스는 많은 저장 공간을 절약합니다.

**비정규화** 는 복잡한 쿼리 속도를 높이고 성능을 향상시키기 위해 테이블에 중복 데이터를 추가하는 프로세스입니다.

**Q # 4-1) 정규화의 장점은?**

**답변:** 데이터베이스 변경 시 이상 현상을 제거하고, 데이터베이스 구조 확장 시 재디자인을 최소화합니다.

**Q # 4-2) 정규화의 단점은?**

**답변:** 릴레이션 분해로 인해 릴레이션 간의 연산(join)이 많아집니다. 이로 인해 응답 시간이 느려질 수 있습니다.

**Q # 5) 데이터베이스 뷰란?**

**답변 :** 허용된 데이터를 제한적으로 보여주기 위해 하나 이상의 테이블에서 유도된 가상 테이블입니다.

**Q # 5-1) 데이터베이스 뷰의 장점과 단점은 무엇입니까?**

**답변 :**

**뷰의 장점 )**

- 뷰의 데이터가 저장되는 물리적 위치가 없으므로 리소스를 낭비하지 않고 출력을 생성합니다.
- 삽입, 업데이트 및 삭제와 같은 명령을 허용하지 않으므로 데이터 액세스가 제한됩니다.

**뷰의 단점 )**

- 해당 뷰와 관련된 테이블을 삭제하면 뷰가 관련이 없습니다.
- 큰 테이블에 대해 뷰를 만들 때 더 많은 메모리가 사용됩니다.

**Q # 6) ER 모델은 무엇입니까?**

**답변 :** ER 모델은 데이터베이스의 개념적 뷰를 정의하는 엔터티-관계 모델입니다.

ER 모델은 기본적으로 실제 실체와 그 연관 / 관계를 보여줍니다. 여기서 엔티티는 데이터베이스의 속성 세트를 나타냅니다.

**Q # 6-1) 엔터티(Entity), 엔터티 타입(Entitiy type) 및 엔터티 집합(Entitiy set)을 정의하십시오.**

**답변 :** 엔터티는 실세계에서 독립적으로 존재하는 장소, 클래스 또는 객체 일 수 있습니다.

엔터티 타입은 유사한 특성을 가진 엔터티 집합을 나타냅니다.

데이터베이스에 설정된 엔티티는 특정 엔티티 유형을 갖는 엔티티의 집합을 나타냅니다.

**Q # 7) 데이터베이스 트랜잭션이란 무엇입니까?**

**답변 :** 데이터베이스의 일관성있는 상태를 다른 것으로 변경하는 작업 순서를 데이터베이스 트랜잭션이라고합니다. 트랜잭션 완료 후 성공적인 완료가 시스템에 반영되거나 트랜잭션이 실패하고 변경 사항이 반영되지 않습니다.

**Q # 7-1) 트랜잭션의 4 가지 성질에 대해 설명해보세요.**

**답변 :** 총 네가지 성질을 가지고있습니다.

Atomicity(**원자성**) 는 트랜잭션의 연산이 DB 에 모두 반영되던지 전혀 반영이되지 않던지 둘중에 하나만 수행해야한다.

Consistency(**일관성**) 는 트랜잭션이 성공적으로 완료된 후에는 언제나 일관성 있는 DB 상태로 변환되어야한다.

Isolation(**독립성**) 은 수행중인 트랜잭션이 완전히 완료되기 전에는 다른 트랙잭션에서 수행 결과를 참조할 수 없다.

Durablility(**지속성**) 는 성공적으로 완료된 트랜잭션의 결과는 시스템이 고장나더라도 영구적으로 반영되어야 한다.

**Q # 8) 인덱스 란?**

**답변 :** 인덱스는 데이터분야에 있어서 테이블에 대한 동작의 속도를 높여주는 자료 구조를 말합니다. 인덱스는 테이블 내의 1 개의 컬럼, 혹은 여러 개의 컬럼을 이용하여 생성될 수 있습니다. 고속의 검색 동작 뿐만 아니라 레코드 접근과 관련하여 효율적인 순서 매김 동작에 대한 기초를 제공합니다.

**인덱스를 사용해야 하는 경우**

- 데이터의 양이 많고 검색이 변경보다 빈번한 경우
- 인덱스를 걸고자 하는 필드의 값이 다양한 값을 가지는 경우

**인덱스를 사용할 시 단점**

- DB 의 10%정도 공간이 요구됩니다.

- 인덱스를 생성하는 시간이 크게 요구됩니다.
- INSERT, DELETE, UPDATE 쿼리문을 실행할 때 별도의 과정이 추가적으로 발생하기 때문에 DB 의 변경작업이 잦으면 성능이 저하됩니다.

인덱스 헌팅은 인덱스 수집을 향상시켜 데이터베이스 성능뿐만 아니라 쿼리 성능을 향상시키는 프로세스입니다.

**Q # 8-1) 인덱스 헌팅을 사용하여 쿼리 성능을 향상시키는 방법은 무엇입니까?**

**답변 : 인덱스 헌팅은 다음을 통해 쿼리 성능을 향상시킵니다.**

- 쿼리 최적화 프로그램을 사용하여 워크로드와 쿼리를 조정합니다.
- 인덱스 및 쿼리 배포의 성능 및 효과 관찰

**Q # 9) 검사점(Checkpoint)를 정의하십시오.**

**답변 :** Checkpoint 는 모든 로그가 저장 디스크에 영구적으로 저장되고 일관성이없는 지점을 선언합니다. 충돌이 발생하면 시스템이 검사 점에서 다시 시작할 수 있으므로 작업량과 시간이 절약됩니다.

**Q # 10) 데이터 사전(Data dictionary)을 설명하십시오.**

**답변 :** 데이터 사전은 테이블과 데이터베이스 개체의 내용과 구조를 설명하는 정보 집합입니다. 데이터 딕셔너리에 저장된 정보의 작업은 데이터베이스 요소 간의 관계를 제어, 조작 및 액세스하는 것입니다.

**Q # 11) 기본 키(Primary key)와 복합 키(Compound key)를 설명하십시오.**

**답변 : 기본 키** 는 모든 행 데이터가 고유하게 식별되는 테이블의 해당 열입니다. 테이블의 모든 행에는 기본 키가 있어야하며 두 행은 동일한 기본 키를 가질 수 없습니다. 기본 키 값은 절대로 null 이거나 수정하거나 업데이트 할 수 없습니다. **복합 키** 는 열 세트가 테이블의 모든 행을 고유하게 식별하는 후보 키의 양식입니다.

**Q # 12) Unique 키로 무엇을 이해하십니까?**

**답변 :** Unique 키는 유일성을 가지기 위해 설정해 놓은 키로 중복이 되는 것을 방지합니다. Primary 키는 오직 하나만 생성할 수 있지만, Unique 키는 여러개 생성이 가능합니다. Primary 키의 경우 NULL 값을 허용하지 않지만, Unique 키는 NULL 값을 허용합니다.

**Q # 13) 데이터베이스 트리거로 무엇을 이해하십니까?**

**답변 :** 테이블에서 삽입 전, 삽입 후, 업데이트시, 행 삭제시와 같은 이벤트가 발생할 때 자동으로 실행되는 명령 세트를 데이터베이스 트리거라고합니다.

**Q # 14) 저장 프로 시저를 정의하십시오.**

**답변 :** 저장 프로시저는 사전 컴파일 된 SQL 쿼리의 모음으로, 사전에 준비해 둔 많은 명령을 자동으로 실행할 수 있기 때문에 작업의 효율성도 높일 수 있습니다.

**Q # 15) 'DELETE', 'TRUNCATE'및 'DROP'명령을 구분하십시오.**

**답변 :** 'DELETE' 연산을 실행 한 후 손실 된 데이터를 검색하기 위해 COMMIT 및 ROLLBACK 문을 수행 할 수 있습니다.

'TRUNCATE' 조작을 실행 한 후 손실 된 데이터를 검색하기 위해 COMMIT 및 ROLLBACK 문을 수행 할 수 없습니다.

