Install keras, and tensorflow for CPU and GPU with

```
!pip install tensorflow
!pip install tensorflow-gpu
!pip install keras
```

Import modules

```
import tensorflow as tf
import matplotlib.pyplot as plt
import keras
import numpy as np
import pandas as pd
from google.colab import files
import cv2
import time
```

Check version using

```
print(tf.__version__)
print(keras.__version__)
print(cv2.__version__)
```

Import and unzip datasets using

```
!wget -c https://github.com/D-Bhatta/OpenCV-Masker/raw/initial-code/videos/VID_20201025_022413.mp4
```

Write Code

The idea is to replace the red pixels with background pixels for each frame of the video

First, we get the video file

```
captured_video = cv2.VideoCapture("VID_20201025_022413.mp4")
time.sleep(2)
```

For the first 100 frames, get the video background pixels

```
background = 0
for i in range(50):
  return_value, background = captured_video.read()
  if return_value == False:
    continue
```

Flip the frame array so that we get the pixels mirrored

```
background = np.flip(background, axis=1)
```

Loop over each frame of the video

- Get a video frame img
- If the end of the file is reached and no frame is returned, break
- Flip the video frame img array so that we get it mirrored
- Convert from RGB to HSV, to make it easier to segment the colors
- Set the threshold values for `mask1` and `mask2`
- Add `mask1` and `mask2` to create a `mask1` that can be used to generate the outline of the red area
- Refine the mask according to detected color—Remove noise using `Opening`
- Dilate the mask to get the complete red area
- Get the rest of the image in `mask2`
- Replace the mask area with the background
- Set the rest of the image as it is

- Add the entire image to get a frame
- Show the output

```
count = 0
fourcc = cv2.VideoWriter_fourcc(*'MP4V')
out = cv2.VideoWriter("output_video2.mp4", fourcc, 20.0, (1920,1080))
while captured_video.isOpened():
  return_val, img = captured_video.read()
  if not return_val:
    break
  count += 1
  img = np.flip(img, axis=1)

  hsv = cv2.cvtColor(img, cv2.COLOR_BGR2HSV)

  # Creating initial masks
  lower_blue = np.array([75,125,20])
  upper_blue = np.array([95,255,255])
  mask1 = cv2.inRange(hsv, lower_blue, upper_blue)

  lower_blue = np.array([95,125,20])
  upper_blue = np.array([130,255,255])
  mask2 = cv2.inRange(hsv, lower_blue, upper_blue)

  mask1 = mask1 + mask2

  mask1 = cv2.morphologyEx(mask1, cv2.MORPH_OPEN, np.ones((3,3), np.uint8), iterations=1)
  mask1 = cv2.dilate(mask1, np.ones((3,3), np.uint8), iterations=1)

  mask2 = cv2.bitwise_not(mask1)

  # Generate frame

  res1 = cv2.bitwise_and(background, background, mask=mask1)
  res2 = cv2.bitwise_and(img,img, mask=mask2)
  final_output = cv2.addWeighted(res1, 1, res2, 1, 0)

  out.write(final_output)
  print(f"written frame {count}")

captured_video.release()
out.release()
cv2.destroyAllWindows()
```