문제 1. 본인이 생각하는 프로그래밍이란 무엇인가요?

답변 : 프로그래밍은 컴퓨터에게 명령을 내리는 과정이라고 생각합니다. 내가 짠 소스코드를 컴퓨터가 이해하여 원하는 대로 실행을 해주는 작업이라고 생각합니다.

문제 2. ES6이전버전과 ES6버전이후에 차이에 대해서 설명하세요.

답변 :  let, const, 화살표함수, 클래스, 모듈 등을 도입하였다.

문제 3 Ajax가 나오고나서 어떤 것이 가능해졌나요?

답변 : 비동기적으로 서버와 브라우저가 데이터를 교환할 수 있게 되었다. 웹페이지에서 변경이 필요한 부분만 다시 렌더링 할 수 있게끔 되었다.

문제 4. [www.naver.com](http://www.naver.com)을 브라우저 주소창에 쳤을 때 동작하는 과정을 설명.

네이버 닷컴 이라고 치면 -> 네이버 서버를 찾아감. -> DNS서버가 실제적인 서버가 어디에 있는지 알고있는 애인데 얘가 어디로 연결해줄게 라고 함. -> [https://www.naver.com -> 난 서버와 통신을 https로 할거야](https://www.naver.com/)

-> 서버의 기본설정이 대부분 index.html(메인페이지)로 되어있다. 그래서 메인페이지를 뿌려줌.

-> 텍스트파일로 이뤄진 index.html 을 파싱한다.

(파싱 : 텍스트파일로 이뤄진 파일들을 브라우저가 실행하기위해서 그 텍스트들을 해석하는 것)

-> 브라우저가 이해할 수 있는 구조(자료구조)에 맞게 만들어서 메모리에 올려놓는다.

-> 한줄한줄 읽는다. 읽는 과정에서 html 등 을 만난다. 그러면 이 부분을 메모리에 기억

-> 그러다가  도중에 link태그 만났다. css를 요청해 서버에다가 나 style.css라는 거 줘 , 그래 줄게

-> CSS를 파싱함. 파싱도중 import를 만나면, 다시 또 서버에 요청과 응답

-> CSS파싱이 끝나면 중단한 html을 다시 읽어들어가기 시작 dom트리를 완성시켜나감.

-> DOM트리와 CSSOM트리 합쳐서(Merge) Render Tree를 만듬.

-> 그린다.(display)

문제 5. Script태그를 body태그 마지막 쪽에 써주는 이유는?

답변 : body태그 이전 타임이HTML파서가 HTML요소를 모두 파싱하여 DOM객체를 생성한 이후이며, 자바스크립트가 실행되기 이전에 HTML요소가 파싱되어 렌더링되므로 페이지 로딩 시간이 단축된다. 또한 DOM이 완성되지 않은 상태에서 자바스크립트가 DOM을 조작하는 에러가 발생할 우려도 없다.