In [2]:

```
import cv2
import numpy as np
import os
from sklearn.neighbors import KNeighborsClassifier
from sklearn.preprocessing import LabelEncoder
```

In [3]:

```
video_file1 = "C:\\Users\\Dummy\\Desktop\\IMG_8916.MOV"
video_file2 = "C:\\Users\\Dummy\\Desktop\\IMG_8917.MOV"
cap1 = cv2.VideoCapture(video_file1)
cap2 = cv2.VideoCapture(video_file2)


face_cascade = cv2.CascadeClassifier("C:\\Users\\Dummy\\Desktop\\haarcascade_frontalface
_alt.xml")

if face_cascade.empty():
    print("Error: Unable to load the cascade classifier.")
else:

    while True:
        ret1, frame1 = cap1.read()
        ret2, frame2 = cap2.read()

        if not ret1 and not ret2:
            break

        if ret1:
            faces1 = face_cascade.detectMultiScale(frame1, 1.3, 5)
            for (x, y, w, h) in faces1:
                cv2.rectangle(frame1, (x, y), (x + w, y + h), (0, 255, 0), 3)
            cv2.imshow("Video 1 frame", frame1)

        if ret2:
            faces2 = face_cascade.detectMultiScale(frame2, 1.3, 5)
            for (x, y, w, h) in faces2:
                cv2.rectangle(frame2, (x, y), (x + w, y + h), (0, 255, 0), 3)
            cv2.imshow("Video 2 frame", frame2)


        key_pressed = cv2.waitKey(1) & 0xFF
        if key_pressed == ord('q'):
            break

    cap1.release()
    cap2.release()
    cv2.destroyAllWindows()
```

In [4]:

```
def extract_features_from_video(video_file):
    num_frames = 300
    face_data = []


    cap = cv2.VideoCapture(video_file)

    face_cascade = cv2.CascadeClassifier("C:\\Users\\Dummy\\Desktop\\haarcascade_frontal
face_alt.xml")

    while num_frames > 0:
        ret, frame = cap.read()

        if not ret:
            break
```

```
        faces = face_cascade.detectMultiScale(frame, 1.3, 5)
        faces = sorted(faces, key=lambda x: x[2] * x[3], reverse=True)
        faces = faces[:1]

        for (x, y, w, h) in faces:
            face_selection = frame[y:y + h, x:x + w]
            face_selection = cv2.resize(face_selection, (100, 100))
            face_data.append(face_selection)

            print(f"Face shape: {face_selection.shape}")

            num_frames -= 1

    face_data = np.array(face_data)
    return face_data

video_file1 = "C:\\Users\\Dummy\\Desktop\\IMG_8916.MOV"
video_file2 = "C:\\Users\\Dummy\\Desktop\\IMG_8917.MOV"

for video_file in [video_file1, video_file2]:
    name = input(f"Enter your name for video '{video_file}': ")
    features = extract_features_from_video(video_file)
    np.save(name, features)

cv2.destroyAllWindows()
```

```
Enter your name for video 'C:\Users\Dummy\Desktop\IMG_8916.MOV': Anas
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
```

```
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Enter your name for video 'C:\Users\Dummy\Desktop\IMG_8917.MOV': Hamza
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
```

```
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
```

```
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
Face shape: (100, 100, 3)
```

In [5]:

```
import os
import numpy as np
from sklearn.preprocessing import LabelEncoder

directory_path = 'C:\\Users\\Dummy\\Desktop\\NPY'
files = [file for file in os.listdir(directory_path) if file.endswith('.npy')]
names = [file[:-4] for file in files]

label_encoder = LabelEncoder()

names_encoded = label_encoder.fit_transform(names)

print(names_encoded)

face_data = []

for filename in files:
    data = np.load(os.path.join(directory_path, filename))
    print(data.shape)
    face_data.extend([data] * 300)

face_data = np.array(face_data)
face_data = face_data.reshape((-1, face_data.shape[-1]))
print(face_data.shape)

names_encoded = np.repeat(names_encoded, 300)

names_encoded = names_encoded[:len(face_data)]

dataset = np.hstack((face_data, names_encoded[:, np.newaxis]))
print(dataset.shape)
```

```
[0 1]
(106, 100, 100, 3)
(85, 100, 100, 3)
(1, 600)
(1, 601)
```

```
C:\Users\Dummy\AppData\Local\Temp\ipykernel_20848\1486791722.py:26: VisibleDeprecationWar
ning: Creating an ndarray from ragged nested sequences (which is a list-or-tuple of lists
-or-tuples-or ndarrays with different lengths or shapes) is deprecated. If you meant to d
o this, you must specify 'dtype=object' when creating the ndarray.
