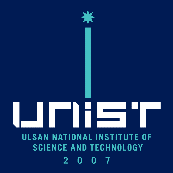
**“유니스트 적정기술 프로젝트 – Python – Analog”**

**[main.py]**

**아날로그 게이지를 읽는데 필요한 변수들인 ‘디바이스의 이름’, ‘화면의 각도(카메라를 처음 설치했을 때 기기의 배치에 따라 화면이 회전했을 때의 경우를 대비’, ‘단위’, ‘최솟값과 최댓값’, ‘최솟값과 최댓값이 가지는 각도’를 통해 값을 읽는다. 최솟값과 최댓값의 위치를 파악하고 그 사이의 거리와 값을 등분하여 바늘의 위치와 매칭하여 값을 읽을 수 있다. Main.py에서는 변수를 받아오고 func 함수를 실행하는 역할을 한다.**

**1. Library 설치**

****

파이썬 라이브러리 호출한다. func 파일은 같이 첨부되어 있는 func.py 파일이다.

**2. Argument 설정**



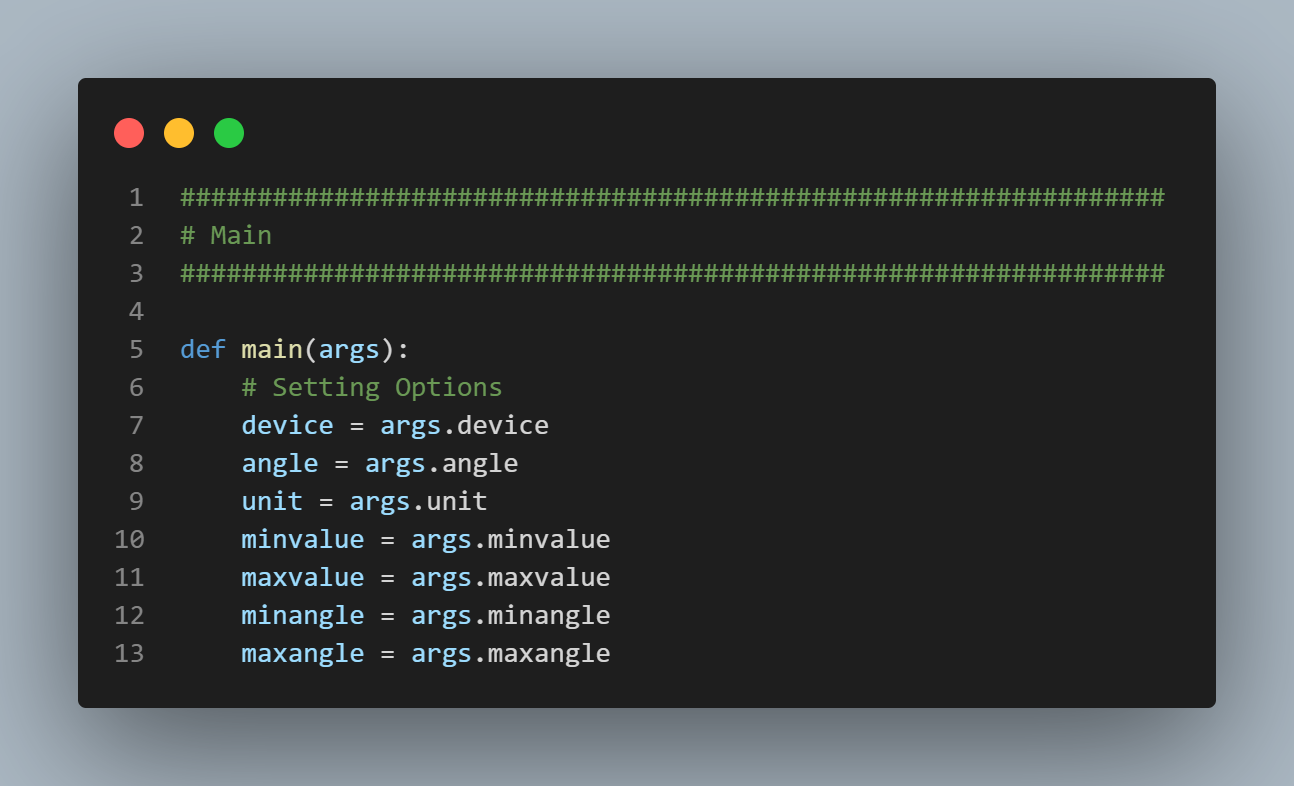
argparse 라이브러리에서 가져온 Parser 함수를 통해 Argument 값을 각각 지정한다. 변수 이름에서 확인할 수 있듯이 1) ‘device’ = 기기 이름, 2) ‘angle’ = 화면의 회전 각도 3) ‘unit’ = 단위 4) ‘minvalue’ = 측정장비의 최솟값 5) ‘maxvalue’ = 측정장비의 최댓값 6) ‘minangle’ = 최솟값의 위치 7) ‘maxangle’ = 최댓값의 위치를 각각 표현한다.

