Number Plate Detection using OpenCV

> Number plate detection using OpenCV involves using computer vision techni
> ate and extract the
> license plate region from an image or video frame.

In [1]:
```
import cv2
import imutils #We will need this library to resize our images.
import pytesseract #We will need this library to extract the license plate te
```

In [2]:
```
pytesseract.pytesseract.tesseract_cmd = r'C:\Users\LENOVO T480\Downloads\tess
```

In [3]:
```
image = cv2.imread(r'C:\Users\LENOVO T480\Downloads\car_images\pic_6.jpg')
resized_image = imutils.resize(image)
cv2.imshow('original image', image)
cv2.waitKey(0)
```

```
-1
```

In [4]:
```
gray_image = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
cv2.imshow("greyed image", gray_image)
cv2.waitKey(0)
```

```
-1
```

In [5]:
```
gray_image = cv2.bilateralFilter(gray_image, 11, 17, 17)
cv2.imshow("smoothened image", gray_image)
cv2.waitKey(0)
```

```
-1
```

In [6]:
```
edged = cv2.Canny(gray_image, 30, 200)
cv2.imshow("edged image", edged)
cv2.waitKey(0)
```

```
-1
```

In [7]:
```
cnts,new = cv2.findContours(edged.copy(), cv2.RETR_LIST, cv2.CHAIN_APPROX_SIM
image1=image.copy()
cv2.drawContours(image1,cnts,-1,(0,255,0),3)
cv2.imshow("contours",image1)
cv2.waitKey(0)
```

```
-1
```

In [8]:
```
cnts = sorted(cnts, key = cv2.contourArea, reverse = True) [:30]
screenCnt = None
image2 = image.copy()
cv2.drawContours(image2,cnts,-1,(0,255,0),3)
cv2.imshow("Top 30 contours",image2)
cv2.waitKey(0)
```

```
-1
```

In [9]:
```
i=7
for c in cnts:
        perimeter = cv2.arcLength(c, True)
        approx = cv2.approxPolyDP(c, 0.018 * perimeter, True)
        if len(approx) == 4:
                screenCnt = approx
                x,y,w,h = cv2.boundingRect(c)
                new_img=image[y:y+h,x:x+w]
                cv2.imwrite('./'+str(i)+'.png',new_img)
                i+=1
                break
```

In [10]:
```
cv2.drawContours(image, [screenCnt], -1, (0, 255, 0), 3)
cv2.imshow("image with detected license plate", image)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

In [ ]:

In [ ]:

In [ ]: