!!!!!편규리 파이팅!!!!!

오버로딩과 오버라이딩

연산자는 오버로딩. 부등호나 등호는 오버라이딩

오버로딩은 같은 이름의 메소드를 여러 개 가지면서 매개변수의 유형과 개수가 다르도록 하는 것. parameter는 다를 수 있지만 결국 내용은 같음.

오버라이딩은 상위 클래스가 가지고 있는 메소드를 하위 클래스가 재정의 해서 사용한다. 상속 관계에 있는 클래스 간에 같은 이름의 메소드를 정의하는 기술을 오버라이딩이라고 한다.

abstract class와 interface

inerface와 abstract class는 모두 선언만 있고 구현 내용이 없는 클래스이다. interface는 그 안에 적혀 있는 method를 무조건 다 구현해야 하고, abstract는 일부만 구현해도 된다.

추상 클래스는 메소드에 대한 definition만 되어있고 정의가 안 되어 있음. 그걸 클래스에서 강제로 메소드의 바디를 만들게 되어 있다. 그래서 클래스가 각자의 flow를 가지게 한다. 어떤 클래스를 생성할 때 컴파일 타임에 생성할 수 도 있지만, 런타임에 결정되어야 할 때가 있다. 그럴 때 추상클래스를 쓰게 된다.

사람에 대한 클래스를 만들면 교직원/학생 클래스가 그걸 받아서 씀. 사람은 추상클래스니까 메소드가 선언만 되어 있고 교직원/학생이 각자의 flow를 가지고 구현이 되어 있음. 근데 런타임 때 교직원 1명 생성해줘! 이럴 수 있잖아! 그러면 클래스가 런타임에서 결정이 되어야 할 때 임.

ACID(원자성, 일관성, 고립성, 지속성)

데이터베이스 트랜잭션이 안전하게 수행된다는 것을 보장하기 위한 성질을 가리키는 약어이다.데이터베이스에서 데이터에 대한 하나의 논리적 실행단계를 트랜잭션이라고 한다. 예를 들어, 은행에서의 계좌이체를 트랜잭션이라고 할 수 있는데, 계좌이체 자체의 구현은 내부적으로 여러 단계로 이루어질 수 있지만 전체적으로는 '송신자 계좌의 금액 감소', '수신자 계좌의 금액 증가'가 한 동작으로 이루어져야 하는 것을 의미한다.

- **원자성**(Atomicity)은 트랜잭션과 관련된 작업들이 부분적으로 실행되다가 중단되지 않는 것을 보장하는 능력이다. 예를 들어, 자금 이체는 성공할 수도 실패할 수도 있지만 보내는 쪽에서 돈을 빼오는 작업만 성공하고 받는 쪽에 돈을 넣는 작업을 실패해서는 안된다. 원자성은 이와 같이 중간 단계까지 실행되고 실패하는 일이 없도록 하는 것이다.
- **일관성**(Consistency)은 트랜잭션이 실행을 성공적으로 완료하면 언제나 일관성 있는 데이터베이스 상태로 유지하는 것을 의미한다. 무결성 제약이 모든 계좌는 잔고가 있어야 한다면 이를 위반하는 트랜잭션은 중단된다.
- **고립성**(Isolation)은 트랜잭션을 수행 시 다른 트랜잭션의 연산 작업이 끼어들지 못하도록 보장하는 것을 의미한다. 이것은 트랜잭션 밖에 있는 어떤 연산도 중간 단계의 데이터를 볼 수 없음을 의미한다. 은행 관리자는 이체 작업을 하는 도중에 쿼리를 실행하더라도 특정 계좌간 이체하는 양쪽을 볼 수 없다. 공식적으로 고립성은 트랜잭션 실행내역은 연속적이어야 함을 의미한다. 성능관련 이유로 인해 이 특성은 가장 유연성 있는 제약 조건이다. 자세한 내용은 관련 문서를 참조해야 한다.
- **지속성**(Durability)은 성공적으로 수행된 트랜잭션은 영원히 반영되어야 함을 의미한다. 시스템 문제, DB 일관성 체크 등을 하더라도 유지되어야 함을 의미한다. 전형적으로 모든 트랜잭션은 로그로 남고

시스템 장애 발생 전 상태로 되돌릴 수 있다. 트랜잭션은 로그에 모든 것이 저장된 후에만 commit 상태로 간주될 수 있다.

객체지향의 개념

원래 절차적 프로그래밍이 있었다. 그러나, 이 방식은 간단한 알고리즘이면 모를까 조금만 복잡해지면 순서도로 나타내는 것이 불가능하다. 그래서 뭐 객체를 가지고 프로그래밍

class == 추상(abstraction) == type

표현 대상의 abstraction(== 특징에 대한 서술)을 말한다.

object == 실체(instance) == variable

object는 클래스의 instance이다. 즉, 클래스가 실체로 만들어진 것을 뜻한다. '고양이'라는 클래스가 있다면, '소금', '홍조' 들은 object 즉 구체, 실제 존재, 고유성을 갖고 있다. int a; 에서 int는 class, a는 object로 나타낼 수 있다.

다음 세가지 특성을 지원하는 언어를 객체지향 프로그래밍 언어라고 한다.

- encapsulation (캡슐화)

관련있는 것을 묶어서 생각하는 것. 묶어서 이름을 부여하는 것 -> abstraction

- inheritance (상속성)

하나의 클래스를 가지고 있는 특징들을 그대로 다른 클래스가 물려 받는 것

- polymorphism (다형성)

상속성의 계층을 따라서 각각의 클래스에 한 가지 이름을 줄 수 있는 것

map, set, list

내부구조는 트리로 되어 있고, key값으로 value를 찾을 수 있다.

맵에서 key나 integer가 있는데 내가 만든 object가 있을 때 연산자 오버로딩을 꼭 해주어야 한다. 자바는 hash code라는 게 있는데 그걸 오버라이딩 해줘야 한다.

set은 객체 중복을 허용하지 않는 집합이고, list는 인덱싱, 중복을 허용한다. 자바에서는 list가 필요에 의해 자동으로 늘어난다. C에서는 vector가 그렇겠지.

Map은 key와 data를 같이 저장한다. 이 때 key는 data의 성격을 설명해주는 값이다.

** 함수 call을 계속하면 지역변수와 매개변수가 저장되는 스택프레임이 계속 생긴다. 해당 스택 영역을 넘어가도 데이터가 저장되면, 해당 프로그램은 오동작을 하게 되거나 보안상의 크나큰 취약점을 가지게 된다. C에서는 스택 오버 플로우 발생하면 바로 강제 종료시킴

array와 linked-list

linked-list는 입력 (insert)일 때는 무조건 헤드에 연결하기 때문에 O(1)만에 insert입력을 진행할 수 있다. 그래서 insert가 많을 때는 linked-list쓰고 find많을 때는 array를 쓰는 것이 좋다.

UML

소프트웨어 공학에서 사용하는 표준화된 범용 모델링 언어.

HTTP 메소드 종류와 특징

브라우저에서 인터넷 주소 입력하면 일어나는 과정 (디테일하게)

cache, proxy에서 같은 url과 request가 있는지, fresh한지 확인하고 있다면 display한다.

없다면, DNS를 lookup해서 domain에 상응하는 IP를 찾는다.

찾은 걸로 TCP three hand shaking을 해서 TCP connection을 한다.

검색 request가 들어오면 parsing해서 검색 api를 호출하여 content를 받아온다.

content로 페이지를 채워서 response를 보낸다.

browser는 http 응답을 받는다.

(요기 밑에는 http1 기준)

- 애플리케이션에서 HTML 태그 데이터를 내려준다

- 브라우저는 태그를 파싱해서, 또 요청을 날려야하는 리소스들이 뭔지 파악한다

- CSS / 자바스크립트 / 이미지 등등을 요청해서 받아온다.

- 브라우저에 그려준다.

웹 트래픽 규모를 낮추는 방법

WAS에서 부하가 많이 나는 경우 대처하는 방법

로드밸런스 랜덤으로 하면 일어나는 문제점

멀티스레드를 사용하지 않고 프로그램을 짠다면?