**3. Config 설정**

****

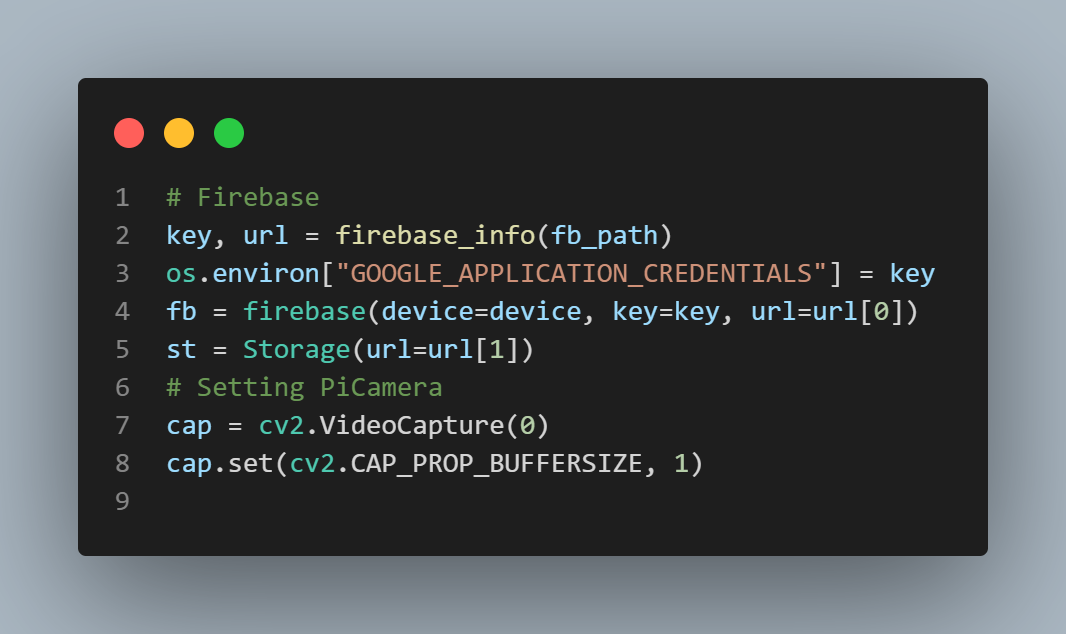
기본 설정된 값 및 파일 지정 경로를 불러온다.

**4. 변수 지정**

****

arguments 받아온 값으로 코드에서 사용할 변수를 지정한다.

**5. Firebase 지정 및 파이카메라 선언**



Firebase url의 경우 만들어진 key.json을 통해 url 선언을 하였고, 파이카메라의 경우 이미지 생성을 위해 사용되는 video capture 기능을 사용하여 선언하였다.

\* Firebase 프로젝트 설정은 https://firebase.google.com/?hl=ko 에서 할 수 있다.

추가 링크 : <https://blog.naver.com/PostView.nhn?blogId=dsz08082&logNo=221992549793&parentCategoryNo=&categoryNo=124&viewDate=&isShowPopularPosts=true&from=search>

파이어베이스를 이용하지 않고 일반적인 서버에 연결할 때에는 socket 라이브러리 사용

<https://m.blog.naver.com/heennavi1004/222051331011>

**6. 카메라 작동 알고리즘**

****

Line 4-11: 아날로그 게이지의 외부 프레임을 찾는 과정

Line 13-15: 이미지 데이터 저장

Line 17-19: 게이지를 읽고 값 저장

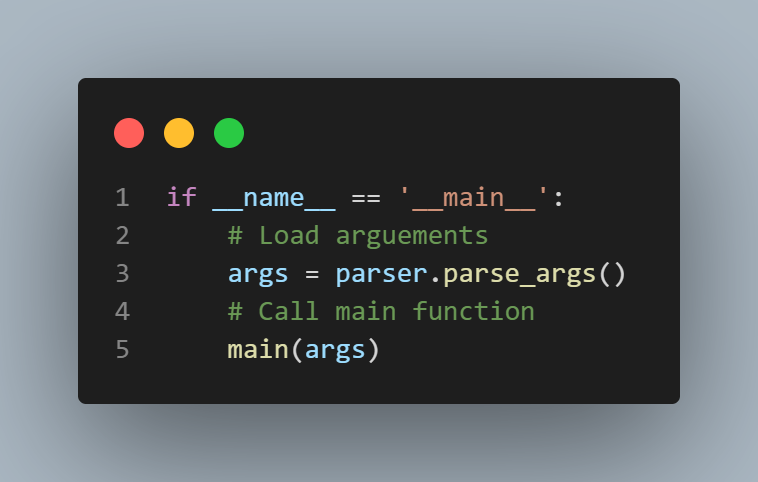
Line 21-22: 위의 arguments 값을 읽어 들이고 저장

Line 28-33: 읽어 들인 값에 대한 로그 출력

Line 35: Firebase로 데이터 전송

Line 37-38: Break을 통해 제어 흐름을 중단하고 빠져나옴.