  face_data = np.array(face_data)
```

In [7]:

```
import numpy as np
import os
from sklearn.neighbors import KNeighborsClassifier
from sklearn.preprocessing import LabelEncoder

directory_path = 'C:\\Users\\Dummy\\Desktop\\NPY'
files = [file for file in os.listdir(directory_path) if file.endswith('.npy')]
names = [file[:-4] for file in files]

label_encoder = LabelEncoder()

label_encoder.fit(names)

face_data_list = []
names_encoded_list = []

for filename in files:
    data = np.load(os.path.join(directory_path, filename))
    num_samples = data.shape[0]
```

```
    data = data.reshape((num_samples, -1))

    face_data_list.append(data)


    names_encoded = label_encoder.transform([filename[:-4]] * num_samples)
    names_encoded_list.append(names_encoded)

face_data = np.vstack(face_data_list)
names_encoded = np.hstack(names_encoded_list)

dataset = np.column_stack((face_data, names_encoded))

face_pred = KNeighborsClassifier()
face_pred.fit(dataset[:, :-1], dataset[:, -1])
```

Out[7]:

▾ KNeighborsClassifier

KNeighborsClassifier()

In [ ]:

```
cap=cv2.VideoCapture('C:\\Users\\Dummy\\Desktop\\IMG_8918.MOV')
face_cascade=cv2.CascadeClassifier('C:\\Users\\Dummy\\Desktop\\haarcascade_frontalface_alt.xml')

while True:
    ret , frame = cap.read()
    if not ret:
        continue
    # Find All the faces in the frame
    faces = face_cascade.detectMultiScale(frame , 1.3 ,5)


    print(faces)

    for face in faces:
        x,y,w,h = face
        face_selection = frame[y:y+h , x:x+w]
        print(face_selection.shape)

        face_selection = cv2.resize(face_selection,(100,100))
        print(face_selection.shape)
        face_cropped = face_selection.reshape((1,-1))
        print(face_cropped.shape)


        pred = face_pred.predict(face_cropped)
        pred = label_encoder.inverse_transform(pred)

        cv2.rectangle(frame,(x,y),(x+w,y+h),(0,255,0),5)
        cv2.putText(frame, pred[0], (x, y), cv2.FONT_HERSHEY_SIMPLEX, 1, (255, 255, 255), 2)
    cv2.imshow("Feed" , frame)
    key = cv2.waitKey(1)
    if key & 0xFF == ord('q'):
        break

cap.release()
cv2.destroyAllWindows()
```

```
[[745 171 307 307]]
(307, 307, 3)
(100, 100, 3)
(1, 30000)
[[754 180 291 291]]
(291, 291, 3)
(100, 100, 3)
(1, 30000)
[[457  93 233 233]
```

```
 [751 177 296 296]]
(233, 233, 3)
(100, 100, 3)
(1, 30000)
(296, 296, 3)
(100, 100, 3)
(1, 30000)
[[450  90 244 244]
 [750 176 301 301]]
(244, 244, 3)
(100, 100, 3)
(1, 30000)
(301, 301, 3)
(100, 100, 3)
(1, 30000)
