Graded Quiz: Test your Project Understanding
Graded Quiz • 30 min

Due Aug 10, 12:29 PM IST

Congratulations! You passed!
TO PASS 80% or higher

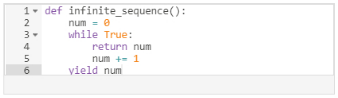
Keep Learning

GRADE
100%

Graded Quiz: Test your Project Understanding

LATEST SUBMISSION GRADE
100%

1. Which line of code is incorrect, for this Python generator: **1 / 1 point**

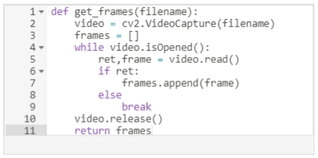


```
1  def infinite_sequence():
2      num = 0
3      while True:
4          return num
5          num += 1
6      yield num
```

4

✓ **Correct**
Yes, we must use yield instead of return.

2. The main problem with the following piece of code is: **1 / 1 point**



```
def get_frames(filename):
    video = cv2.VideoCapture(filename)
    frames = []
    while video.isOpened():
        ret,frame = video.read()
        if ret:
            frames.append(frame)
        else
            break
    video.release()
    return frames
```

[✓ Too much memory space required]
Not a Python generator
Too slow

✓ **Correct**
Correct, this function reads the entire video into memory before returning everything.

3. To represent a color video frame using a numpy array, how many dimensions should the array have? **1 / 1 point**

3

✓ **Correct**
We must store the location of the pixel (two dimensions), plus another dimension for the Red, Green and Blue color intensities.

4. The reason we can't read a video frame from our movie file without reading the preceding ones first is: **1 / 1 point**

Frame rate is not available in video header.
A Python generator starts iterating from the first item.
[✓ Video compression.]

✓ **Correct**
Yes, after compressing the video, information in one frame depends on the previous ones, so we cannot just jump to a specific frame and ignore the preceeding.

5. The color conventions we saw in this project were: **1 / 1 point**

matplotlib uses BGR, OpenCV uses RGB
[✓ OpenCV uses BGR, matplotlib uses RGB]

✓ **Correct**
That is the correct convention.

6. If we were to set all the pixel values to 255, our video would be (choose the best): **1 / 1 point**

All Black
[✓ All White]
Brighter

○ Darker

✓ **Correct**
The color values for each pixel range from 0 to 255, with 255 as the brightest value. So setting the three color channels to 255 gives pure white.

7. Select the correct syntax for drawing a circle on the video frame: **1 / 1 point**

- [✓ cv2.circle(frame, center=(200,200), radius=50, color=(0,0,255), thickness=10)]
- frame = cv2.circle(frame, center=(200,200), radius=50, color=(0,0,255), thickness=10)
- Neither of those

✓ **Correct**
That is the correct syntax.

8. Select the command that is not related to writing a video to file: **1 / 1 point**

- [✓ video = cv2.VideoCapture(VFILE)]
- video_out = cv2.VideoWriter("new.mp4", fourcc, 20, (640,480))
- fourcc = cv2.VideoWriter_fourcc('M', 'P', '4', 'V')
- video_out.write(frame)

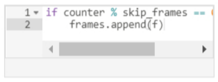
✓ **Correct**
Correct, this instantiates a VideoCapture object which is useful for reading video, not writing video.

9. To stack images vertically, we call np.concatenate() with the axis argument set to: **1 / 1 point**

- 1
- [✓ 0]

✓ **Correct**
Correct, axis=0 will result in vertically stacked images.

10. The purpose of this piece of code is: **1 / 1 point**



```
if counter % skip_frames ==
    frames.append(f)
```

- [✓ To collect a set of video frames that are spaced at equal intervals throughout the video file.]
- To collect all the video frames after skipping the first skip_frames.
- To skip directly to the correct frame index so a single video frame can be read.

✓ **Correct**
Correct, the modulo operator will give a zero result every time we've skipped the correct number of frames.