스레드를 쓰는 것의 장단점

process는 프로그램 실행시 code, data, stack, heap의 구조로 되어있는 독립된 메모리 영역을 할당 받는다.

모든 process는 자신만의 address 공간을 가지고, thread는 자신만의 address 공간을 갖지 않는다. 따라

서 Thread는 메모리와 IO 장치들을 서로 공유한다. 생성하는 데 시간은 훨씬 적게 걸린다. thread는 빠른 프로세스 생성, 적은 메모리 사용, 쉬운 정보 공유 라는 장점이 있지만, 자원을 공유하기 때문에 Dead lock 상태에 빠질 수 있다는 단점이 있다.

디비 정규화에도 단점이 있다면?

join이 느려질 수 있다. 비정규화를 적절히 해주면 나아진다.

사용해 본 디자인 패턴 이름과 설명

HTTP2 특징

로드밸런싱 L2,L4,L7 특징

L2: mac주소를 기반으로 load balancing

L3: ip주소를 기반으로 load balancing

L4: Transport layer에서 load balancing

L7: application layer에서 load balancing

HTTP Status 코드들 범위별 내용

라인 동계 인턴 면접

대부분 자기소개서 기반으로 물음. 3명 면접관. 전부 해당 직무관련(핀테크) 개발자분들.

2분은 자바, 1분은 파이썬.

45분 면접- 15분 질문하기로 진행한다고 했지만 50분 면접-5분 질문으로 마무리 했음. 손코딩은 없었고 분위기만 좋았다….

경험 위주의 자기소개 요청받음. 30초정도로 준비한 것들 말함.

재미있었던 수업 뭔지 물어서 db, os라고 대답. -〉 계속 이거 2개 관련 질문만 함…

db disk-based, buffer, join, logging. 한 프로젝트 간단하게 설명함.

각 구현에서 어떤 어려움, 아쉬운것들 있었는지 물어서 답함.

logging에서 wal policy, transaction등에 대해서 설명함 -〉 이후 transaction 질문 개많았음.

transaction 개념 설명. ACID 설명 -〉 고립도 level로 새끼질문 -〉 고립도를 어떻게 설정해주어야 하느냐고 물음 -〉 장단점 등등에 대해서 또 물음.

객체지향 수업들은 것으로 넘어가서 -〉 객체지향의 개념, 설계 원리, 디자인 패턴 등에 대해서 물음 -〉 오버라이딩, 오버로딩 차이점.

자료구조 질문도 함. array와 linked-list에서 입력이 많이 일어날때 어떤 걸 써야하느냐 라고 함 -〉 입력이 insert를 말하는 거냐고 구체화 한뒤 답.

map, set, list 설명시킴 -〉 map에서 key로 object를 넣을때 어떤 추가적인 것들을 해줘야하는냐라고 물음 -〉 =, 〉 같은 것들을 오버라이딩 해줘야한다고 함 -〉 java에서는 hashcode라는 것을 오버라이딩 해야한다고 답을 말씀해주심 -〉 파이썬 사용하시는 면접관이 파이썬에서는 부등호 하면 된다고 해서 좀 무마됨,…

스택오버플로우 익셉션 일으키는 방법 질문 -〉 gets로 input 개많이 받아와서 eip 조작하고 그런게 있을 수 있다고 했는데 -〉 recursion인 경우에 대해서도 말해보라고 하심 -〉 stack frame 쌓아서 터진다고 함 -〉 알고보니깐 recursion같은거 계속해서 stack frame계속 쌓아서 터지는거를 답으로 원하신듯 -〉 보안동아리 해서 스택오버플로우에 대한 개념이 너무 치우쳐져있었다고 대답함…

소켓 구조체, 모나드, 프로퍼게이션에 대해서 들어봤냐 설명하라고 했는데 3개다 몰라서 모른다고 함.

db sharding질문 들어와서 설명함 -〉 naver d2 tech blog 봤다고 자백함… -〉 partitioning도 질문하심.

요즘 관심있는거 설명하래서 컴퓨터비전에서 배운 거 설명함.

오게 되면 어떤 직무 하고 싶냐고 해서 어플 만드는건 싫다고 답함 백앤드 좋아여…하고 했음.