[[450  90 244 244]
 [748 176 305 305]]
(244, 244, 3)
(100, 100, 3)
(1, 30000)
(305, 305, 3)
(100, 100, 3)
(1, 30000)
[[447  87 249 249]
 [753 175 305 305]]
(249, 249, 3)
(100, 100, 3)
(1, 30000)
(305, 305, 3)
(100, 100, 3)
(1, 30000)
[[456  96 233 233]
 [751 175 305 305]]
(233, 233, 3)
(100, 100, 3)
(1, 30000)
(305, 305, 3)
(100, 100, 3)
(1, 30000)
[[461  94 230 230]
 [758 178 294 294]]
(230, 230, 3)
(100, 100, 3)
(1, 30000)
(294, 294, 3)
(100, 100, 3)
(1, 30000)
[[750 175 301 301]
 [458  93 228 228]]
(301, 301, 3)
(100, 100, 3)
(1, 30000)
(228, 228, 3)
(100, 100, 3)
(1, 30000)
[[463  96 221 221]
 [756 175 300 300]]
(221, 221, 3)
(100, 100, 3)
(1, 30000)
(300, 300, 3)
(100, 100, 3)
(1, 30000)
[[464  96 223 223]
 [756 177 303 303]]
(223, 223, 3)
(100, 100, 3)
(1, 30000)
(303, 303, 3)
(100, 100, 3)
(1, 30000)
[[465  99 221 221]
```

```
 [751 176 308 308]]
(221, 221, 3)
(100, 100, 3)
(1, 30000)
(308, 308, 3)
(100, 100, 3)
(1, 30000)
[[750 178 308 308]]
(308, 308, 3)
(100, 100, 3)
(1, 30000)
[[752 175 303 303]]
(303, 303, 3)
(100, 100, 3)
(1, 30000)
[[753 175 303 303]]
(303, 303, 3)
(100, 100, 3)
(1, 30000)
[[754 177 300 300]]
(300, 300, 3)
(100, 100, 3)
(1, 30000)
[[764 177 289 289]]
(289, 289, 3)
(100, 100, 3)
(1, 30000)
[[756 178 299 299]]
(299, 299, 3)
(100, 100, 3)
(1, 30000)
[[750 175 305 305]]
(305, 305, 3)
(100, 100, 3)
(1, 30000)
[[472  96 221 221]
 [745 175 308 308]]
(221, 221, 3)
(100, 100, 3)
(1, 30000)
(308, 308, 3)
(100, 100, 3)
(1, 30000)
[[753 180 298 298]]
(298, 298, 3)
(100, 100, 3)
(1, 30000)
[[473  96 223 223]
 [752 185 296 296]]
(223, 223, 3)
(100, 100, 3)
(1, 30000)
(296, 296, 3)
(100, 100, 3)
(1, 30000)
[[750 183 301 301]]
(301, 301, 3)
(100, 100, 3)
(1, 30000)
[[747 178 303 303]]
(303, 303, 3)
(100, 100, 3)
(1, 30000)
[[748 176 303 303]]
(303, 303, 3)
(100, 100, 3)
(1, 30000)
[[477 100 212 212]
 [745 177 303 303]]
(212, 212, 3)
(100, 100, 3)
(1, 30000)
```

```
(303, 303, 3)
(100, 100, 3)
(1, 30000)
[[477 100 212 212]
 [744 178 301 301]]
(212, 212, 3)
(100, 100, 3)
(1, 30000)
(301, 301, 3)
(100, 100, 3)
(1, 30000)
[[746 179 303 303]]
(303, 303, 3)
(100, 100, 3)
(1, 30000)
[[746 181 299 299]]
(299, 299, 3)
(100, 100, 3)
(1, 30000)
[[740 180 295 295]]
(295, 295, 3)
(100, 100, 3)
(1, 30000)
[[470  95 212 212]
 [737 179 296 296]]
(212, 212, 3)
(100, 100, 3)
(1, 30000)
(296, 296, 3)
(100, 100, 3)
(1, 30000)
[[473  95 205 205]
 [729 177 303 303]]
(205, 205, 3)
(100, 100, 3)
(1, 30000)
(303, 303, 3)
(100, 100, 3)
(1, 30000)
[[470  95 212 212]
 [731 180 296 296]]
