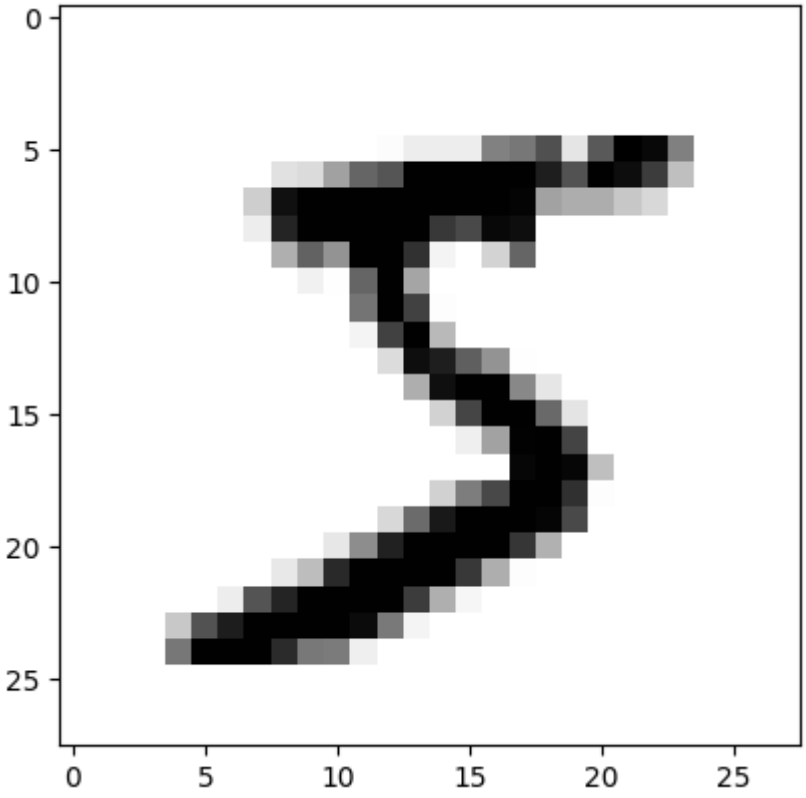
Лабораторная работа №4

Выполнил студент группы БВТ2003 Глазков Даниил

In [1]:
```
import tensorflow as tf
mnist = tf.keras.datasets.mnist
(train_images, train_labels),(test_images, test_labels) = mnist.load_data()
```

In [2]:
```
import matplotlib.pyplot as plt
plt.imshow(train_images[0],cmap=plt.cm.binary)
plt.show()
print(train_labels[0])
```




```
5
```

In [3]:
```
train_images = train_images / 255.0
test_images = test_images / 255.0
```

In [4]:
```
from keras.utils import to_categorical
train_labels = to_categorical(train_labels)
test_labels = to_categorical(test_labels)
```

In [5]:
```
from tensorflow.keras.layers import Dense, Activation, Flatten
from tensorflow.keras.models import Sequential

model = Sequential()
model.add(Flatten())
model.add(Dense(256, activation='relu'))
model.add(Dense(10, activation='softmax'))

model1 = Sequential()
```

```
model1.add(Flatten())
model1.add(Dense(128, activation='relu'))
model1.add(Dense(256, activation='relu'))
model1.add(Dense(10, activation='softmax'))
```

In [6]:
```
model.compile(optimizer='adam',loss='categorical_crossentropy', metrics=['accura
model1.compile(tf.keras.optimizers.SGD(learning_rate=0.1),loss='categorical_cros
```

In [7]:
```
model.fit(train_images, train_labels, epochs=5, batch_size=128)
```

```
Epoch 1/5
469/469 [==============================] - 1s 1ms/step - loss: 0.3096 - accurac
y: 0.9136
Epoch 2/5
469/469 [==============================] - 1s 1ms/step - loss: 0.1311 - accurac
y: 0.9623
Epoch 3/5
469/469 [==============================] - 1s 1ms/step - loss: 0.0882 - accurac
y: 0.9746
Epoch 4/5
469/469 [==============================] - 1s 1ms/step - loss: 0.0664 - accurac
y: 0.9805
Epoch 5/5
469/469 [==============================] - 1s 3ms/step - loss: 0.0500 - accurac
y: 0.9856
```

