1. 개발 환경 구성 확인(3)

- 내가 어떤환경 속에서

2. url 매핑(4)

- url 주소를 어떤식으로 할수있는지

- get 인지 post 방식인지

3. pom.xml(5)

- 어떤 설정을 할수있는지

- maven은 라이브러리 관리, 컴파일, 빌드를 동시에 수행하면서 테스트를 병행하고 서버 측 배포 자원을 관리할 수 있는 빌드 도구

- pom.xml 은 위의 메이븐의 기능 중 라이브러리 관리를 해주는 역할을 한다

4. 스프링의 주요 개념(10)

- 4문제

- 스프링 안에 존재하는 개념들

- 주요 개념들 (서술형) - 쓸데없이 ~이고 ~다 하며 두개적어서 틀리지말라- 하나라도 틀리면 틀린것

5. 스프링에 대한 개요 및 특징(3)

- 스프링은 무엇인가?

6. Junit 사용법(5)

- 오지선다형

7. Controller 접근 방법(5)

- 컨트롤러에 어떻게 접근할것인가?

8. 자바스크립트 유효성 검사(5)

- 어떻게 validation 체크를 할것인가?

- 자바스크립트 문법을 적어야한다 (오타나면 안됨)

9. jUnit 어노테이션의 의미(4)

- jUnit 도 프레임 워크라서 어노테이션이 있다

- 그 어노테이션의 의미를 묻는 것

10. 오류 원인 유추하기(서술형)(5)

- 어디서 오류가 나는지 적고 확인하는 것

- 어려움

- 형변환, 주소 ... 등이 틀렸다

11. 블랙 박스 테스트? 화이트 박스 테스트?(3)

- 개발자 테스트, 차후 수업예정

12. 단위 테스트의 종류 특징(4)

- 어려움, 차후 수업예정

**-> Mock 단위 테스트 (이게 정답!!)**

13. 테스트 이슈 관리 이해(4)

- 이슈가 생기면 어떻게 관리하나?

- 이론상으로 차후 수업예정

14. 애플리케이션 개선 과정(5)

- 이론상

15. 프로그램 성능 향상 기법(5)

<포트폴리오>

게시판 만들기

강사님이 CRUD 중 한곳을 오류를 만들어 두었다. 나머지 세개는 완벽

네 곳 중 어디가 문제인지 찾아서 한 곳을 해결하면 전체가 구현되어있으므로 잘 돌아가게 만들어둠

- jUnit 으로도 테스트해야함!!

- 직접적으로 잘못된 오류까지 수정하고

1. 개발 환경 구성 확인(3)

* 버전까지 구체적으로 안 적으면 틀린 것!!!!!

1. 프로세서

* Intel core i5-7400 CPU @ 3.00GHz

1. RAM

* 용량만 8GB

1. 시스템 종류

* 64비트 운영체제

1. OS

* Windows 10 pro

1. 컴파일러

* JDK8

1. IDE (통합 개발 툴, Integrated Development Environment)

* 이클립스 (버전까지 적어야!!!)
* Eclipse IDE for Enterprise Java Developers.

(Eclipse 2018-12)

1. Build

* Maven (우리가 사용하는 것)

1. Test 도구

* **JUnit 4**

1. 형상관리도구

* Git (우리가 배운 것), SVN

1. DBMS

* Oracle xe 11g

컴퓨터 부품의 정보 – 메모리, 가격

컴퓨터에 대한 기본정보보기 (제어판 -> 시스템 및 보안 -> 시스템)

* 시스템

(프로세서, 설치된 메모리, 시스템 종류, 펜 및 터치)

8. 자바스크립트 유효성 검사(5)

: 자바 스크립트를 글자로 적어야 한다

이클립스 켜서 타이핑 직접하고 적어라

* Validation의 종류
* 유효성이 뭐고, 만들고 뭘 체크해야하는지

11. 블랙 박스 테스트? 화이트 박스 테스트?(3)

**[ 블랙박스 테스트]**

1. 등기 분할
2. 경계 값 분석
3. 결정 테이블 테스트
4. 분류 트리 기법
5. 상태 전이 테스트 (필기 시험에서 자주 나옴)
6. 유즈 케이스 테스트

: (사용자의 입장과 시스템의 입장)

사용자의 입장에서 시스템이 어떻게 보여줄것인가?

잘못적은 것에 대해 어떻게 보여줄것인가

맞게 적은 것에 대해 어떻게 보여줄것인가

– ex) 회원가입 트랜젝션의 단위

: 아이디 입력이 잘못되면 시스템이 알려줘야함

**[ 화이트 박스 테스트 ]**

1. 구문 커버리지

: 구문을 짰으면 그것이 다실행이 되야함

1. 결정 커버리지
2. 조건 커버리지

12. 단위 테스트의 종류 특징(4)

**-> Mock 단위 테스트**

(junit 보다 좀 큰 단위의 테스트)

13. 테스트 이슈 관리 이해(4)

- 테스트를 했는데 안됐을 때 관리 할 수 있는 문서도 존재한다