(212, 212, 3)
(100, 100, 3)
(1, 30000)
(296, 296, 3)
(100, 100, 3)
(1, 30000)
[[470  94 212 212]
 [725 175 305 305]]
(212, 212, 3)
(100, 100, 3)
(1, 30000)
(305, 305, 3)
(100, 100, 3)
(1, 30000)
[[472  95 212 212]
 [715 181 296 296]]
(212, 212, 3)
(100, 100, 3)
(1, 30000)
(296, 296, 3)
(100, 100, 3)
(1, 30000)
[[469  92 219 219]
 [720 185 285 285]]
(219, 219, 3)
(100, 100, 3)
(1, 30000)
(285, 285, 3)
(100, 100, 3)
(1, 30000)
[[470  87 223 223]
```

```
 [706 181 296 296]]
(223, 223, 3)
(100, 100, 3)
(1, 30000)
(296, 296, 3)
(100, 100, 3)
(1, 30000)
[[704 175 303 303]]
(303, 303, 3)
(100, 100, 3)
(1, 30000)
[[695 176 302 302]
 [472  94 204 204]]
(302, 302, 3)
(100, 100, 3)
(1, 30000)
(204, 204, 3)
(100, 100, 3)
(1, 30000)
[[468  87 221 221]
 [701 178 290 290]]
(221, 221, 3)
(100, 100, 3)
(1, 30000)
(290, 290, 3)
(100, 100, 3)
(1, 30000)
[[466  87 221 221]
 [696 175 298 298]]
(221, 221, 3)
(100, 100, 3)
(1, 30000)
(298, 298, 3)
(100, 100, 3)
(1, 30000)
[[705 179 280 280]
 [471  85 223 223]]
(280, 280, 3)
(100, 100, 3)
(1, 30000)
(223, 223, 3)
(100, 100, 3)
(1, 30000)
[[699 175 284 284]
 [468  84 221 221]]
(284, 284, 3)
(100, 100, 3)
(1, 30000)
(221, 221, 3)
(100, 100, 3)
(1, 30000)
[[467  86 210 210]
 [702 173 285 285]]
(210, 210, 3)
(100, 100, 3)
(1, 30000)
(285, 285, 3)
(100, 100, 3)
(1, 30000)
[[706 175 279 279]
 [456  78 233 233]]
(279, 279, 3)
(100, 100, 3)
(1, 30000)
(233, 233, 3)
(100, 100, 3)
(1, 30000)
[[456  78 233 233]]
(233, 233, 3)
(100, 100, 3)
(1, 30000)
[[690 167 303 303]]
```

```
(303, 303, 3)
(100, 100, 3)
(1, 30000)
[[462  83 214 214]
 [707 177 276 276]]
(214, 214, 3)
(100, 100, 3)
(1, 30000)
(276, 276, 3)
(100, 100, 3)
(1, 30000)
[[467  84 204 204]
 [716 175 290 290]]
(204, 204, 3)
(100, 100, 3)
(1, 30000)
(290, 290, 3)
(100, 100, 3)
(1, 30000)
[[719 167 306 306]]
(306, 306, 3)
(100, 100, 3)
(1, 30000)
[[736 169 298 298]]
(298, 298, 3)
(100, 100, 3)
(1, 30000)
[[444  80 212 212]
 [747 172 288 288]]
(212, 212, 3)
(100, 100, 3)
(1, 30000)
(288, 288, 3)
(100, 100, 3)
(1, 30000)
[[748 163 312 312]]
(312, 312, 3)
(100, 100, 3)
(1, 30000)
[[763 169 296 296]]
(296, 296, 3)
(100, 100, 3)
(1, 30000)
[[772 167 298 298]]
(298, 298, 3)
(100, 100, 3)
(1, 30000)
[[776 165 301 301]]
(301, 301, 3)
(100, 100, 3)
(1, 30000)
[[783 162 306 306]]
(306, 306, 3)
(100, 100, 3)
(1, 30000)
[[784 161 317 317]]
(317, 317, 3)
(100, 100, 3)
(1, 30000)
[[783 154 322 322]]
(322, 322, 3)
(100, 100, 3)
(1, 30000)
[[799 164 303 303]]
(303, 303, 3)
(100, 100, 3)
(1, 30000)
