안녕하십니까. 1조에서 발표를 맡게된 김채원이라고 합니다.
저희가 만든 프로그램은 “썸띵”으로, 두 사람 사이의 “썸”을 도와주는 채팅프로그램입니다./

이 프로그램의 실행과정에 대해서 간략히 설명드리자면 우선 두 사람이 서버파일과 클라이언트 파일을 각각 맡아서 실행시켜 통신을 시작합니다. 그 후 서로 자기소개, 공통질문 답변, 질의응답, 밸런스 게임을 통해서 서로가 얼마나 잘 맞는지. 마지막엔 궁합 점수가 나오게 됩니다. 점수를 통해서 그 둘에게 알맞은 결말과 애프터 약속을 추천하게 되면서 프로그램이 종료됩니다./

이 프로그램에서의 핵심 기능은 바로 소켓을 이용하여 두 컴퓨터 간에 대화를 주고받는 것입니다. 두 컴퓨터가 같은 와이파이에 접속되어 있다면 서버의 ip주소로 클라이언트가 접속할 수 있게 됩니다. 이때 파이선에서 통신을 하기 위해 불러와야하는 라이브러리를 소켓이라고 합니다. 소켓을 통해 프로그램은 클라이언트에게 데이터를 보내고 서버는 클라이언트에게 데이터를 받게 됩니다. 이를 예시를 들어 설명해드리겠습니다. /

우리가 매일 쓰는 네이버가 서버에 해당되고, 다양한 플랫폼을 통해서 네이버에 접속하는 사람들이 클라이언트에 해당됩니다. 사람들이 접속을 시도하면 이 신호를 서버에 보내게 되고, 그 후 접속허락 이라는 데이터를 네이버가 사람들에게 보냄으로써 사람들이 웹페이지에 접속할 수 있게 되는 것입니다. 이러한 데이터를 주고 받을 때 이용되는 매개체가 바로 소켓입니다. /

서버와 클라이언트는 각각의 소켓을 사용하는데, 먼저 서버에서는 소켓을 생성하고, 바인딩을 해야합니다. 바인딩이란, 운영 체제에 소켓을 어떠한 형태로, 어떤 포트로 사용하겠다는 것을 알려주는 것입니다.

바인딩을 마친 후에는 listen을 통해서 다른 클라이언트가 들어올 때까지 접속을 대기합니다. 클라이언트가 서버 ip주소와 생성해놓은 포트번호로 접속을 하면은 데이터를 주고 받을 준비가 됩니다.

이제 서로의 이름을 물어보고, 서로에게 질문을 하기 위해서는 기본적인 정보를 알아야 하기 때문에 공통질문을 각자의 창에 띄웁니다. 질문에 대한 답은 send와 recv(리시브)를 통해서 받을 수 있습니다. /

그 다음에는 먼저 서버에 있는 사람이 질문을 하고, 그 질문을 클라이언트에게 전송하면, 클라이언트가 그 질문을 받아서, 화면에 출력하게 되면, 클라이언트에 있는 사람이 대답을 하게 됩니다. 이 대답은 다시 서버에 전송되어서 화면에 출력해줍니다. 이에 대한 자세한 내용은 뒤에 도전적인 부분을 이야기할 때 추가적으로 설명하겠습니다.
서버가 질문을 5번 하면은 이제 클라이언트가 질문할 차례가 되면서 앞의 과정을 반복합니다. /

질의응답 시간이 끝나면, 밸런스 게임을 시작합니다. 밸런스 게임은 썸을 도와준다는 취지에

맞게 연애 밸런스 게임으로 설정했습니다. 밸런스 게임의 원리는 화면에 출력한 질문에 대해서 둘이 같은 답을 하는지와 그 답을 하는데 얼마나 시간이 걸리는지를 측정해서, if문으로 비교한 후 각각 return 하는 점수를 20, 10, 0을 부여합니다. 총 다섯 번의 질문을 통해서 나온 점수를 합산해서 50점 이상이면, 애프터로 밥을 먹으러 가는 것을 추천하자는 의미에서 함께 먹을 음식 메뉴를 정하는 단계로 넘어가게 됩니다.