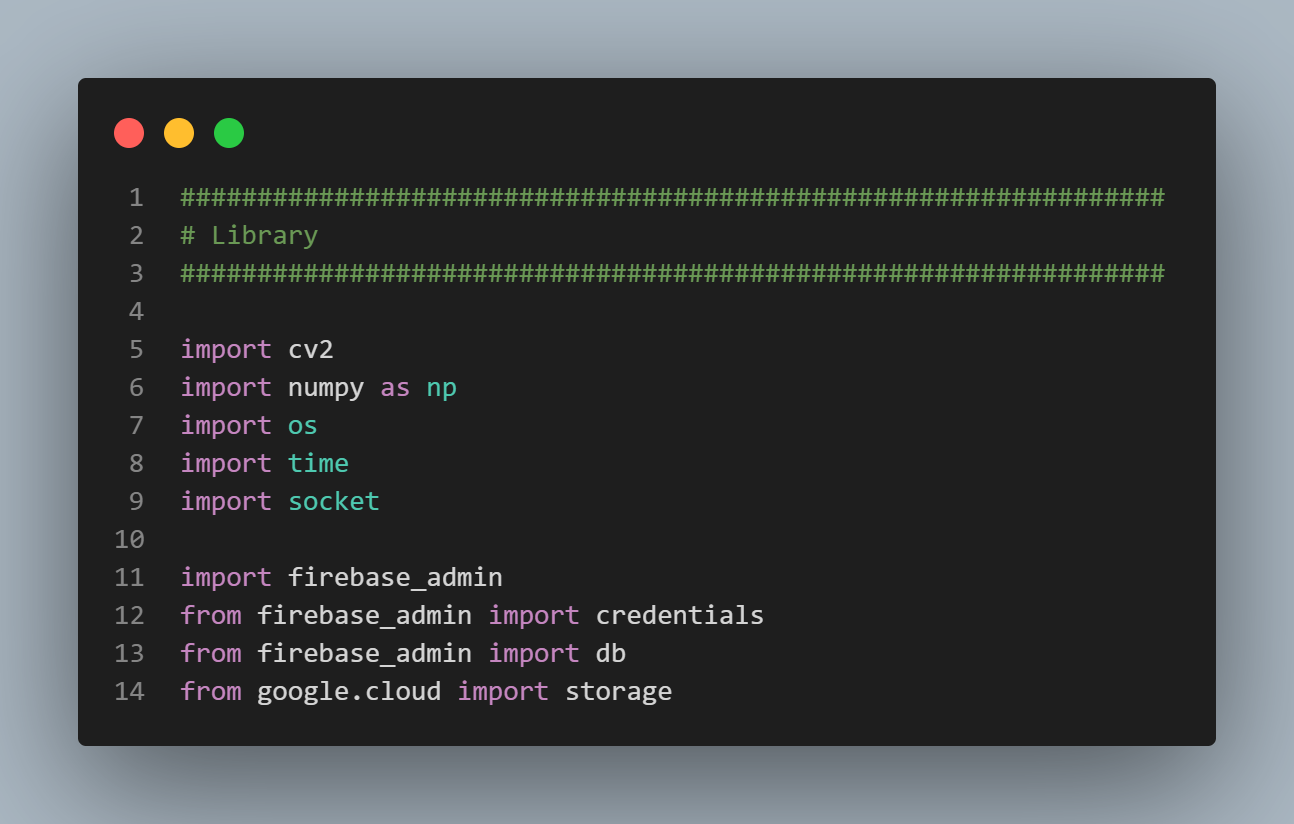
**7. Arguments 값 불러오고 main function 실행**



