In [1]:

```
!pip install cvzone
!pip install cv2
```

```
Collecting cvzone
  Downloading cvzone-1.5.6.tar.gz (12 kB)
  Preparing metadata (setup.py): started
  Preparing metadata (setup.py): finished with status 'done'
Requirement already satisfied: opencv-python in c:\users\abc\anaconda3\lib\s
ite-packages (from cvzone) (4.5.5.62)
Requirement already satisfied: numpy in c:\users\abc\anaconda3\lib\site-pack
ages (from cvzone) (1.22.4)
Building wheels for collected packages: cvzone
  Building wheel for cvzone (setup.py): started
  Building wheel for cvzone (setup.py): finished with status 'done'
  Created wheel for cvzone: filename=cvzone-1.5.6-py3-none-any.whl size=1877
1 sha256=b5effb1b6bd2e5b4c05f586ed9edde3f864e8f4ab6ee164fc43aa015a3321e96
  Stored in directory: c:\users\abc\appdata\local\pip\cache\wheels\67\60\9a
\e5060027d8eae2c01ba872fdbc094a6fe3fee15794ee3dc709
Successfully built cvzone
Installing collected packages: cvzone
Successfully installed cvzone-1.5.6

WARNING: There was an error checking the latest version of pip.
ERROR: Could not find a version that satisfies the requirement cv2 (from ver
sions: none)
ERROR: No matching distribution found for cv2
WARNING: There was an error checking the latest version of pip.
```

In [2]:

```
!pip install mediapipe as mp
```

```
Requirement already satisfied: mediapipe in c:\users\abc\anaconda3\lib\site-
packages (0.8.9.1)

ERROR: Could not find a version that satisfies the requirement as (from vers
ions: none)
ERROR: No matching distribution found for as
WARNING: There was an error checking the latest version of pip.
```

In [3]:

```
import cv2
import cvzone
import mediapipe as mp
```

In [4]:

```
from cvzone.FaceMeshModule import FaceMeshDetector
camera = cv2.VideoCapture(0)
```

In [5]:

```
Face_Detector = FaceMeshDetector(maxFaces=1) # Setting hyperparameter as one, so it detect
while True:
    success , image = camera.read()
    image , faces = Face_Detector.findFaceMesh(image)
    # If any face is available
    if faces:
        face = faces[0]
        LeftEYE = face[145] # Landmark of left eye
        RightEYE = face[374] # Landmark of right eye
        cv2.circle(image, LeftEYE, 10, (255, 0, 255), cv2.FILLED) # visualizing left eye
        cv2.circle(image, RightEYE, 10, (255, 0, 255), cv2.FILLED) # visualizing right eye
        cv2.line(image, LeftEYE, RightEYE, (0, 200, 0), 3) # Drawing Line to check for dist
        w, _ = Face_Detector.findDistance(LeftEYE, RightEYE) # storing distance between eye
        W = 6.3 # Average Pupillary Distance
        # Finding the Focal Length
        d = 44
        f = (w*d)/W
        print(f)
    cv2.imshow("Calculate Distance", image)
    cv2.waitKey(1)
```

```
---------------------------------------------------------------------------
-
KeyboardInterrupt                         Traceback (most recent call las
t)
<ipython-input-5-35812ef84063> in <module>
     18         print(f)
     19     cv2.imshow("Calculate Distance", image)
---> 20     cv2.waitKey(1)

KeyboardInterrupt:
```

In [ ]:

```
import cv2
import cvzone
from cvzone.FaceMeshModule import FaceMeshDetector
camera = cv2.VideoCapture(0)
Face_Detector = FaceMeshDetector(maxFaces=1)
while True:
    success , image = camera.read()
    image , faces = Face_Detector.findFaceMesh(image,draw=False)
    if faces:
        face = faces[0]
        RightEYE = face[374] # Landmark of Right eye
        LeftEYE = face[145] # Landmark of left eye
        w, _ = Face_Detector.findDistance(LeftEYE, RightEYE) # width between eyes in pixel
        W = 6.3 # Average Pupillary Distance
        f = 685.3346979937062 # Calculated Focal Length
        d = (W * f) / w # distance between camera and eyes
        cvzone.putTextRect(image, f'DISTANCE: {int(d)}cm',(face[10][0] - 100, face[10][1] -
    cv2.imshow("Calculate Distance", image)
    cv2.waitKey(1)
```