'DROP' 명령은 기본 키 / 외래 키와 같은 테이블 또는 키를 삭제하는 데 사용됩니다.

**Q # 16) 이상현상이란?**

**답변 :** 릴레이션에서 일부 속성들의 종속으로 인해 데이터의 중복이 발생하는 것을 말합니다.

**Q # 16-1) 이상의 종류에 대해 말하세요.**

**답변 :**

**삽입 이상** 은  원하지 않는 자료가 삽입된다든지, 삽입하는데 자료가 부족해 삽입이 되지 않아 발생하는 문제점을 말한다.

**삭제 이상** 은 하나의 자료만 삭제하고 싶지만, 그 자료가 포함된 튜플 전체가 삭제됨으로 원하지 않는 정보 손실이 발생하는 문제점을 말한다.

**갱신 이상** 은 정확하지 않거나 일부의 튜플만 갱신되어 정보가 모호해지거나 일관성이 없어져 정확한 정보 파악이 되지 않는 문제점을 말한다.

**Q # 17) 트리거에 대해 설명하고, 트리거를 쓰는 이유에 대해 말하세요.**

**답변 :** 자동으로 실행되도록 정의된 저장 프로시저입니다. INSERT/UPDATE/DELETE 문에 대한 응답으로 자동 호출합니다.

**트리거를 사용하는 이유**

- 업무 규칙을 보장
- 업무 처리 자동화
- 데이터 무결성 강화(변경, 생성, 제거, 복구 시)

**Q # 18) 데이터베이스 무결성이란?**

**답변 :** 데이터 베이스에 저장된 데이터 값과 그것이 표현하는 현실 세계의 실제값이 일치하는 정확성을 말합니다.

**개체 무결성** 은 릴레이션에서 기본키를 구성하는 속성은 NULL 값이나 중복값을 가질 수 없다.

**참조 무결성** 은 외래키 값은 NULL 이거나 참조 테이블의 기본키 값이어야함

**Q # 19) 조인에 대해 설명하고 조인의 종류에 대해 말하세요.**

**답변 :** 두 개 이상의 테이블이나 데이터베이스를 연결하여 데이터를 검색하는 방법입니다.

Inner Join 은 2 개 이상의 테이블에서 교집합만을 추출

Left Join 은 2 개 이상의 테이블에서 from 에 해당하는 부분을 추출

Right Join 은 2 개 이상의 테이블에서 from 과 join 하는 테이블에 해당하는 부분을 추출

Outer Join 은 아웃터 조인 또는 풀 조인이라고 말함, 2 개 이상의 테이블에서 모든 테이블에 해당하는 부분을 추출

**Q # 20) 교착상태란?**

**답변 :** 2 개 이상의 트랜잭션이 특정 자원(테이블 또는 행)의 잠금(Lock)을 획득한 채 다른 트랜잭션이 소유하고 있는 잠금을 요구하면 아무리 기다려도 상황이 바뀌지 않는 상태가 되는데 이를 **교착상태** 라고 합니다.

**Q # 20-1) 교착상태를 방지하기 위한 방법에 대해 설명하세요.**

**답변 :**

- 트랜잭션을 자주 커밋한다.
- 정해진 순서로 테이블에 접근한다.
- SELECT ~ FOR UPDATE 의 사용을 피한다.

**Q # 21) NoSQL 이 기존 RDBMS 와 다른 점은?**

**답변 : NoSQL** 은 스키마가 없습니다. 즉 데이터 관계와 정해진 규격(table-column 의 정의)이 없습니다.

관계 정의가 없으니 Join 이 불가능하고 (하지만 reference 와 같은 기능으로 비슷하게 구현은 가능) 트랜잭션을 지원하지 않습니다.

분산처리(수평적 확장)의 기능을 쉽게 제공한다는 장점이 있습니다.

대부분의 NoSQL DB 는 분산처리기능을 목적으로 나왔기 때문에 분산처리 기능을 자체 프레임워크에 포함하고 있다.

**Q # 21-1) 어떤상황에서 NoSQL 을 쓰는 것이 더 적합한가?**

**답변 :** 비정형 데이터를 저장해야할 때 가장 적합합니다.

**Q # 22) 테이블을 드롭(DROP)하는 것과 자르는 것(Truncate), 그리고 테이블 내 모든 레코드를 삭제(Delete)하는 것의 차이점은 무엇입니까?**

**답변 :** DELETE TABLE 은 로그되는 작업이기 때문에 삭제되는 각 행은 트랜잭션 로그에 기록되고 이것은 작업을 느리게 합니다. TRUCATE TABLE 역시 테이블 내 행들을 삭제하지만 삭제되는 각 행을 기록하지 않고 대신 테이블의 데이터베이스 할당 해제를 기록하여 작업이 빠릅니다. TRUNCATE TABLE 는 롤백할 수 없습니다.

DELETE 명령어는 데이터는 지워지지만 테이블 용량은 줄어들지 않는다. 원하는 데이터만 지울 수 있다. 삭제 후 RollBack 가능하다.

TRUNCATE 명령어는 용량이 줄어들고, 인덱스 등도 모두 삭제된다. 테이블은 삭제하지는 않고 데이터만 삭제한다. 한꺼번에 다 지워야 한다. 삭제 후 절대 되돌릴 수 없다.

DROP 명령어는 테이블 전체를 삭제,공간, 객체를 삭제한다. 삭제 후 절대 되돌릴 수 없다.

# [Servlet & JSP]

**1. Server Side Applet인 Servlet에 대하여 정의하고 Servlet클래스를 만들기 위한 방법을 기술하시오.**
JAVA Servlet은 JAVA를 사용하여 웹페이지를 동적으로 생성하는 서버측 프로그램 혹은 그 사양을 말하며, 흔히 "서블릿"이라 불린다.
서블릿은 JAVA EE 사양의 일부분으로, 주로 이 기능을 이용하여 쇼핑몰이나 온라인 뱅킹 등의 다양한 웹 시스템이 구현되고 있다.
서블릿은 외부 요청마다 프로세스보다 가벼운 스레드로써 응답하므로 보다 가볍다. 또한, 서블릿은 JAVA로 구현되므로 다양한 플랫폼에서 동작한다.

**2. Servlet과 JSP의 차이점에 대하여 말해 보세요.**
- Servlet은 JAVA 소스에 HTML코드가 삽입된다.
- JSP는 반대로 HTML코드에 JAVA코드가 삽입된다.
- Servlet Class는 컴파일과정과 등록 과정이 필요하지 하지만 JSP는 필요 없다.
- Servlet 보다는 JSP 디자인과 로직에 대한 구분이 명확해서 유지보수가 용이하다.
- 간단한 로직을 구현할 때는 JSP가 더 간편하다.
- 하지만 복잡한 로직을 구현할 때에 HTML 중심의 코드가 이해하기 어렵게 만들 수 있고, 프로그래밍 언어를 모르는 사람이 실수로 중요한 코드를 건들 우려도 있다. 그리고 힘들게 개발한 로직의 유출을 막기 위해서도 Servlet 기술이 필요하게 된다.
  그래서 요즘은 JSP 기술과 Servlet 기술을 혼용한 MVC 프로그래밍 방법이 권장되고 있다. 프로그램의 기능을 구현하는 복잡한 로직은 서블릿 클래스 안에 기술하고, 그 결과를 가져다가 출력하는 일만 JSP 페이지가 담당하도록 만드는 방법이다.

**3. JSP구성요소인 지시어(Directive), 주석문(Comments), 선언문(Declarations), 연산문(Expressions), 수행문(Scriptlets) 을 설명하세요.**
* 지시어(Directive) --> <%@ %>
 1)page : JSP 페이지에 대한 정보를 지정. (문서의 타입, 출력 버퍼 크기, 에러 페이지 등.)
  - language, contentType, import, session, buffer, autoFlush, errorPage, isErrorPage 등등
 2)tablib : JSP 페이지에서 사용 할 태그 라이브러리 지정
 3)include : JSP 페이지의 특정 영역에 다른 문서를 포함

* 주석문(Comment)
 1)HTML <!-- --> : 소스보기로 코드를 볼 수 있다
 2)JAVA //, /* ~ */ : 소스보기로 코드를 볼 수 없다
 3)JSP <%-- --%>
* 선언문(Declarations)
  <%! %> : JAVA 메서드를 만든다
* 연산문(Expressions)
  <%= %> : 값을 출력한다
* 수행문(Scriptlets)
  <% %> : 자바 코드를 실행한다.
- Action Tag : <jsp:~ />
- Custom Tag
- EL (Expression Language)
- Implicit Object -- 내장객체

**3-1. request와 response 객체에 대해 말해보시오.**
request : 클라이언트에서 넘어 오는 데이터를 전달 받기 위한 객체
response : 서버에서 클라이언트로 데이터를 전달하기 위한 객체

**3-2. 클라이언트에서 데이터를 넘기는 방법에 대해 설명하시오**
1) html에서 form 객체를 작성하여 action 으로 정해진 서버의 url로 전달하는데 전달할 때는 submit 함수를 이용해 전달하게 된다.
2) get 방식으로 url 뒤에 ?를 이용하여 작성하여 준다.

**4. name, age를 담는 javaBean을 작성해 보세요.**

```
public class Person {
    private String id;
    private String name;
    public String getId() {
        return id;
    }
    public void setId(String id) {
        this.id = id;
    }
    public String getName() {
        return name;
    }
    public void setName(String name) {
        this. name = name;
    }
}
```

**5. <jsp:setProperty/><jsp:getProperty/>그리고<jsp:useBean/>을 언제 사용하는 것입니까?**

화면에서 입력 받은 데이터를 받아서 저장할 객체를 생성하고(<jsp:useBean/>)
생성된 객체의 멤버변수에 데이터를 저장(<jsp:setProperty/>)
저장된 데이터를 사용하기 위하여 가져오는 경우(<jsp:getProperty/>) 사용

**6. Servlet의 LifeCycle에 해당하는 메소드와 그 메소드에서 처리될 내용에 대해 설명하시오.**

* 서블릿은 init(), service(), destroy() 메소드를 호출하는 Life Cycle을 가짐.
* init() 메소드
  - 서블릿이 메모리에 Load 되면 init() 메소드 수행.
  - 서블릿이 서비스하기 위해 필요한 초기화 작업 수행.
  - 한 번만 수행. (병행적으로 수행되지 않음.)
  - 서블릿이 실행하기 위해서 필요한 각종 환경을 설정하기 위한 목적으로 사용.
    (필요한 파일 열기, 데이터베이스에 연결 등)
  - 맨 처음 클라이언트의 요청에 의해서 메모리에 로드되며, 메모리에 로드된 후에는 메모리에 계속 남아 클라이언트의 요청을 처리.
* service() 메소드
  - 클라이언트의 요청이 있을 때마다 Thread가 생성되어서 병행적으로 service() 메소드 수행.
  - 병행성 문제 고려.
  - HTTP의 method 타입에 따라 GET 방식이면 doGet()메소드를, POST 방식이면 doPost()메소드 호출.
  * destroy() 메소드
  - 메모리가 Unload되기 전에 destroy() 메소드 수행.
  - 한 번만 수행. (병행적으로 수행되지 않음.)
  - 종료 시에 필요한 끝내기에 관련된 작업을 처리.
* 서블릿 작업 중단
  - 서블릿 작업 중단을 위해서는 doPost(), doGet(), service()등의 함수에서 return문을 사용.
  - System.exit()를 호출하면 서블릿 컨테이너가 종료.

**8. HTTP수행방식 중 GET방식과 POST방식의 차이점을 기술하시오.**

- GET은 URL에 값이 ?뒤에 쌍으로 이어 붙고 POST는 숨겨진 값으로 전송된다.
- GET은 URL에 이어 붙기 때문에 길이제한이 있어서 많은 양의 데이터는 보내기 어렵고 POST는 많은 양의 데이터를 보내기에도 적합하다.
- GET은 가져오는 것(Select)이고 POST는 수행(Insert, Update, Delete)하는 것입니다.

**9. 쿠키(Cookie)와 세션(Session)을 정의하고 특징을 설명하시오.**

* 쿠키(cookie)
  - 클라이언트(브라우저)에 데이터를 저장, setCookie
* 세션(Session)
  - SID(session ID)를 식별자로 서버에 데이터를 저장
  - SID로는 쿠키나 도메인 파라미터를 사용, 주로 사용자 인증시에 사용함

**10. 쿠키(Cookie)와 세션(Session)의 장단점에 대하여 말해 보세요.**

- 쿠키(Cookie) : 클라이언트에서 실행됨으로 보안이 취약하나 속도가 빠르다.

- 세션(Session) : 서버에서 실행되며 속도는 느리지만 보안 기능이 강력하다. 로그인 인증은 보안 때문에 세션(Session)으로 만든다.

**11. "Hello Servlet"을 출력하는 ServletCode를 작성하시오..**

```
public class HelloServlet extends HttpServlet {
    protected void doGet(HttpServletRequest request,
                    HttpServletResponse response) throws ServletException, IOException {
        response.setContentType("text/html;charset=euc-kr");
        PrintWriter out = response.getWriter();
        out.println("<html><head><title>Hello JSP</title></head>");
        out.println("<body>Hell Servlet</body><html>");
        out.close();
    }
}
```

**12. XML의 특징을 설명하시오.**

XML(Extensible Markup Language)은 W3C에서 다른 특수 목적의 마크업 언어를 만드는 용도에서 권장되는 다목적 마크업 언어이다. XML은 SGML의 단순화된 부분집합이지만, 수많은 종류의 데이터를 기술하는 데 적용할 수 있다. XML은 주로 다른 시스템, 특히 인터넷에 연결된 시스템끼리 데이터를 쉽게 주고 받을 수 있게 하여 HTML의 한계를 극복할 목적으로 만들어졌다.
XML은 마크업 언어의 일종으로, 문서를 사람과 기계 모두가 읽을 수 있는 형식으로 부호화하는 규칙의 집합을 정의한다.
XML의 설계 목표는 단순성, 일반성, 인터넷을 통한 사용가능성을 강조하였다. XML은 텍스트 데이터 형식으로 유니코드를 통해 전 세계 언어를 지원한다. XML 설계가 문서에 집중하지만, 임의의 자료구조를 나타내는 데 널리 쓰이고 있다. 예를 들어 웹 서비스가 그렇다.

**13. 웹 프로그래밍에서 한글 깨짐 현상이 발생하는 2가지 경우와 이를 방지하기 위해 사용할 수 있는 방법을 기술하시오.**

★. 한글 지원 : 국내는 euc-kr과 UTF-8을 사용하고 있다
euc-kr : 한글과 영어 지원
UTF-8 : 다국어 지원

★. 따라서 아래에서 사용된 euc-kr과 UTF-8은 대체해도 답이 될 수 있다. 물론 같은 프로젝트 내에서는 한 가지로 통일해야 한다

1. response(응답)
   1) response.setContentType(“text/html;charset=”UTF-8”);
2. request(요청)
   1) 공통 : new String(ko.getBytes("euc-kr"),"8859_1");
   2) post방식 request.setCharacterEncoding("UTF-8");
   3) get방식(Tomcat의 경우)
      . 톰캣홈 ₩conf 폴더와 이클립스의 [프로젝트 탐색기]뷰에서 [Servers]-[Tomcat v5.5 서버]항목 에 있는 server.xml 파일 <Connector>태그의 속성에 URIEncoding="EUC-KR"문장 추가
      . Server.xml Connector태그에 useBodyEncodingForURI=“true”추가

**14. MVC 란 무엇인가?**

모델-뷰-컨트롤러(Model-View-Controller, MVC)는 소프트웨어 공학에서 사용되는 아키텍처 패턴이다. 이 패턴을 성공적으로 사용하면, 사용자 인터페이스로부터 비즈니스 로직을 분리하여 애플리케이션의 시각적 요소나 그 이면에서 실행되는 비즈니스 로직을 서로 영향 없이 쉽게 고칠 수 있는 애플리케이션을 만들 수 있다. MVC 에서 모델은 애플리케이션의 정보(데이터)를 나타내며, 뷰는 텍스트, 체크박스 항목 등과 같은 사용자 인터페이스 요소를 나타내고, 컨트롤러는 데이터와 비즈니스 로직 사이의 상호동작을 관리한다.

**15. 아이바티스란 무엇인가? (요즘은 마이바티스로 이름을 바꿈)**

iBatis(아이바티스)는 SQL에 기반한 데이터베이스와 자바, 닷넷(.NET), 루비(Ruby) 등을 연결시켜 주는 역할을 하는 영속성 프레임워크이다. 이러한 연결은 프로그램의 소스코드에서 SQL 문장을 분리하여 별도의 XML 파일로 저장하고 이 둘을 서로 연결시켜주는 방식으로 작동한다.

# [웹표준 기술]

### 1. jQuery 란 무엇인가?
jQuery(제이쿼리)는 브라우저 호환성이 있는 HTML 속 자바스크립트 라이브러리이며 클라이언트 사이드 스크립트 언어를 단순화 할 수 있도록 설계되었다. 존 레식이 2006 년 뉴욕 시 바캠프(Barcamp NYC)에서 공식적으로 소개하였다. jQuery 는 오늘날 가장 인기 있는 자바스크립트 라이브러리 중 하나이다.

### 2. DOM 은 무엇인가?
문서 객체 모델(DOM; Document Object Model)은 객체 지향 모델로써 구조화된 문서를 표현하는 형식이다. DOM 은 플랫폼/언어 중립적으로 구조화된 문서를 표현하는 W3C 의 공식 표준이다.

### 3. Ajax 란 무엇인가?
Ajax(Asynchronous JavaScript and XML, 에이잭스)는 대화식 웹 애플리케이션의 제작을 위해 아래와 같은 조합을 이용하는 웹 개발 기법이다.
- 표현 정보를 위한 HTML (또는 XHTML) 과 CSS
- 동적인 화면 출력 및 표시 정보와의 상호작용을 위한 DOM, 자바스크립트
- 웹 서버와 비동기적으로 데이터를 교환하고 조작하기 위한 XML, XSLT, XMLHttpRequest (Ajax 애플리케이션은 XML/XSLT 대신 미리 정의된 HTML 이나 일반 텍스트, JSON, JSON-RPC 를 이용할 수 있다).

### 4. Node.js 란 무엇인가?
Node.js 는 확장성 있는 네트워크 어플리케이션(특히 Server-side) 개발에 사용되는 소프트웨어 플랫폼이다. Node.js 는 작성언어로 자바스크립트를 활용하며 Non-blocking I/O 와 단일 스레드 이벤트 루프를 통한 높은 처리성능을 가지고 있다.
Node.js 는 내장 HTTP 서버 라이브러리를 포함하고 있어 웹서버에서 아파치 등의 별도의 소프트웨어 없이 동작하는 것이 가능하며 이를 통해 웹서버의 동작에 있어 더 많은 통제를 가능케 한다.

### 5. CSS에 대하여 설명해 보세요.
- Cascading StyleSheet 의 약자로 태그 속성들을 하나의 스타일로 설정하는 용도로도 쓰인다
- HTML 문서 내에 글자의 글꼴 종류, 크기, 색, 여백 등을 지정
- 글자의 정렬 방식을 결정하거나 글자에 그림자를 지정하는 등 다양한 효과
- 사용자의 웹 브라우저 환경에 상관없이 일정한 화면을 보여줄 수 있다

### 6. CSS는 개발자가 다루어야 할 부분인지? 웹 디자이너가 다뤄야 할 부분인지?

# [Spring Framework Programming]

### 1. CI 란 무엇인가?
continuous integration 의 약자로 말 그대로 지속적인 통합을 말한다.
개발이 완료되는 시점에 단위기능들을 통합하는 것이 아닌 개발과 동시에 통합을 진행함으로써 소프트웨어의 품질을 향상시키는 것을 목표로 한다.

### 2. DI 란 무엇인가?
Dependency Injection 의 약자로 의존성을 주입하는 것을 뜻한다.
설정 파일을 통해 객체간의 의존관계를 설정함으로써 외부 Assembler 가 객체간의 의존 관계를 정의하게 되며, 객체는 직접 의존하고 있는 객체를 생성하거나 검색할 필요가 없어지므로 코드의 관리가 쉬워진다.

### 3. AOP 란 무엇인가?
Aspect Oriented Programming 을 뜻하며 다양한 곳에서 자주 사용되는 공통 관심요소를 단일 기능으로 뽑아내어 코드의 중복을 줄이고 관리의 효율성을 높이는 것을 목적으로 한다.
로깅이나 로그인 등의 기능을 예로 들 수 있다.

### 4. POJO 란 무엇인가?
Plain Old Java Object, 간단히 POJO는 말 그대로 해석을 하면 오래된 방식의 간단한 자바 오브젝트라는 말로서 J2EE등의 중량 프레임워크들을 사용하게 되면서 해당 프레임워크에 종속된 "무거운" 객체를 만들게 된 것에 반발해서 사용되게 된 용어이다. POJO라는 용어는 이후에 주로 특정 자바 모델이나 기능, 프레임워크 등을 따르지 않은 자바 오브젝트를 지칭하는 말로 사용되었다. 스프링 프레임워크는 POJO 방식의 프레임워크이다.

### 5. Maven(메이븐)에 대하여 말해 보세요.
- JAVA용 프로젝트 관리 툴
- 아파치(Apache) 오픈 소스 빌드 툴
- 빌드
- 문서화
- 리포팅
- 의존 관계
- 소스 코드 관리
- 릴리즈
- 배포
- 프로젝트 관리에 필요한 모든 작업을 추상화하고 표준화해서 반복을 제거
- 메이븐이 접근할 수 있는 저장소를 지원
- 이 저장소를 통해 템플릿 프로젝트인 아키타입, 의존 관계에 있는 라이브러리, 메이븐 플러그인 기능을 지원
- 프로젝트 빌드에 필요한 라이브러리, 플러그인을 저장소에서 개발자 PC로 자동으로 다운로드

### 6. SVN(Subversion)에 대하여 말해 보세요.
- SVN은 버전관리 시스템으로 서버기반으로 사용할 수 있는 프로그램이다.
- 로컬과 SVN서버와 커밋 넘버링 방식으로 SVN버전을 체크하여서 동기화를 하는 방식이다.

### 7. Hibernate에 대하여 말해 보세요.
Hibernate은 ORM(Object-Relational Mapping) 프레임워크의 한 종류이다.
많이 사용하는 iBatis도 같은 부류의 프레임워크이다.

### 8. O/R Mapping에 대하여 말해 보세요.
ORM 이란 객체와의 관계를 맵핑시킨다는 뜻인데, 객체와 DB 정보의 관계를 맵핑시켜 좀더 효율적으로 데이터를 처리하고자 하는 프레임워크이다.

### 9. iBATIS역할에 대하여 말해 보세요.
iBatis(아이바티스)는 SQL에 기반한 데이터베이스와 JAVA, 닷넷(.NET), 루비(Ruby) 등을 연결시켜 주는 역할을 하는 영속성 프레임워크이다. 이러한 연결은 프로그램의 소스코드에서 SQL 문장을 분리하여 별도의 XML 파일로 저장하고 이 둘을 서로 연결시켜주는 방식으로 작동한다.
또 다른 영속성 프레임워크인 하이버네이트(Hibernate)와 비교하여 하이버네이트는 객체모델을 사용자가 생성을 하면 프레임워크에서 데이터베이스와 연결을 시켜주는 방식인데 반해 iBatis는 사용자가 SQL 문장을 만들면 그에 적합한 객체모델을 생성하는 방식으로 작동한다.