[[799 160 311 311]]
(311, 311, 3)
(100, 100, 3)
(1, 30000)
[[791 161 324 324]]
```

```
(324, 324, 3)
(100, 100, 3)
(1, 30000)
[[793 162 317 317]]
(317, 317, 3)
(100, 100, 3)
(1, 30000)
[[811 163 313 313]]
(313, 313, 3)
(100, 100, 3)
(1, 30000)
[[811 161 317 317]]
(317, 317, 3)
(100, 100, 3)
(1, 30000)
[[813 160 317 317]]
(317, 317, 3)
(100, 100, 3)
(1, 30000)
[[811 164 317 317]]
(317, 317, 3)
(100, 100, 3)
(1, 30000)
[[805 158 327 327]]
(327, 327, 3)
(100, 100, 3)
(1, 30000)
[[814 162 317 317]]
(317, 317, 3)
(100, 100, 3)
(1, 30000)
[[805 155 331 331]]
(331, 331, 3)
(100, 100, 3)
(1, 30000)
[[805 165 317 317]]
(317, 317, 3)
(100, 100, 3)
(1, 30000)
[[804 161 322 322]]
(322, 322, 3)
(100, 100, 3)
(1, 30000)
[[796 163 331 331]]
(331, 331, 3)
(100, 100, 3)
(1, 30000)
()
[[797 167 317 317]]
(317, 317, 3)
(100, 100, 3)
(1, 30000)
[[806 186 294 294]]
(294, 294, 3)
(100, 100, 3)
(1, 30000)
[[795 180 310 310]]
(310, 310, 3)
(100, 100, 3)
(1, 30000)
[[782 172 323 323]]
(323, 323, 3)
(100, 100, 3)
(1, 30000)
[[783 172 317 317]]
(317, 317, 3)
(100, 100, 3)
(1, 30000)
[[785 180 301 301]]
(301, 301, 3)
(100, 100, 3)
(1, 30000)
```

```
[[779 177 312 312]]
(312, 312, 3)
(100, 100, 3)
(1, 30000)
[[789 182 303 303]]
(303, 303, 3)
(100, 100, 3)
(1, 30000)
[[787 185 292 292]]
(292, 292, 3)
(100, 100, 3)
(1, 30000)
[[789 181 298 298]]
(298, 298, 3)
(100, 100, 3)
(1, 30000)
[[777 173 309 309]]
(309, 309, 3)
(100, 100, 3)
(1, 30000)
()
[[774 165 322 322]]
(322, 322, 3)
(100, 100, 3)
(1, 30000)
[[795 179 299 299]]
(299, 299, 3)
(100, 100, 3)
(1, 30000)
[[789 174 309 309]]
(309, 309, 3)
(100, 100, 3)
(1, 30000)
[[794 172 303 303]]
(303, 303, 3)
(100, 100, 3)
(1, 30000)
[[793 174 306 306]]
(306, 306, 3)
(100, 100, 3)
(1, 30000)
[[785 164 313 313]]
(313, 313, 3)
(100, 100, 3)
(1, 30000)
[[787 169 312 312]]
(312, 312, 3)
(100, 100, 3)
(1, 30000)
[[784 164 317 317]]
(317, 317, 3)
(100, 100, 3)
(1, 30000)
[[786 170 309 309]]
(309, 309, 3)
(100, 100, 3)
(1, 30000)
[[791 170 309 309]]
(309, 309, 3)
(100, 100, 3)
(1, 30000)
[[791 170 309 309]]
(309, 309, 3)
(100, 100, 3)
(1, 30000)
[[790 170 306 306]]
(306, 306, 3)
(100, 100, 3)
(1, 30000)
[[784 169 312 312]]
(312, 312, 3)
(100, 100, 3)
```

```
(1, 30000)
[[791 167 312 312]]
(312, 312, 3)
(100, 100, 3)
(1, 30000)
[[783 163 331 331]]
(331, 331, 3)
(100, 100, 3)
(1, 30000)
[[785 165 323 323]]
(323, 323, 3)
(100, 100, 3)
(1, 30000)
[[794 171 307 307]]
(307, 307, 3)
(100, 100, 3)
(1, 30000)
[[789 171 317 317]]
(317, 317, 3)
(100, 100, 3)
(1, 30000)
[[789 172 312 312]]
(312, 312, 3)
(100, 100, 3)
(1, 30000)
[[790 174 307 307]]
(307, 307, 3)
(100, 100, 3)
(1, 30000)
[[781 172 309 309]]
(309, 309, 3)
(100, 100, 3)
(1, 30000)
[[779 177 301 301]]
(301, 301, 3)
(100, 100, 3)
(1, 30000)
[[764 169 321 321]]
(321, 321, 3)
(100, 100, 3)
(1, 30000)
[[769 176 310 310]]
(310, 310, 3)
(100, 100, 3)
(1, 30000)
[[768 177 309 309]]
(309, 309, 3)
(100, 100, 3)
(1, 30000)
[[778 180 298 298]]
(298, 298, 3)
(100, 100, 3)
(1, 30000)
[[775 182 296 296]]
(296, 296, 3)
(100, 100, 3)
(1, 30000)
[[777 180 303 303]]
(303, 303, 3)
(100, 100, 3)
(1, 30000)
[[776 180 307 307]]
(307, 307, 3)
(100, 100, 3)
(1, 30000)
[[772 180 310 310]]
(310, 310, 3)
(100, 100, 3)
(1, 30000)
[[768 179 310 310]]
(310, 310, 3)
(100, 100, 3)
```

```
(1, 30000)
[[764 176 317 317]]
(317, 317, 3)
(100, 100, 3)
(1, 30000)
[[771 182 307 307]]
(307, 307, 3)
(100, 100, 3)
(1, 30000)
[[779 188 298 298]]
(298, 298, 3)
(100, 100, 3)
(1, 30000)
[[770 183 309 309]]
(309, 309, 3)
(100, 100, 3)
(1, 30000)
[[781 188 296 296]]
(296, 296, 3)
(100, 100, 3)
(1, 30000)
[[773 189 303 303]]
(303, 303, 3)
(100, 100, 3)
(1, 30000)
[[773 189 303 303]]
(303, 303, 3)
(100, 100, 3)
(1, 30000)
[[762 180 312 312]]
(312, 312, 3)
(100, 100, 3)
(1, 30000)
[[765 180 312 312]]
(312, 312, 3)
(100, 100, 3)
(1, 30000)
[[769 178 312 312]]
(312, 312, 3)
(100, 100, 3)
(1, 30000)
[[766 178 317 317]]
(317, 317, 3)
(100, 100, 3)
(1, 30000)
[[778 185 295 295]]
(295, 295, 3)
(100, 100, 3)
(1, 30000)
[[776 184 303 303]]
(303, 303, 3)
(100, 100, 3)
(1, 30000)
[[766 181 314 314]]
(314, 314, 3)
(100, 100, 3)
(1, 30000)
[[770 182 307 307]]
(307, 307, 3)
(100, 100, 3)
(1, 30000)
[[769 178 313 313]]
(313, 313, 3)
(100, 100, 3)
(1, 30000)
[[776 184 303 303]]
(303, 303, 3)
(100, 100, 3)
(1, 30000)
[[774 185 303 303]]
(303, 303, 3)
(100, 100, 3)
```

```
(1, 30000)
[[770 183 303 303]]
(303, 303, 3)
(100, 100, 3)
(1, 30000)
[[773 185 301 301]]
(301, 301, 3)
(100, 100, 3)
(1, 30000)
[[770 184 303 303]]
(303, 303, 3)
(100, 100, 3)
(1, 30000)
[[772 185 306 306]]
(306, 306, 3)
(100, 100, 3)
(1, 30000)
[[771 185 308 308]]
(308, 308, 3)
(100, 100, 3)
(1, 30000)
[[771 185 308 308]
 [476  86 212 212]]
(308, 308, 3)
(100, 100, 3)
(1, 30000)
(212, 212, 3)
(100, 100, 3)
(1, 30000)
[[770 188 306 306]
 [473  86 212 212]]
(306, 306, 3)
(100, 100, 3)
(1, 30000)
(212, 212, 3)
(100, 100, 3)
(1, 30000)
[[766 181 310 310]]
(310, 310, 3)
(100, 100, 3)
(1, 30000)
[[771 189 303 303]]
(303, 303, 3)
(100, 100, 3)
(1, 30000)
```

In [ ]: