```
#   ************** START OF PROGRAM ****************    #
#   ************** ACTUAL PROGRAM ****************    #
#   **************** PROGRAM TO PREDICT HAND GESTURES AND UPLOAD THE
PREDICTIONS TO FIREBASE ****************    #
# *************** Import from keras h5 model Prediction.py File
*************** #

import tensorflow.keras
from PIL import Image, ImageOps
import numpy as np

# *************** Import from OpenCV DC.py File *************** #
import cv2

# *************** Import from FB DB upload.py File *************** #
from firebase import firebase

# *************** Additional Imports *************** #
import time
import threading  # Process Schedulers

# *************** Determine the way floating point numbers,
# arrays and other NumPy objects are displayed *************** #
np.set_printoptions(suppress=True)

# *************** Accessing firebase real-time database project
*************** #
firebase = firebase.FirebaseApplication("https://rtmc-hg-default-
rtdb.firebaseio.com/", None)

# *************** Assigning the saved keras.h5 model to a variable
*************** #
model = tensorflow.keras.models.load_model('keras_model.h5')

# *************** Assigning/Re-assigning the data in firebase with a
Startup String *************** #
firebase.put("/Data", "Preds", "** PROGRAM START **")
time.sleep(3)

# *************** Using OpenCV VideoCapture for capturing live video
and setting window parameters *************** #
kernel = np.array([[-1, -1, -1], [-1, 9, -1], [-1, -1, -1]])
cap = cv2.VideoCapture(0, cv2.CAP_DSHOW)
cap.set(3, 1000)
cap.set(4, 1000)

# *************** Creating global variables for later usage inside
functions *************** #
data = np.ndarray(shape=(1, 224, 224, 3), dtype=np.float32)
size = (224, 224)
round_prediction = None
off = False


# *************** Constantly predicts the output for the input image
and
```

```
# updates the global variable round_prediction *************** #


def constant_prediction():
    global round_prediction, off
    while True:
        # *************** Code for reading each frame, and
preprocessing the data *************** #
        success, img = cap.read()
        imgGrey = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
        imgblur = cv2.GaussianBlur(imgGrey, (3, 3), sigmaX=0,
sigmaY=0)
        imgOut = cv2.flip(imgblur, 1)
        imgOut = imgOut[300:850, 850:2700]
        imgOut = cv2.filter2D(imgOut, -1, kernel)
        imgOut = cv2.resize(imgOut, (224, 224))
        cv2.imshow("Data", imgOut)
        # *************** Code for prediction *************** #
        image = Image.fromarray(imgOut).convert("RGB")
        image = ImageOps.fit(image, size, Image.ANTIALIAS)
        image_array = np.asarray(image)
        normalized_image_array = (image_array.astype(np.float32) /
127.0) - 1
        data[0] = normalized_image_array
        prediction = model.predict(data)
        round_prediction = [round(i) for i in prediction[0]]
        k1 = cv2.waitKey(1)
        # *************** Captures ESC key press and ends the program
*************** #
        if k1 % 256 == 27:
            print("Escape hit")
            off = True  # Variable set to True to stop other threads
            break
    cap.release()
    cv2.destroyAllWindows()


def constant_upload():
    global round_prediction, off
    # *************** Constant upload of the prediction to firebase on
interval of 1 sec *************** #
    while not off:
        print(round_prediction)
        time.sleep(1)
        firebase.put("/Data", "Preds", str(round_prediction))
    exit()


# *************** threads for keeping active the function
constant_prediction and
# constant_upload without both interfering *************** #

t1 = threading.Thread(target=constant_prediction)
t1.start()

time.sleep(3)
```

```
t2 = threading.Thread(target=constant_upload)
t2.start()
#   ************** END OF PROGRAM *****************    #
```