**[func.py]**

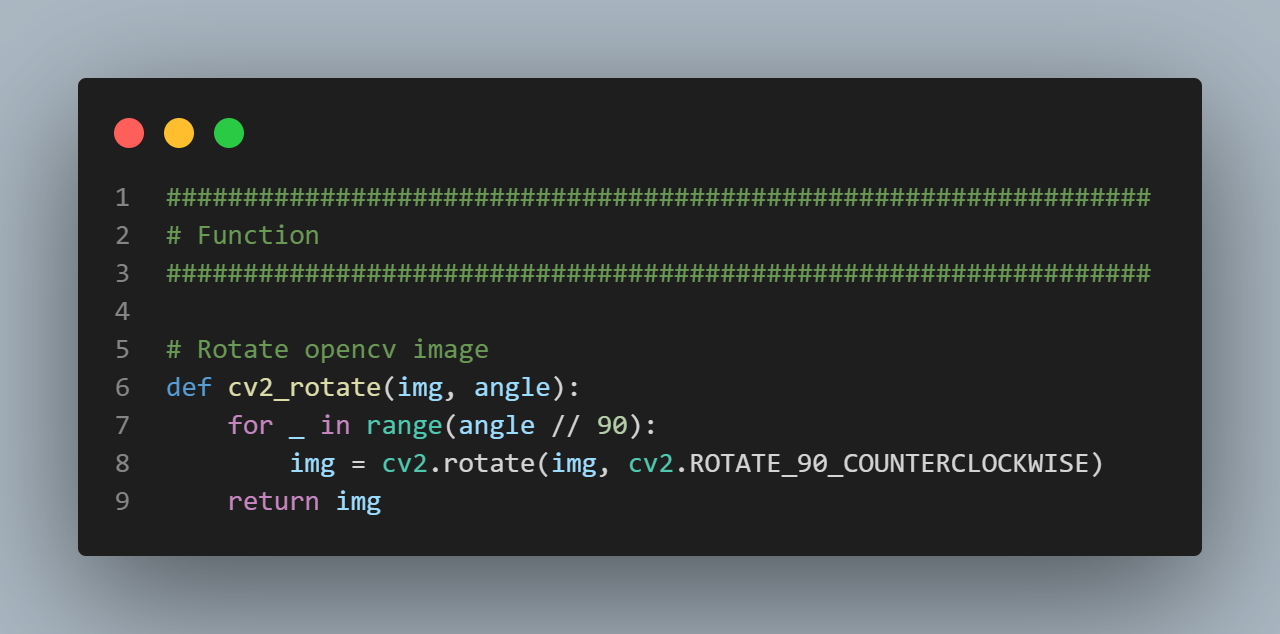
**Gauge 바늘의 경사각을 계산하기 위해서, Gauge의 가장 자리를 찾은 뒤 원을 생성하고 단위 간격과 격자 좌표를 설정 후 이미지에 기록하여 저장한다. 그 후, 원과 선이 만나는 지점을 찾고 각도를 계산하여 Gauge 값을 추출한다.**

**1. Library 설치**

****

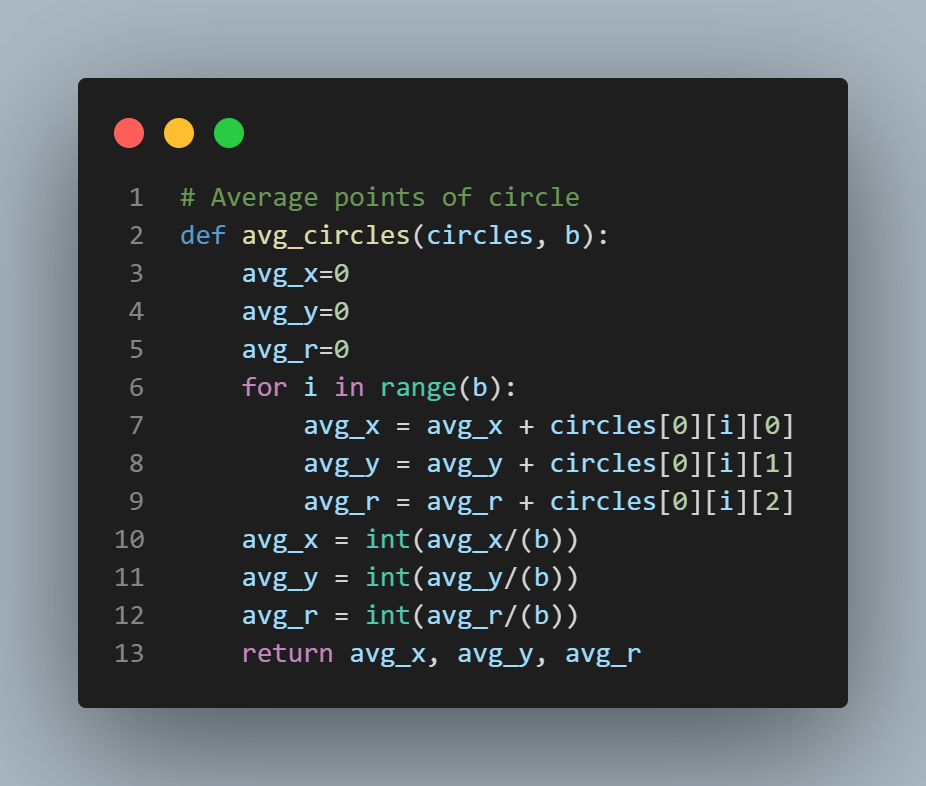
파이썬 라이브러리 호출한다.

**2. 이미지 회전**

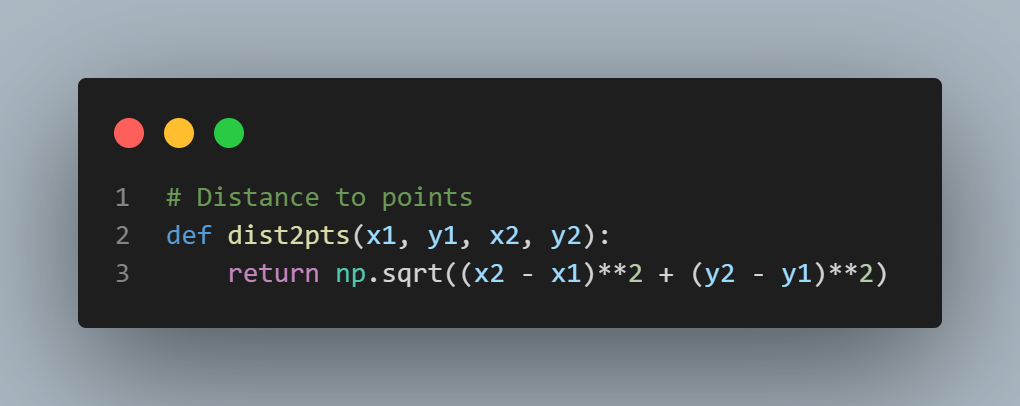


파이카메라가 원하는 이미지의 방향을 출력하지 않을 때 이미지 회전을 위한 코드이다.