**10. REST 서비스란 무엇인가?**

확장성 생성 언어(XML) 파일로 된 웹 페이지를 읽어 원하는 정보를 수집하는 기능. 웹 페이지를 만드는 사람은 주기적으로 내용을 개정하고 사용자는 그 페이지의 URL만 알면 웹 브라우저로 읽어 정보를 얻을 수 있다. 하이퍼텍스트 전송 규약(HTTP)과 XML을 포함한 웹 기술 및 프로토콜을 사용하는 구조적 형태로서 단순 객체 접근 프로토콜(SOAP)보다 사용이 간편하고, 사이트 내용을 기술하는 RSS(RDF Site Summary)의 정보 편집 기능과 유사하다. RSS는 자원 기술 개념(RDF)을 사용한다.

**11. iBatis와 MyBatis ORM 프레임워크에서 하나 이상의 레코드를 검색해서 컬렉션 리스트로 반환하는 쿼리문 실행 메소드에 대해 말해 보세요.**

- iBatis : queryForList()
- MyBatis : selectList()

**12. iBatis의 inset() 메소드와 MyBatis insert() 메소드의 차이점에 대해 말해 보세요.**

- iBatis의 inset() 메소드는 반환값이 Object 형태 입니다.
- MyBatis insert(0 메소드는 실행된 레코드 행의 수를 int 정수형으로 반환합니다.

**13. iBatis의 inset() 메소드를 사용하면 반환값이 Object형이므로 저장 성공과 실패를 if문으로 판별하기가 어렵다. 그러면 iBatis에서 레코드를 저장해서 반환값을 쿼리문으로 실행된 레코드의 갯수로 반환해서 if문으로 분기 하려면 어떻게 해야 하는지 말해 보세요.**

- update() 메소드를 사용하면 됩니다.

**14. MyBatis를 이용할 때 쿼리의 종류에는 어떤 것이 있는지 말해 보세요.**

- 한건 조회 : selectOne
- 여러건 조회 : selectList

# [네트워크]

## 1. 네트워크란?

: 물리적 전송 매체를 사용하여 서로 연결된 장치 세트

- 컴퓨터 네트워크는 하드웨어, 데이터 및 소프트웨어와 같은 정보 및 리소스를 통신하고 공유하기 위해 서로 연결된 컴퓨터 그룹
- 네트워크에서 노드는 둘 이상의 네트워크를 연결하는 데 사용됨

### 1-1. 네트워크 망의 종류

- LAN : 한 건물 또는 사무실 내의 호스트들 간에 연결된 소규모 네트워크

- WAN : LAN 과 LAN 을 연결하는 대규모 네트워크

## 2. Cast 의 종류

- Unicast : 1:1 통신, 원하는 대상 하나를 정해서 통신
- Multicast : 1:N 통신, 원하는 대상 여러 명을 정해서 통신
- Broadcast : 1:all 통신, 내 의지와 상관없이 무조건 받아들여야 하는 통신

## 3. 회선, 대역폭이란?

: 전송되는 데이터를 허용할 수 있는 동시접속자 수

## 4. ISP 란?

: Internet Service Provider, 인터넷 서비스 공급자로 다양한 회선 상품을 제공하며 기업마다 서비 스가 다름

**5. VPN 이란?**

: Virtual Private Network, 가설사설망으로 ISP 에 정보를 넘겨주지 않고 익명성을 유지하여 인터넷 에 접속

**6. DSL 이란?**

: Digital Subscriber Line, 전화선을 이용한 인터넷 서비스, 현재는 잘 쓰이지 않음
- ADSL : 비대칭, 다운로드는 빠르고 업로드는 느림, 전화선에 모뎀과 마이크로 필터를 사용해서 인 터넷에 연결, 전화국으로부터 거리가 멀어도 OK
- VDSL : 초고속, 대칭/비대칭 모두를 지워하며 ADSL 처럼 전화선을 이용하나 속도가 훨씬 빠름. 전 화국으로부터 거리가 멀면 안 됨

**7. FTTH 란?**

: Fiber To The Home, 광통신, 초고속 기가 인터넷, 집안까지 광케이블을 통해 인터넷을 제공하는 서비스

**8. IP 란?**

: Internet Protocol Address, 컴퓨터 네트워크에서 기기들이 서로를 인식하고 통신하기 위해 사용 하는 식별번호

**9. 패킷이란?**

: 네트워크 상에서 전송하는 데이터를 일정한 크기로 자른, 작게 나눠진 데이터의 묶음 - 누구에게 어디로 무엇을 보내야 하는지에 대한 정보가 담겨 있음, 안정성 때문에 나눠서 보냄

**10. TCP/IP 프로토콜 4 계층**

- LINK 계층 : 물리적인 계층, LAN, WAN, MAN 과 같은 네트워크 표준과 관련된 프로토콜을 정의 하는 영역
- IP 계층 : 데이터 경로 설정, 특정한 규칙 없음, 오류 발생하면 다른 임의의 경로로 변경
- TCP/UDP(전송) 계층 : 데이터의 실제 송수신, IP 계층에서 발생한 문제를 해결

- APPLICATION 계층 : 서버와 클라이언트를 만드는 과정에서 프로그램의 성격에 따라 정한 데이터 송수신에 대한 약속(규칙)

**10-1. 네트워크 애플리케이션의 역할**

- TCP/IP 소프트웨어에 데이터를 전달할 때, 데이터를 받을 호스트의 주소인 IP 주소와 포트번호도 함께 전달
- IP 주소 : 네트워크에 연결된 기기를 식별하는 유일한 번호
- 포트 번호 : 수신 측에서 동작하는 여러 애플리케이션 중 데이터를 수신할 애플리케이션을 식별하 는 번호

**11. TCP 와 UCP 의 특징과 차이점은?**

**TCP**
: 연결지향형 전송규약
- 흐름 중심 프로토콜, 통신을 주고받는 것을 중요시함

- 중간에 패킷이 손실되는 경우 재전송을 통해(SYN-ACK handshaking) 신뢰성을 보장함(느림) - 대부분의 통신에서 사용됨, 특히 파일이나 데이터 전송 시에 사용

- 데이터 경계 구분이 없음 (바이트 스트림 서비스)

**UDP**

: 비연결지향형 전송규약
- 데이터 중심 프로토콜, 주고받는 통신보다 데이터를 일방적으로 보내는 것을 중요시함 - 데이터 전송의 신뢰성 보장 X, (빠름)
- P2P, 스트리밍, 전화에 사용

| TCP | UDP |
|---|---|
| 연결지향형<br>세그먼트<br>순서 보장, 느림<br>HTTP, 메일, 파일<br>헤더(20 바이트) 추가하여 IP 로<br>: 포트번호, 순서번호, 인정 번호, 제어 비트 | 비연결지향형<br>데이터그램<br>순서 보장 X, 빠름<br>DNS, Broadcasting 헤더(8 바이트) 추가하여 전송<br>: 포트번호, 데이터의 길이, 체크섬 |

