LINE BY LINE EXPLANATION

- Line 5: The cv2.VideoCapture() function is used to open the default camera (camera index 0) and store the resulting VideoCapture object in the variable 'cap'.

- Line 8: The cv2.CascadeClassifier() function is used to load a pre-trained Haar cascade classifier from the file "haarcascade_russian_plate_number.xml" and store the resulting CascadeClassifier object in the variable 'licence_cascade'. Haar cascades are a type of object detection algorithm that uses a combination of simple features like edges and lines to detect objects in images.

- Lines 13-15: A while loop is used to continuously capture video frames and detect license plates in them. The read() method of the 'cap' object is used to capture a single frame, which is stored in the 'img' variable. The read() method returns a tuple containing a boolean value indicating whether the frame was read successfully, and the frame itself. The underscore (_) variable is used to store the boolean value and not used later.

- Line 17: The cv2.cvtColor() function is used to convert the captured frame from BGR to grayscale, as the Haar cascade classifier expects grayscale images. The grayscale image is stored in the 'gray' variable.

- Line 19-20: The detectMultiScale() method of the classifier is used to detect license plates in the grayscale frame. The method takes the grayscale frame, a scaling factor (2.3), and a minimum number of neighbors (4) as inputs. The scaling factor controls the size of the window used to scan the image, and a smaller factor means a larger window and a more thorough scan, but also a slower processing time. The minimum number of neighbors is used to eliminate false positives, and a higher value means more selective detection. The method returns a list of rectangles, where each rectangle represents a detected license plate.

- Line 22-27: A for loop is used to iterate over the list of rectangles returned by the detectMultiScale() method. For each rectangle, a green rectangle is drawn around the detected license plate using the cv2.rectangle() function (Line 23). The function takes the image, the top-left corner coordinates of the rectangle, the bottom-right corner coordinates of the rectangle, the color of the rectangle (green), and the thickness of the rectangle (2 pixels) as inputs. A label "LICENCE PLATE" is also added to the top-left corner of the rectangle using the cv2.putText() function (Line 24). The function takes the image, the text to be added, the position of the text, the font of the text, the font scale, the color of the text, and the thickness of the text as inputs.

- Line 25-26: ROI is extracted from the original image using the coordinates of the rectangle and stored in 'imroi' variable. Then 'Canny' is applied on the ROI. Canny edge detection algorithm is used to detect edges of an image.

- Line 28: The modified frame with rectangles and labels is displayed using the cv2.imshow() function. The function takes the name of the window and the image to be displayed as inputs.

- Line 29-34: If the user presses the 'q' key, the current frame is saved to the file "q.jpg" using the cv2.imwrite() function. A blue rectangle and a message "SCAN S