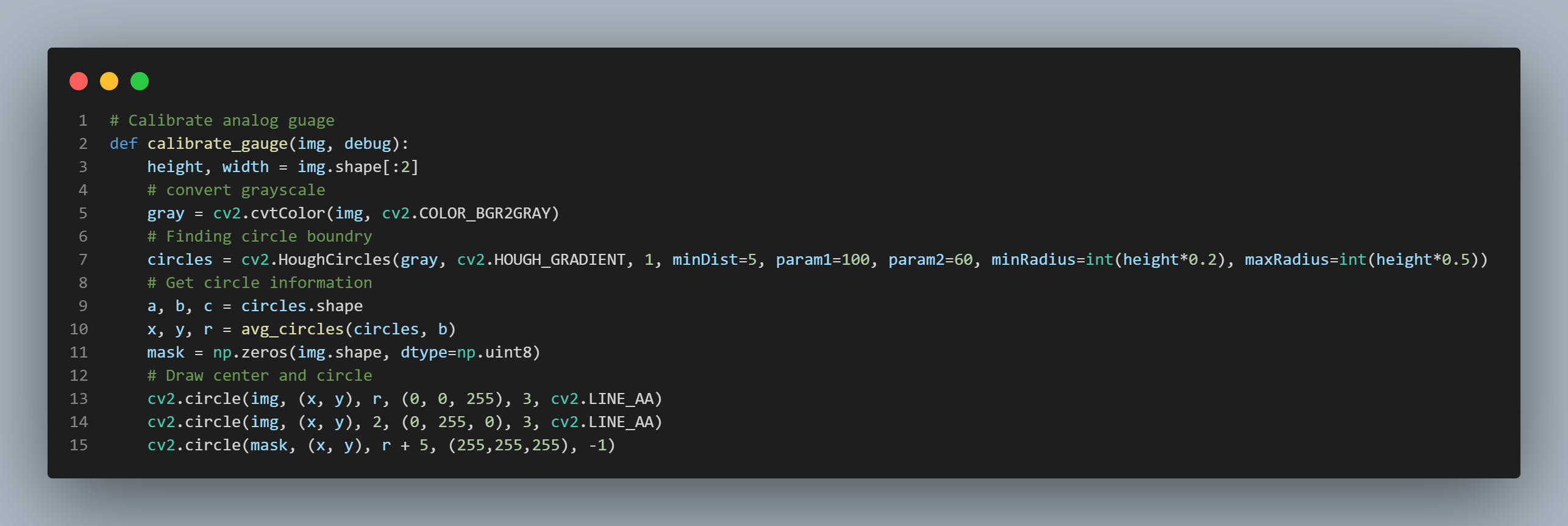
**3. 원들의 평균점 계산**



**4. 점과 점 사이의 거리 계산**

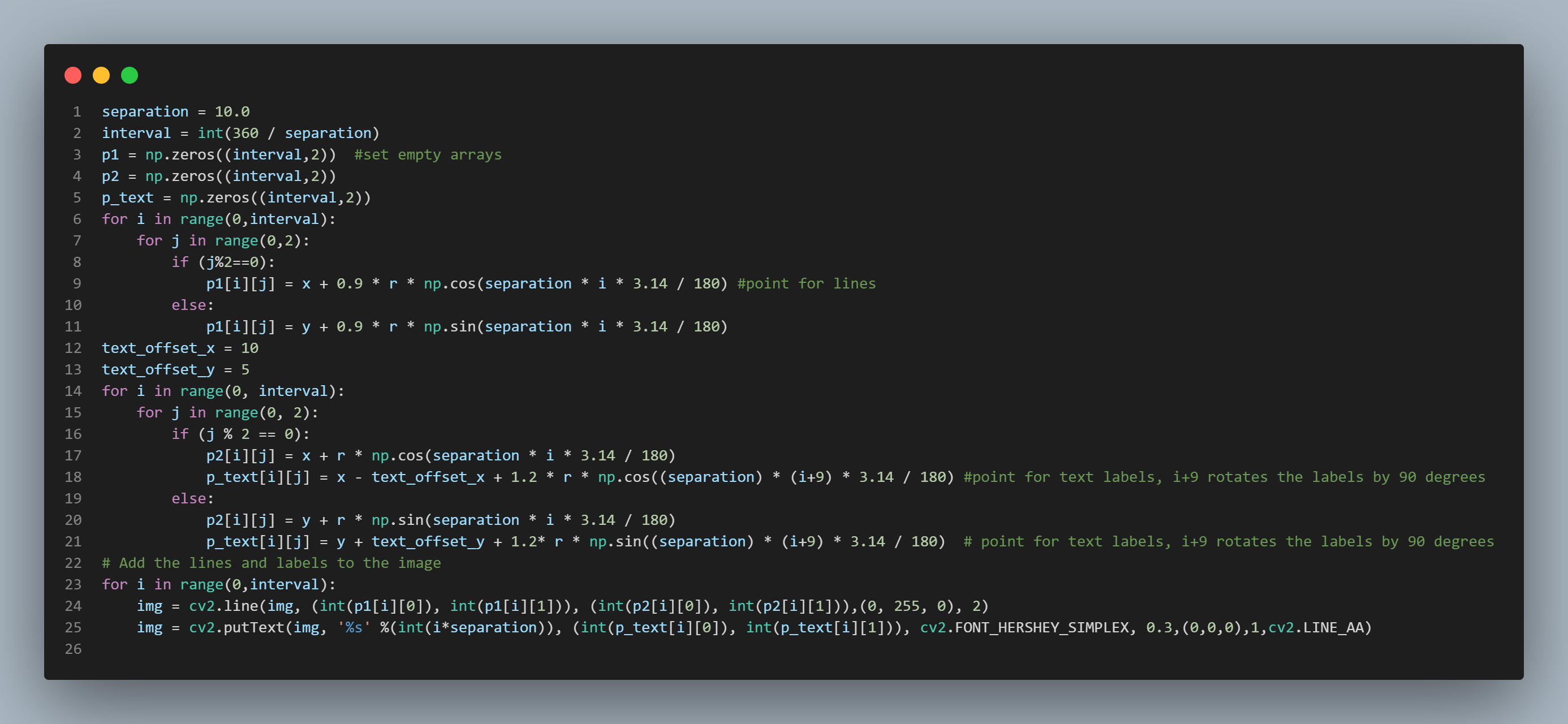


**5. 사진을 흑백으로 바꾸고 원의 가장자리를 찾은 뒤 원 생성**



원활한 이미지 프로세싱을 위해서 사진을 흑백으로 변경하는 것이 좋다. 그 외의 코드는 각각의 주석에 설명되어 있다.

**6. 단위 사이의 간격을 결정 및 이미지로 저장**



격자의 좌표를 설정하고 그 값에 수치를 기록한 뒤 이미지에 기록하여 저장한다.