**12. 3-Handshaking 과 4-Handshaking 의 과정은?**

**12-1. 3-Handshaking**

: TCP 에 쓰이는 연결 설정
- SYN/SYC : 통신 요청 데이터

- ACK : 응답 데이터
- SYN_RCV : 통신 요청 받음

### 12-2. 4-Handshaking

| | Client 상태 | 전송 데이터 | Server 상태 |
|---|---|---|---|
| 1 | CLOSE | # 연결 X | LISTEN |
| 2 | CLOSE | --SYN-> | LISTEN |
| 3 | CLOSE | | **SYN_RCV** |
| 4 | CLOSE | <-ACK+SYN-- | SYN_RCV |
| 5 | **ESTABLISHED** | --ACK-> | SYN_RCV |
| 6 | ESTABLISHED | # 연결 성공 | **ESTABLISHED** |

| | Client 상태 | 전송 데이터 | Server 상태 | 전송 데이터 | 애플리케이션 상태 |
|---|---|---|---|---|---|
| 1 | ESTABLISHED | # 연결 중 | ESTABLISHED | | # 프로세스 진행 |
| 2 | **FIN_WAIT_1** | --FIN-> | ESTABLISHED | | - |
| 3 | FIN_WAIT_1 | <-ACK-- | **CLOSE_WAIT** | | - |
| 4 | **FIN_WAIT_2** | | CLOSE_WAIT | --CLOSE()-> | # 프로세스 종료 |

| | | | | | |
|---|---|---|---|---|---|
| 5 | FIN_WAIT_2 | <-FIN-- | **LAST_ACK** | | |
| 6 | **TIME_WAIT** | --ACK-> | LAST_ACK | | |
| 7 | CLOSED | # 연결 X | **CLOSED** | | |

**12-3. 비정상 종료**

- CLOSE_WAIT 상태 : 애플리케이션에서 close()를 처리해주지 못하면, TCP 포트는 CLOSE_WAIT 상태로 계속 기다리게 된다. CLOSE_WAIT 상태가 statement 에 많아지게 되면, Hang 이 걸려 더는 연결을 하지 못하는 경우가 발생. 여러 상황에 따라 close() 처리를 잘 해 줘야 함
- FIN_WAIT_1 상태 : 상대방 측에 연결 종료를 요청했는데 ACK 를 받지 못한 상태로 기다리는 것. 네트워크 및 방화벽의 문제일 수 있음 TIME OUT 이 되면 자동으로 닫음

- FIN_WAIT_2 상태 : 클라이언트가 서버에 종료를 요청한 후 서버에서 ACK 를 받았지만, FIN 패킷 을 받지 못하고 기다리고 있는 상태. 서버 측에서 CLOSE 를 처리하지 못하는 경우. TIME OUT 이 되 면 스스로 CLOSED 함

***. 3-H / 네 번째 줄에서, 클라이언트가 서버가 보낸 ACK+SYC 를 받지 못하면?**

A. 두 번째 줄에서 클라이언트는 서버로 SYC 를 보내고 시간을 잰다. Timeout 이 되기 전까지 ACK+SYC 가 오지 않으면, 다시 SYC 를 보내고 ACK+SYC 수신을 대기한다.

*** 4-H / 서버가 마지막에 FIN 을 보내는 이유?**

서버가 아직 클라이언트에 보낼 데이터가 남아있을 경우 데이터를 다 전송하지도 못한 채 클라이언트 에서 포트를 닫아버리게 되므로 서버 또한 종료될 준비가 되었다는 의미로 FIN 을 보냄

*** 4-H / 클라이언트가 마지막에 ACK 를 굳이 보내는 이유?**

서버가 보낸 FIN 을 클라이언트가 받지 못하면 클라이언트는 FIN_WAIT_2 상태로 종료가 되지 못한 채 계속 기다려야 한다. 하지만 서버는 이미 포트를 닫고 더는 응답을 하지 않는 상태이기에 클라이 언트는 불필요한 자원을 소모할 수 있음

***HTTPS 환경에서의 3-H**

- Client -> Server : SSL 정보 및 암호화방식, 무작위 바이트 문자열(A)
- Server -> Client : 인증서, 무작위 바이트 문자열(B)
- Client 가 CA 에 인증서 목록에 있는지 확인 후 있다면 공개키 받음
- Client -> Server : 무작위 바이트 문자열 A 와 B 를 조합, 공개키로 암호화하여 전송
- Server 에서 비밀키로 받은 무작위 바이트 문자열 조합을 복호화, 이것으로 session key 를 만듦 - 해당 session key 를 가지고 암호화한 데이터를 주고받음

**13. OSI 7 Layer 란?**

: ISO(국제표준화기구)에서 네트워크 통신 과정을 7 단계로 정의한 국제통신표준규약

1) 물리 : 전송하는데 필요한 기능 제공 (통신 케이블, 허브)
2) 데이터링크 : 송/수신 확인, MAC Address 로 통신 (브릿지, 스위치)
3) 네트워크 : IP 를 기반으로 데이터(패킷) 전송 경로 결정 (라우팅)

4) 전송 : TCP/UDP 포트 정보를 참조해 데이터의 전송
5) 세션 : 통신 시스템 사용자 간의 연결을 유지 및 설정
6) 표현 : 세션 계층 간의 주고받는 인터페이스를 일관성 있게 제공

7) 응용 : 사용자가 네트워크에 접근할 수 있도록 서비스를 제공

**14. 허브와 리피터를 비교해주세요**

| | 리피터<br>허브 | 허브 |
|---|---|---|
| 공통점 | -물리계층에서 전기적인 신호를 증폭시켜 전송 거리를 연장하는 장치<br>-네트워크 신호가 연결된 모든 PC 에 전달되기 때문에 연결된 장치가 많을수록 부하가 심해짐 | |
| 차이점 | | 패킷 모니터링과 멀티 포트를 지원하여 문제가 생긴 곳을 고립시킬 수 있음 |

**15. 브릿지와 스위치를 비교하세요**

| | 브릿지 | 스위치 |
|---|---|---|
| 공통점 | -데이터링크계층에서 전송 거리를 연장하는 장치 | |
| 차이점 | 소프트웨어적으로 프레임을 다시 만들어 전송해 더 느림 | 성능에 따라 L2, L3, L4, L7 로 구분됨<br>하드웨어적으로 처리해 더 빠름 |

**16. ARP (Address Resolution Protocol)와 RARP 를 비교하세요**

| | ARP | RARP |
|---|---|---|
| 공통점 | -네트워크 계층에서 사용되는 주소 결정 프로토콜 | |
| 차이점 | IP 주소에서 MAC 주소를 알아냄 | MAC 주소에서 IP 주소를 알아냄 |

- 상대방 MAC 주소를 모를 때, IP 와 브로드 캐스팅 네트워크 주소 FFFFFFFFFFFF 를 가지는 ARP 패킷을 네트워크에 전송하여 이를 수신한 호스트가 자신의 MAC 주소를 반송하는 메커니즘
- 이때 ARP 캐시라 불리는 메모리에 테이블 형태로 저장하여, 패킷을 전송할 때에 다시 사용됨

**17. 게이트웨이란?**

: 외부로 연결되는 통로, 로컬망 라우터와 외부망 라우터 간의 통로를 말함

**18. 로드 밸런싱이란?**

: 분산식 웹 서비스로 여러 서버에 부하(Load)를 나누어 줌, Round Robin, Least Connection, Response Time, Hash 등의 기법이 있음
1) Round Robin : 각 서버에 session 을 순서대로 연결하는 방식, 모든 클라이언트를 똑같이 취급 하고, 서버별 처리량을 기억하고 있어야 함

2) Least Connextion : 클라이언트와 서버별 연결된 connection 수를 고려하여 가장 적은 서버에 연결하는 방식

**19.** 주요 포트 넘버

| 프로토콜 | 포트 번호 | 내용 |
|---|---|---|
| HTTP | 80 | 웹을 지원하기 위한 프로토콜. GET, PUT 같은 프로토콜 기능을 포함해서 웹 서버에게 어떠한 Content 를 요청하고 또는 웹 서버로 정보를 보냄 |
| FTP | 20, 21 | TCP 를 활용해 대량의 파일을 송신하고 수신하는 프로토콜 |
| TFTP | 69 | UDP 를 사용하는 파일 전송 프로토콜, 라우터나 스위치 등의 네트워크 장 비의 IOS 이미지를 업로드, 다운로드할 때 사용 |
| Telenet | 23 | 원격지에 있는 장비로 표준 터미널 에뮬레이션 기능을 제공함. 네트워크 장비에서는 텔넷을 통해 원격지에서 장비를 설정 |
| SMTP | 25 | 컴퓨터 네트워크를 통해 전자 메일을 전송하는 프로토콜. 받을 때는 POP3 를 활용 |
| SNMP | 161 | 네트워크 장비를 모니터링하고 제어하기 위해 사용하는 프로토콜로 네트워 크 장애 관리, 장비 설정, 통계 성능 및 보안 등을 관리 |
| DNS | 53 | 도메인 주소를 IP ADDRESS 로 변경, 모든 퍼블릭 IP 주소와 호스트 이름 은 DNS 에 저장되고 나중에 해당 IP 주소로 변환 |

**20.** 프로토콜이란?

: 컴퓨터 간 데이터 통신을 원활히 하기 위해 규정한 약속, 신호 송신의 순서(handshaking)나 데이 터표현법, 오류 검출법 등을 정한 것

### 20-1. HTTP 프로토콜이란?

: 하이퍼텍스트를 전송하는 규약
- 하이퍼텍스트 : 한 문서에서 다른 문서로 즉시 접근할 수 있는 텍스트
- 비연결성 프로토콜, REQUEST 에 대한 RESPONSE 만 전달되고 연결 유지 X

### 20-2. 비연결성을 해결하기 위한 방법

- Cookie/Session : Cookie 에 클라이언트에 대한 정보를 저장해뒀다가 사용하거나 Session 을 등록 해서 유지하는 방식
- Session Storage/Local Storage : HTML5 에서 제공, 세션 스토리지는 세션이 유지되고 있을 때 까지 브라우저 내부 저장소에 저장하고 세션이 끊기면 자동으로 없어짐, 로컬 스토리지는 사용자나 프론트엔드 내부적으로 삭제를 하지 않는 이상 영구적으로 저장됨

### 20-3. HTTPS 프로토콜이란?

: HTTP + SSL, HTTP 로 통신하는 소켓을 SSL(Secure Socket Layer) or TLS(Transport Layer Security)라는 프로토콜로 대체한 것 (새로운 별개의 프로토콜이 아니라 연결 방식이 달라진 것)
- HTTP 는 TCP 와 직접 통신하지만, HTTPS 에서는 SSL 과 통신하고 SSL 이 TCP 와 통신하는 방식
- SSL 을 사용하기 때문에 암호화와 증명서, 안전성 보호를 이용할 수 있음

- 공통키 암호화 방식과 공개키 암호화 방식을 혼합한 하이브리드 암호 시스템 사용, 공통키를 공개 키 암호화 방식으로 교환하고 이후 통신은 공통키 암호를 사용하는 방식

### 20-4. HTTP REQUEST - GET 과 POST 의 차이점

- GET : 서버에 데이터를 전달할 때 URL Query 를 사용해야 하므로 보안에 취약함 / 데이터를 받 는 용도로 적합
- POST : 데이터를 Header 에 넣어서 전송하므로 헤더를 열어보지 않으면 확인할 수 없음 / DB 내 용을 갱신하거나 서버로 데이터를 전송할 때 적합

- SSL 을 이용한 HTTPS 프로토콜로 데이터 전송을 암호화하면 보안성을 보완할 수 있음, URL 뒤에 붙는 쿼리스트링 내용 모두 암호화되어 전송되기 때문에 보안성을 강화함

### 20-5. Restful API 에서의 URL 과 일반적인 HTTP 에서의 URL 의 차이는?

- 일반적인 HTTP URL : 기능에 중점을 두어 설계, 예) 회원 정보 호출 - '/getUser'
- Restful API : 자원에 중점을 두고 설계, 예) '/user' 하위에 기능에 대한 구분을 추가, POST, GET, DELETE, PUT 등의 HTTP 메서드를 사용

## 21. 자바스크립트에서 HTTP request 를 동기로 호출하고 값을 처리하면 발생하는 문제점은?

- Request 에 대한 Response 응답시간이 길어질 수도 있으므로 절차 지향적으로 짜놓은 코드가 제 대로 동작하지 않을 수 있음

### 21-1. 해결방안?

- CallBack 함수를 만들어 호출하면 해당 REQUEST 에 대한 응답이 온 후에 이후에 그 값을 가지고 다시 다른 함수를 실행함

### 21-2. 콜백함수의 문제점은? 그리고 해결방안?

- 콜백함수를 이용한 비동기처리를 많이 하면 '콜백헬'이라 불리는 가독성이 매우 떨어지는 코드가

됨. 가독성 저하는 유지보수에 걸림돌이 되므로, Promise 나 Async/Await 을 사용함. Promise 에서는 요청 후 비동기 처리하는 부분은 then 절에 추가하면 되고 Async/Await 는 Await 이 then 절의 역할.

### 21-3. Promise vs Async/Await?

- Promise: Async/Await 가 Promise 로 구성되어 있으므로, Promise 를 잘 모르면 Async/Await 도 잘 쓸 수 없음, Promise 로 다양한 비동기 처리를 경험해보고 다양한 오류를 처리해 봐야 함
- Async : try & catch 를 이용해서 예외처리를 할 수 있고, 간단한 비동기 처리 경우에는 가독성이 뛰어남

## 22. 프록시 서버 기능이란?

- 클라이언트가 프록시 서버를 통해 다른 네트워크 서비스에 간접적으로 접근을 할 수 있게 하는 것 - 프록시 서버는 요청된 내용을 캐시에 저장하고 다음에 같은 요청이 들어오면 캐시에 저장된 정보 를 제공해 전송시간을 단축함

### 22-1. 페이지의 내용과 데이터의 값이 계속해서 바뀌면?

- 캐시 만료기한을 설정함
- 프록시 서버라도 최초로 받는 요청에는 실제 서버로 요청을 보내야 하므로 그때 만료기한을 설정 해서 프록시 서버로 보내면 됨
- 프록시서 버로 사용자가 요청했을 때 요청한 시각이 프록시에서 다운로드 받은 시간에서 만료기한 이내이면 프록시에서 다운로드를 할 것이고, 그렇지 않다면 다시 실제 서버로 요청을 하게 됨

**23. AOT 와 JIT 에 대해 설명해주세요.**

- JIT : Just In Time Compile, 브라우저에서 템프릿 컴파일을 진행하기 때문에 느림, JIT 컴파일러 를 포함해야 하므로 용량도 큼
- AOT : Ahead Of Time Compile, 빌드 시 템플릿을 먼저 컴파일을 함, 빌드에는 시간이 더 소요 되지만 브라우저에서는 컴파일이 실행되지 않기 때문에 상대적으로 빠름

- 개발 시에는 JIT 방식으로 빠르게 빌드해서 변경사항을 확인하고, 실제 서비스 배포 시에는 AOT 방식으로 빌드해서 전체 용량 감소 및 컴파일 시간을 없앰

**24. Big Endian 과 Little Endian 이란?**

- 엔디안 : 컴퓨터 메모리에 연속된 바이트를 배열하는 방법

| Big Endian | Little Endian |
|---|---|
| -최상위 바이트가 앞에 오는 경우<br>-사람이 읽고 쓰는 방법과 같아서 디버깅이 쉬움 -수가 커지면 메모리에 저장된 데이터를 오른쪽으로 옮겨야 함 | -최하위 바이트가 앞에 오는 경우<br>-디버깅이 어려움<br>-수가 커지더라도 |

| | |
|---|---|
| | 오버헤드가 발생하지 않음 |

**25. 가상 DOM 이란?**

- 가상돔 : 추상화한 돔
- 가상돔을 사용하지 않고 div 태그 1000 개에 CSS 효과가 추가된다면? 천개의 돔 노드들을 일일이

검색하고 업데이트 해야 함
- 탐색비용과 업데이트 비용을 좀 더 줄이기 위해, 추상화한 돔에서 탐색과 업데이트를 한 후 변경사 항만 실제 돔에 반영
- 어떻게 돔을 추상화할 것인지, 언제 돔에 변경사항을 적용할지에 대한 알고리즘이 핵심

**26. 방화벽이란?**

- 방화벽 : 컴퓨터 네트워크를 무단 액세스로부터 보호하는 데 사용되는 네트워크 보안 시스템
- 외부로부터의 악의적인 액세스를 방지, 외부 사용자에게 제한된 액세스 권한을 부여하기 위해 방화 벽을 구축함
- 하드웨어 장치, 소프트웨어 프로그램, 또는 이 둘의 조합으로 구성됨
- 방화벽을 통해 라우팅되는 모든 메시지는 특정 보안 기준에 따라 검사되며 기준을 충족하는 메시 지는 네트워크를 통해 성공적으로 통과하거나 해당 메시지가 차단됨
- 다른 컴퓨터 소프트웨어와 마찬가지로 설치할 수 있으며, 나중에 필요에 따라 사용자 정의하고 액 세스 및 보안 기능을 일부 제어할 수 있음

**26-1. Windows 방화벽은?**

: 운영 체제와 함께 제공되는 Microsoft Windows 응용 프로그램, 바이러스, 웜 등을 방지하는 데 도움을 줌

**27. 브라우저 주소창에 http://www.test.com 입력 후 엔터를 눌렀을 때부터 페이지가 렌더링 되는 과정을 설명하세요.**

1) local DNS → 루트 DNS 서버 → .com DNS 서버 → test.com DNS 서버 순서대로 www.test.com 에 해당하는 IP 주소 요청하고, 있다면 그 서버에서 바로 주소를 받음

2) TCP 통신을 통해 소켓 개방
3) HTTP 프로토콜로 요청
4) 라우팅 중 프록시 서버를 만나면 웹 캐시에 저장된 정보를 response 받음
5) 프록시 서버를 만나지 못해 www.test.com 를 서빙하는 서버까지 가면 요청에 맞는 데이터를 response 로 전송함
6) 브라우저의 loader 가 해당 response 를 다운로드 할지 말지 결정
7) 브라우저의 웹 엔진이 다운로드한 .html 파일을 파싱해 DOM 트리를 결정
8) .html 파싱중 script 태그를 만나면 파싱을 중단함
9) script 태그에 있는 자원을 다운로드해 처리가 완료되면 다시 파싱 함
10) CSS parser 가 .css 파일을 파싱해 스타일 규칙을 DOM 트리에 추가하고 렌더 트리를 만듦
11) 렌더트리를 기반으로 브라우저의 크기에 따라 각 노드들의 크기를 결정
12) 렌더링 엔진이 배치를 시작(페인팅)

# [Project]

**1. SI가 무엇인지 말해 보세요.**
SI[system integration] = 시스템통합
기업의 경영목표 달성을 위해 정보시스템을 구축하는 종합서비스를 말한다. 하드웨어, 소프트웨어, 통신망, 전산인력 등의 전산자원을 일의 목적과 특성에 맞게 통합해 최적의 해결점을 제시하고 정보시스템을 개발·유지·보수하는 과정까지 포함하는 광범위한 개념이다.
기업이 필요로 하는 정보시스템에 관한 기획에서부터 개발과 구축, 나아가서는 운영까지의 모든 서비스를 제공하는 일.

**2. 개발 프로세서에 대하여 말해 보세요.**
- ISP(information strategic planning) : 전략정보시스템 계획)
- BPR(business process re-engineering) : 업무 재설계
- 요구사항분석
- 업무분석
- 설계
- 구현(coding) ; 단위시험 포함
- 시험 : 통합시험, 시스템 시험, 인수 시험
- 인수 및 구축(운영서버에 시스템 구축 및 데이터 이행)
- 운영 및 유지 보수

**3. 프로젝트에 대하여 설명하고 본인의 역할은 무엇이었는지? 불화는 없었는지?**
목적, 기대효과와 주요 메뉴를 설명하고 자신의 업무 설명
의견이 대립 될 때 슬기롭게 타협한 과정 설명

**4. 이 작품에서 남들이 쉽게 따라 할 수 없는 자신만의 노하우(기술)는?**
만족할 만한 부분소개

**5. 이 작품에서 좀더 보강되어야 할 부분이 있는지?**
시간이나 기술력 부족으로 완성하지 못했으나 추가하고 싶었던 부분

**6. Spring을 사용한 이유에 대해 말해 보세요.**
Spring은 xml이나 annotation등을 이용한 선언적 관리가 가능하다.
개발의 복잡성이 감소했고, 특정환경에 제약을 받지 않으며, 테스트와 관리 및 유지 보수가 좋아지기 때문에

**7. MVC2 모델을 사용한 이유에 대해 말해 보세요.**
설계는 다소 시간이 필요하지만 프로그램 개발자와 디자이너가 전달되는 데이터만 선언되어 있다면
서로 상관없이 각자 따로 동시에 개발이 가능하다.

**8. Oracle(MySQL 등)을 사용한 이유에 대해 말해 보세요.**
Oracle
- JAVA 개발 시 호환성과 안정성을 중점으로 두고 있으며 그기에 적합한 Oracle을 주로 많이 사용하므로

MySQL
- 저렴한 가격으로 중소형 Project에서 많이 사용하고 다루기 쉬운 옵션을 제공한다.

MS-SQL
- windows 서버를 사용하는 경우 적합한 데이터베이스로 일반관리 사용자가 익숙한 windows와 유사하므로 데이터베이스관리자가 선호할 수 있다.

**9. 가장 힘들었던 경험과 그것을 어떻게 극복했는지?**

**10. JSP 외에 다른 프로그래밍 언어를 공부해 본 경험이 있는지?**

**11. 최근에 관심을 가지고 공부하고 있는 분야는?**

# [빅데이터, 인공지능]

**1. 어떤 데이터를 다루어 보셨나요?**
- 텍스트 데이터
- 이미지 데이터
- 숫자 데이터
- 음성 데이터

**2. 데이터를 수집, 정제, 변환 해 본 경험이 있나요?**
- 크롤링(Crawling), Flume, Spark

**3. 빅데이터 분석 및 시각화를 해 본 경험이 있나요?**
- R / R Studio
- Python
- Python 라이브러리 matplotlib과 seaborn

**4. 딥러닝 프레임워크는 어떤 것을 사용해 보셨나요?**
- TensorFlow
- Keras
- PyTorch(Torch)
- Caffe
- Deep Learning 4j
- Theano
- MxNet
- CNTK
- Lasagne
- Big DL

**5. 인공지능 개발도구는 어떤 것을 사용해 보셨나요?**
- 파이썬, R, JAVA, C/C++, Javascript, Lua, Julia, Swift

**6. TensorFlow 2.0을 사용해 보셨나요? 어떤 부분이 이전과 달라졌나요?**
- Keras와 즉시 실행(eager execution)을 이용한 쉬운 모델 작성
- 어떤 플랫폼에서든 튼튼한(robust) 모델 배포
- 연구를 위한 강력한 실험
- deprecated된 API를 정리하고 중복을 줄임으로써 API 단순화

**7. 다루어 본 인공지능 알고리즘에는 어떤 것들이 있나요?**

**머신러닝 알고리즘**
- 지도 학습(Supervised learning)
- 준지도 학습(Semi-supervised learning)
- 비지도(자율) 학습(Unsupervised learning)
- 강화 학습(Reinforcement learning)

**딥러닝 알고리즘**
- 심층 신경망(DNN, Deep Neural Network)
- 합성곱 신경망(CNN, Convoultional Neural Network)
- 순환 신경망(RNN, Recurrent Neural Network)
- 제한 볼츠만 머신(RBM, Restricted Boltzmann Machine)
- 심층 신뢰 신경망(DBN, Deep Belief Network)