OCR_test

March 4, 2023

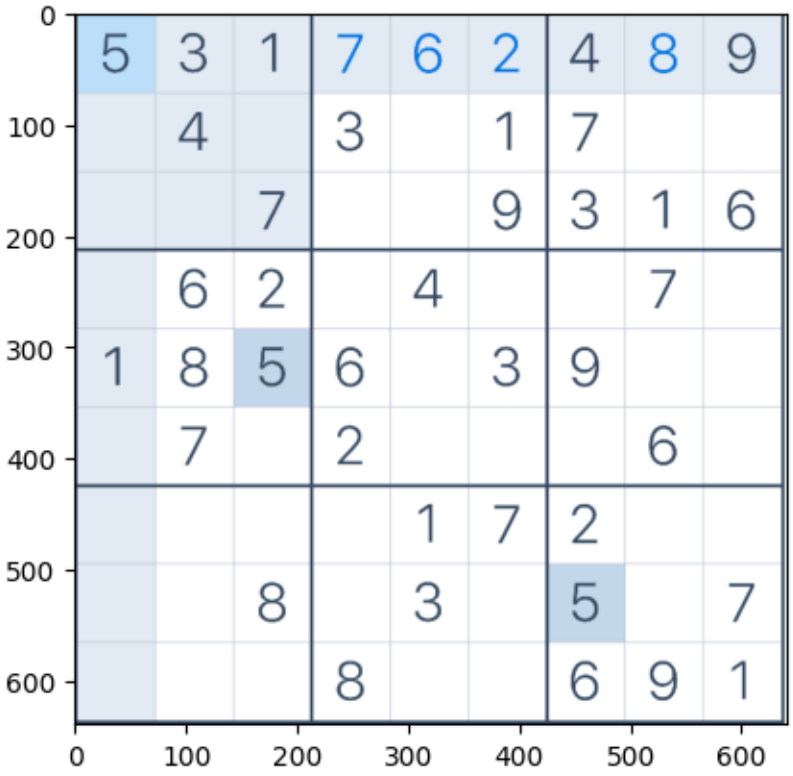
[ ]:
```
import matplotlib.pyplot as plt
import pytesseract
import numpy as np
from PIL import Image
import cv2

pytesseract.pytesseract.tesseract_cmd = r'C:
 ↪\Users\wilko\AppData\Local\Programs\Tesseract-OCR\tesseract.exe'
```

[ ]:
```
test_img = cv2.imread(r'C:\Users\wilko\Desktop\Studia\Projekty␣
 ↪studia\Optymalizacja dyskretna - Sudoku\Dane\Pierdoly\ocr_test.PNG')
```

[ ]:
```
test_img = cv2.cvtColor(test_img, cv2.COLOR_BGR2RGB)
```

[ ]:
```
plt.imshow(test_img)
```

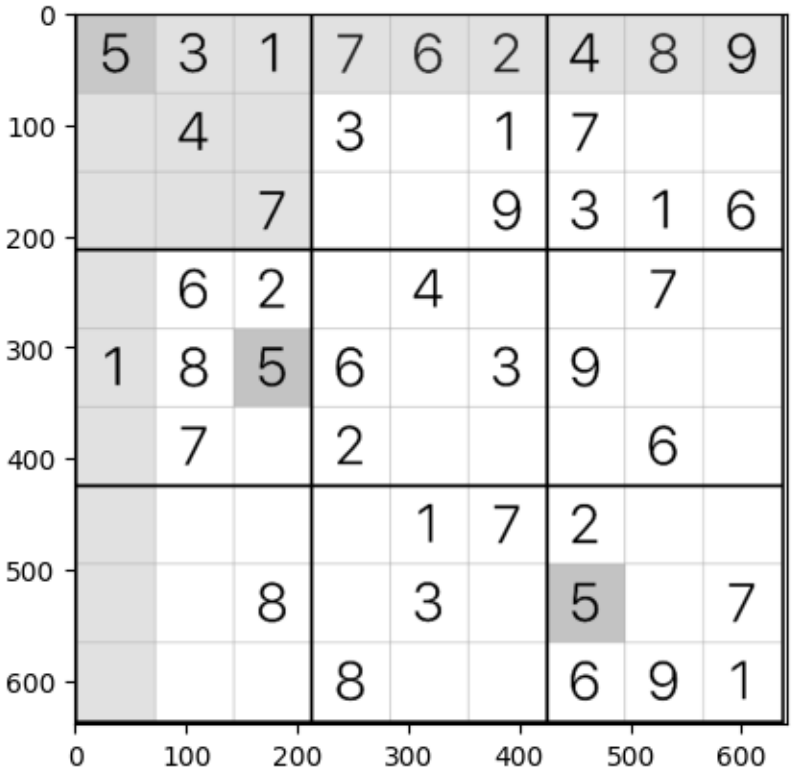
[ ]:
```
<matplotlib.image.AxesImage at 0x27b9008f940>
```



[ ]:
```
gray_img = cv2.cvtColor(test_img, cv2.COLOR_RGB2GRAY)
```

[ ]:
```
plt.imshow(gray_img, cmap='gray')
```

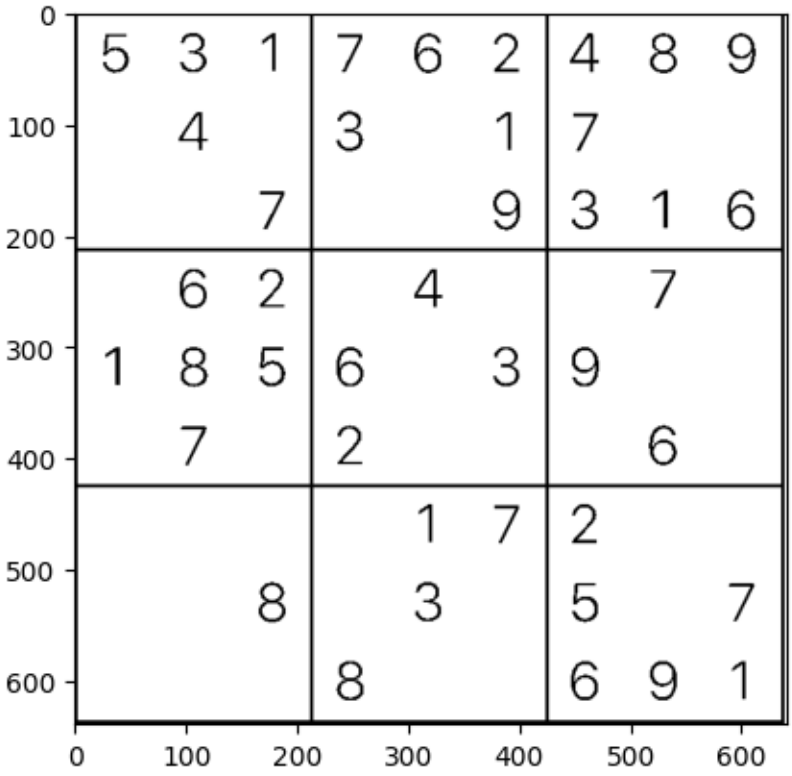
[ ]:
```
<matplotlib.image.AxesImage at 0x27b9374d0f0>
```



[ ]:
```
_,binary_img = cv2.threshold(gray_img,127,255,cv2.THRESH_BINARY)
```

[ ]:
```
plt.imshow(binary_img, cmap='gray')
```

[ ]:
```
<matplotlib.image.AxesImage at 0x27b937cb460>
```



[ ]:
```
binary_img.shape
```

[ ]: (639, 641)

[ ]:
```
binary_img.shape[0]//9
```

[ ]: 71

[ ]:
```
binary_img.shape[1]//9
```

[ ]: 71

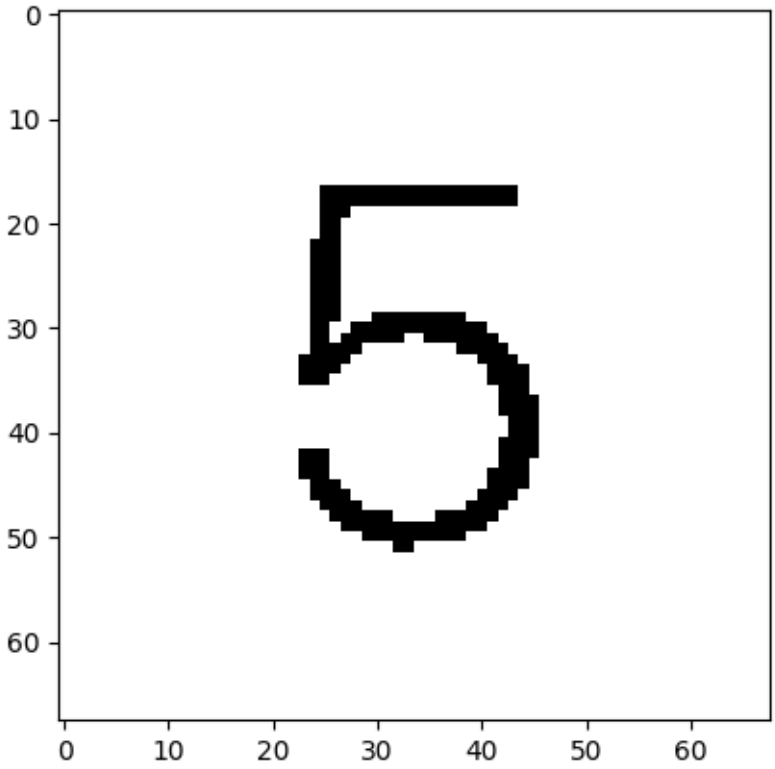
[ ]:
```
images = []

cell_size = binary_img.shape[0]//9

y_start = 0
y_end = binary_img.shape[0]//9

x_start = 0
x_end = binary_img.shape[1]//9
```

```
for i in range(1,10):
    for j in range(1,10):
        images.append(binary_img[y_start : y_end, x_start : x_end])
        x_start += cell_size
        x_end += cell_size

    x_start = 0
    x_end = cell_size

    y_start += cell_size
    y_end += cell_size
```

[ ]:
```
cell1 = binary_img[3:71, 3:71]
```

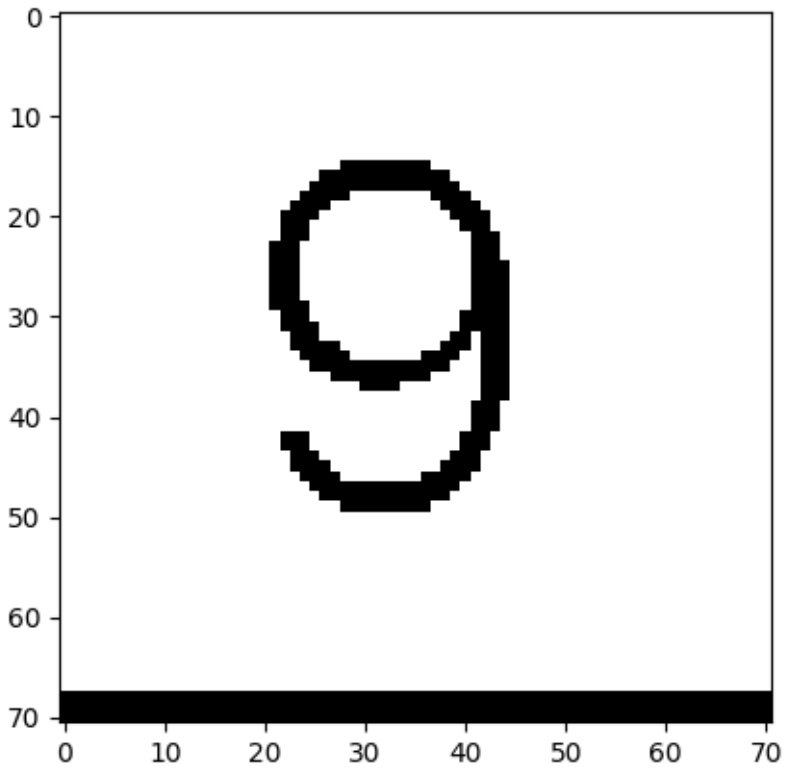
[ ]:
```
plt.imshow(cell1, cmap='gray')
```

[ ]:
```
<matplotlib.image.AxesImage at 0x27b939befb0>
```



[ ]:
```
plt.imshow(images[79],cmap='gray')
```

[ ]:
```
<matplotlib.image.AxesImage at 0x27b9bacc760>
```



[ ]:
```
ocr_test1 = pytesseract.image_to_string(
        images[79], config='--psm 13 --oem 3 -c␣
 ↪tessedit_char_whitelist=123456789')
```

[ ]:
```
ocr_test1
```

[ ]:
```
''
```

[ ]:
```
ocr_test1 = ocr_test1.replace("\n", " ")
```

[ ]:
```
int(ocr_test1)
```

[ ]:
```
9
```

[ ]:
```
sudoku_board = []
for image in images:
    ocr_result = pytesseract.image_to_string(image, config='--psm 13 --oem 3 -c␣
 ↪tessedit_char_whitelist=123456789')
    if ocr_result != '':
        sudoku_board.append(int(ocr_result))
    else:
        sudoku_board.append(0)
```

[ ]:
```
sudoku_board = np.array(sudoku_board)
sudoku_board
```

[ ]:
```
array([5, 3, 1, 7, 6, 2, 0, 8, 9, 0, 4, 0, 3, 0, 1, 7, 0, 0, 0, 0, 7, 0,
       0, 9, 3, 1, 6, 0, 6, 2, 0, 4, 0, 0, 7, 0, 1, 8, 5, 6, 0, 3, 0, 0,
       0, 0, 7, 0, 2, 0, 0, 0, 6, 0, 0, 0, 0, 0, 1, 7, 2, 0, 0, 0, 0, 8,
       0, 3, 0, 5, 0, 7, 0, 0, 0, 8, 0, 0, 6, 0, 1])
```

[ ]:
```
sudoku_board.size
```

[ ]:
```
81
```

[ ]:
```
sudoku_board.reshape((9,9))
```

[ ]:
```
array([[5, 3, 1, 7, 6, 2, 0, 8, 9],
       [0, 4, 0, 3, 0, 1, 7, 0, 0],
       [0, 0, 7, 0, 0, 9, 3, 1, 6],
       [0, 6, 2, 0, 4, 0, 0, 7, 0],
       [1, 8, 5, 6, 0, 3, 0, 0, 0],
       [0, 7, 0, 2, 0, 0, 0, 6, 0],
       [0, 0, 0, 0, 1, 7, 2, 0, 0],
       [0, 0, 8, 0, 3, 0, 5, 0, 7],
       [0, 0, 0, 8, 0, 0, 6, 0, 1]])
```

[ ]:
```

```