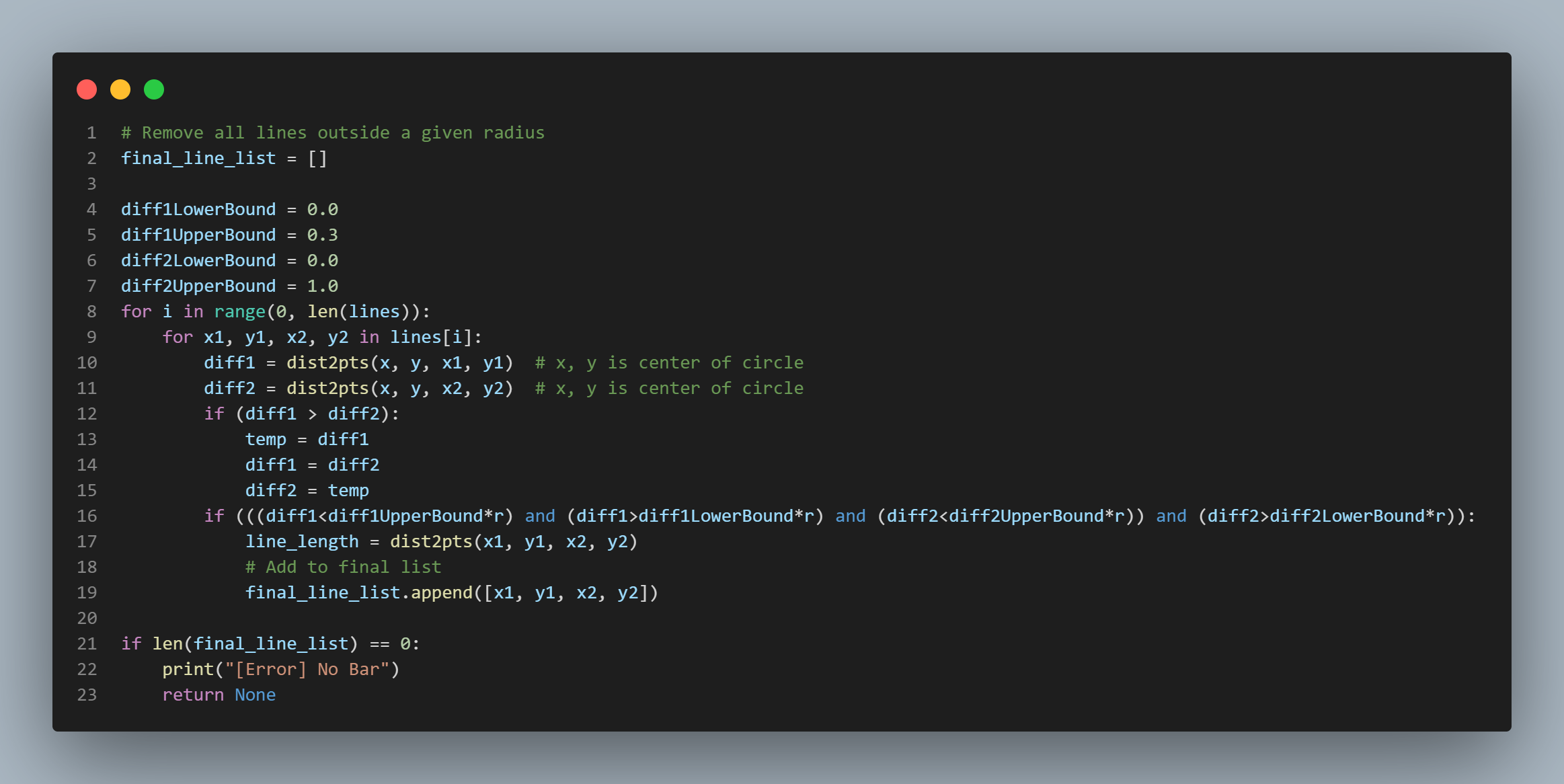
**7. 이미지에 원을 그리고 저장**



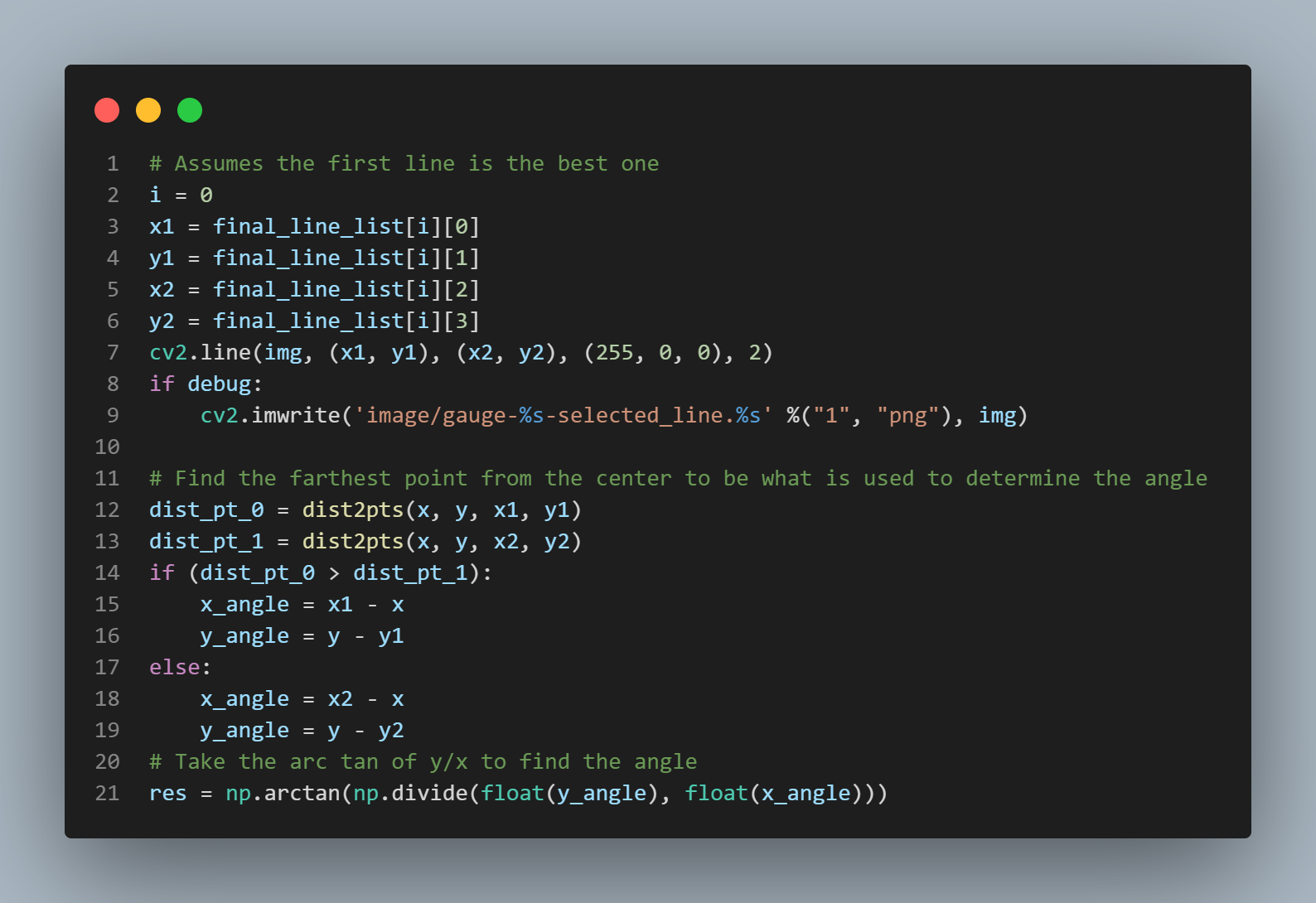
**8. 현재 값을 식별 및 저장**



**9. 원 바깥의 선 삭제**



**10. 원과 선이 만나는 지점 및 각도 계산**



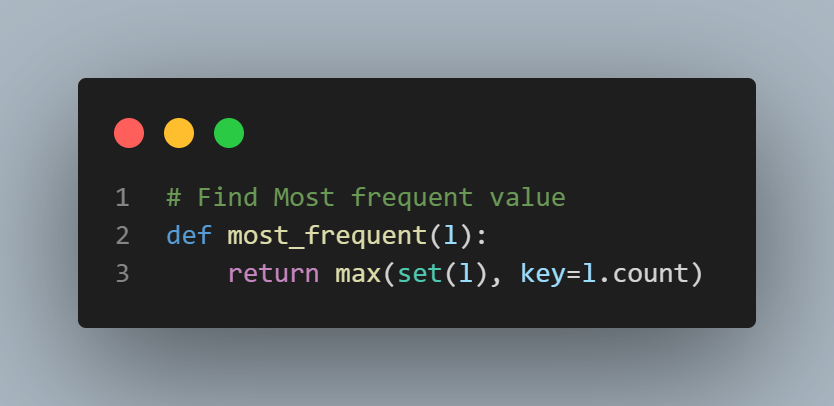
각도 결정을 위해 원과 선이 만나는 지점을 찾고 각도 구하는 코드이다.

**11. 그려진 눈금과 바늘의 경사각 결정 및 계산**



그려진 눈금과 바늘의 경사각을 결정하고 계산한다. 그리고 final angle 값을 추출한다.

**12. 가장 자주 나오는 값을 추출**



**13. Firebase 정보 추출**



연결하는 소켓과 ip는 때에 따라서 수정해야 한다. 사용하는 기기의 IP 주소를 Line 218 칸에 있는 IP 대신 입력해주면 된다.

**14. Firebase 관련 클래스 코드**



Firebase 설정을 위해 기존의 같은 태그를 없애고 기기의 ip를 서버로 업데이트하는 코드이다.



Firebase에 데이터를 업로드하고 저장하며 기기의 IP를 서버의 태그 하단에 저장한다.



Firebase 데이터를 클리어하는 코드와 이미지를 업로드하는 코드이다. 이 코드에서는 이미지를 ‘analog.png’로 업로드 하였는데, 파일 명을 변경할 수도 있다.