"먹고 싶으신 음식이 있으신가요?"라는 질문과 함께 두 사람으로부터 동시에 대답을 받고, 그 대답이 같을 경우에는 그 음식을 먹으러 가라는 문구를 출력하고, 채팅 프로그램이 종료됩니다. 그러나, 먹고 싶은 음식이 다를 경우에는 음식 메뉴를 추천하는 함수가 실행됩니다. /

메뉴 추천 함수에서는 서버 컴퓨터에서 실행하는 파이썬 파일에 food라는 변수에 다양한 음식들을 리스트의 형태로 선언합니다. 그리고, random모듈의 randint를 이용해서 범위를 0부터 food라는 리스트에 들어있는 item의 길이보다 하나 작은 수까지로 설정해서 무작위로 index값을 하나 만들도록 합니다. 그리고 이 index값을 통해서 무작위로 리스트에 있는 item을 반환합니다.

클라이언트는 앞서 서버에서도 공통적으로 출력한 질문에 대답을 서버로 보내면 서버에서는 이에 대한 답을 받아서 두 개의 답이 같은지를 if문을 이용해 비교합니다. 만일 두 명의 의견이 같다면 그 음식의 이름을 return해주고 그렇지 않으면, 랜덤으로 메뉴 하나를 return하도록 하고 해당 음식 이름을 화면에 출력하고 나서 채팅 프로그램을 종료하게 됩니다. /

저희 프로그램이 동작하는 영상을 잠시 보시겠습니다. 해당 동영상은 개인 핫스팟을 통해 연결된 두 컴퓨터가 채팅하는 것입니다.

<영상 보여주고 나서> / 지금 영상은 점수가 60점일 때, 메뉴 추천 함수가 실행되어서 서로 먹고 싶은 것이 달랐을 때 나오는 결과를 보여주는 영상입니다.

<각자 다른 결과가 나온 사진도 부착>
위의 사진은 먹고 싶은 것이 서로 같은 경우
아래 사진은 서로 밸런스 게임에 대한 대답이 모두 다른 경우입니다./

저희가 짠 프로그램에서 도전적이었던 부분은 먼저, 순서대로 데이터를 받을 수 있게 서버 코드와 클라이언트 코드를 짜는 과정입니다. 같이 질문을 하고, 같이 대답을 하게 되면 원활한 대화가 되지 않을뿐더러, 순서가 정해지지 않으면 변수에 저장되는 데이터의 값을 send(), recv()하는 과정에서 데이터가 섞일 위험이 있었기 때문입니다.
그래서 동시에 질문을 띄우기보다는 서버에서 질문을 띄우고, 그 질문을 클라이언트에게 보내고, 클라이언트가 질문에 대한 답을 하면, 그 답을 다시 서버에 보내는 형식으로 순서를 정하게 되었습니다.

두 번째로 도전적이었던 부분은 대화가 끝난 후에 밸런스 게임을 실행하도록 하는 것이었습니다. 저희가 참고한 코드에서는 while True를 이용하여 지속해서 대화가 가능하도록 했기

때문에 대화를 끝내고 그 다음 단계로 넘어가는 것이 어려웠습니다. 이 때 저희는 파이썬이 코드를 짠 순서대로 실행된다는 점을 이용해서 while문이 다 끝난 후에 else를 이용하여 밸런스 게임을 진행하도록 하였습니다.

마지막으로는 엔터를 치면 오류가 나는 부분을 해결하는 과정입니다. 질문이나 답장을 받은 후에 실수로 엔터를 치게 되면은 프로그램이 멈추는 오류가 있었습니다. 엔터를 누르게 되면 한쪽에서는 데이터를 보냈다고 생각한 것인데, 받는 입장에서는 빈문자열이 왔기 때문에 화면에 표시할 것이 없게 되고 끝없이 데이터를 기다리면서 결국 프로그램을 더 이상 이용할 수 없는 것이었습니다.
그래서 생각한 것이 질문하고 싶지 않거나, 대답하고 싶지 않을 때 엔터를 치면 그 의사를 표현한 것으로 코드를 짰습니다. 이렇게 되면, 서버 입장에서도 빈 문자열을 보내지 않게 되고, 클라이언트 입장에서도 빈 문자열을 받지 않아서 기다리지 않아도 되기 때문에 프로그램이 의도치 않게 멈추는 경우는 없어졌습니다./

외부 코드에서는 반복적으로 주고 받는 채팅에 관한 것을 참고하였습니다. 저희가 참고한 외부코드는 while True를 이용해서 서로 제한 시간 없이 계속해서 대화만 가능하도록 했기 때문에, while문의 조건을 I가 5보다 작다로 설정해서 한 주기가 돌면 변수에 1을 더해서 총 5번이 실행되도록 지정하였습니다.
또한 주소를 할당하는 과정을 참고하였고, 데이터를 한국어로 송수신하는 과정을 참고하였고, 나머지 부분은 저희가 상의해서 코드를 짰습니다. /