1슬라이드 : 안녕하세요, 11조 발표를 맡은 김준성입니다.

2슬라이드 : 인덱스는 Development, Design, Execution, Test로 구성되어 있습니다.

3슬라이드 : 먼저 Development입니다.

4슬라이드 : 개발 OS는 window8/10에서 진행하였고, Tool은 JAVA입니다. 또한, 코드를 공유하며 변경사항을 진행하기 위하여 GitHub를 이용하였습니다.

5슬라이드 : 저희 개발에 있어서 process model은 waterfall model로 채택했습니다. Waterfall model은 계획, 요구조건, 설계 및 구현, 테스트, 유지보수의 단계로 이루어져 있으며, 저희는 만들고자 하는 제품의 요구사항이 명확히 주어졌으므로 프로그래밍 하는 과정에서 큰 변화가 없을 것이라고 생각하여 채택하였습니다. 먼저 계획하는 단계에서는 어떻게 프로그램을 만들지 조사하며 프로그래밍 언어, 개발 도구 등을 정했습니다. 요구 분석 단계에서는 SRS를 작성하여 제품의 전체적 기능과 설명, UseCase, 시스템 특징, 기능의 요구 사항, 외부 인터페이스 요구사항, 제한 조건 등을 정리하였습니다.

6슬라이드 : 두번째로 Design입니다.

7슬라이드 : 저희의 view와 controller입니다. 뷰는 JavaFX SceneBuilder2.0로 fxml파일을 열어서 디자인을 했습니다. 우선 화면을 크게 2개로 나눠줘야 하므로 VBox를 2개를 생성하여 panel을 만들고 각각 내부에 글을 받아오거나 쓸 TextArea를 생성하였습니다. 그리고 edit, load, save, merge, compare 등 기능이 되는 버튼을 만들고 각각의 버튼에 onAction을 설정하여 이벤트가 발생하면 컨트롤러에 전달할 수 있게 했습니다.

프로젝트의 규모가 크지 않다고 생각하여 컨트롤러는 하나만 만들었습니다. 뷰에서 onAction 이벤트가 발생하면 컨트롤러에 각각 해당하는 eventlistener로 들어가서 실행이 되게끔 했습니다.

8슬라이드 : MVC 패턴입니다. MVC는 Model, View, Controller의 약자 입니다. 하나의 애플리케이션, 프로젝트를 세가지의 역할로 구분한 패턴입니다.  위의 그림처럼 사용자가 controller를 조작하면 controller는 model을 통해서 데이터를 가져오고 그 정보를 바탕으로 시각적인 표현을 담당하는 View를 제어해서 사용자에게 전달하게 됩니다. 세가지로 분리되어 각자의 역할에 집중할 수 있게끔 하여 개발을 하고 그렇게 애플리케이션을 만든다면, 유지보수성, 애플리케이션의 확장성, 그리고 유연성이 증가하고, 중복코딩이라는 문제점 또한 사라지게 되는 것입니다.

9슬라이드 : 세번째로 실행화면과 구현입니다.

10슬라이드 : 첫번째 그림은 처음 스크린의 모습이고 두번째는 compare한 후의 모습, 세번째는 merge를 한 후의 모습입니다.

11슬라이드 : 4번째로 테스트입니다.

12슬라이드 : 저희는 Functional unit test를 controller test 초점에 맞춰 진행하였습니다. Controller와 data, view를 테스트진행하였습니다.

13슬라이드 : 질문이 있으시다면 받도록 하겠습니다.

14슬라이드 : 감사합니다!