Out[7]:
```
<keras.src.callbacks.History at 0x1631b647350>
```

In [8]:
```
model1.fit(train_images, train_labels, epochs=25, batch_size=128)
```

```
Epoch 1/25
469/469 [==============================] - 2s 4ms/step - loss: 0.4611 - accurac
y: 0.8704
Epoch 2/25
469/469 [==============================] - 2s 4ms/step - loss: 0.2240 - accurac
y: 0.9347
Epoch 3/25
469/469 [==============================] - 2s 4ms/step - loss: 0.1673 - accurac
y: 0.9511
Epoch 4/25
469/469 [==============================] - 2s 4ms/step - loss: 0.1348 - accurac
y: 0.9605
Epoch 5/25
469/469 [==============================] - 2s 4ms/step - loss: 0.1114 - accurac
y: 0.9675
Epoch 6/25
469/469 [==============================] - 2s 4ms/step - loss: 0.0955 - accurac
y: 0.9718
Epoch 7/25
469/469 [==============================] - 2s 4ms/step - loss: 0.0829 - accurac
y: 0.9759
Epoch 8/25
469/469 [==============================] - 2s 4ms/step - loss: 0.0731 - accurac
y: 0.9789
Epoch 9/25
469/469 [==============================] - 2s 4ms/step - loss: 0.0652 - accurac
y: 0.9808
Epoch 10/25
469/469 [==============================] - 2s 4ms/step - loss: 0.0584 - accurac
y: 0.9825
Epoch 11/25
469/469 [==============================] - 2s 4ms/step - loss: 0.0525 - accurac
y: 0.9842
Epoch 12/25
469/469 [==============================] - 2s 4ms/step - loss: 0.0477 - accurac
y: 0.9863
Epoch 13/25
469/469 [==============================] - 2s 4ms/step - loss: 0.0432 - accurac
y: 0.9874
Epoch 14/25
469/469 [==============================] - 2s 4ms/step - loss: 0.0390 - accurac
y: 0.9893
Epoch 15/25
469/469 [==============================] - 2s 4ms/step - loss: 0.0355 - accurac
y: 0.9900
Epoch 16/25
469/469 [==============================] - 2s 4ms/step - loss: 0.0319 - accurac
y: 0.9909
Epoch 17/25
469/469 [==============================] - 2s 4ms/step - loss: 0.0291 - accurac
y: 0.9922
Epoch 18/25
469/469 [==============================] - 1s 3ms/step - loss: 0.0262 - accurac
y: 0.9930
Epoch 19/25
469/469 [==============================] - 2s 4ms/step - loss: 0.0240 - accurac
y: 0.9939
Epoch 20/25
469/469 [==============================] - 2s 4ms/step - loss: 0.0213 - accurac
y: 0.9947
```

```
Epoch 21/25
469/469 [==============================] - 2s 4ms/step - loss: 0.0193 - accurac
y: 0.9953
Epoch 22/25
469/469 [==============================] - 2s 4ms/step - loss: 0.0179 - accurac
y: 0.9957
Epoch 23/25
469/469 [==============================] - 2s 4ms/step - loss: 0.0159 - accurac
y: 0.9963
Epoch 24/25
469/469 [==============================] - 2s 4ms/step - loss: 0.0144 - accurac
y: 0.9970
Epoch 25/25
469/469 [==============================] - 2s 4ms/step - loss: 0.0132 - accurac
y: 0.9976
```

Out[8]: <keras.src.callbacks.History at 0x1631c9ce7d0>

In [9]:
```
test_loss, test_acc = model.evaluate(test_images, test_labels)
print('test_acc:', test_acc)
```

```
313/313 [==============================] - 1s 2ms/step - loss: 0.0682 - accurac
y: 0.9782
test_acc: 0.9782000184059143
```

In [10]:
```
test_loss, test_acc = model1.evaluate(test_images, test_labels)
print('test_acc:', test_acc)
```

```
313/313 [==============================] - 1s 2ms/step - loss: 0.0729 - accurac
y: 0.9796
test_acc: 0.9796000123023987
```

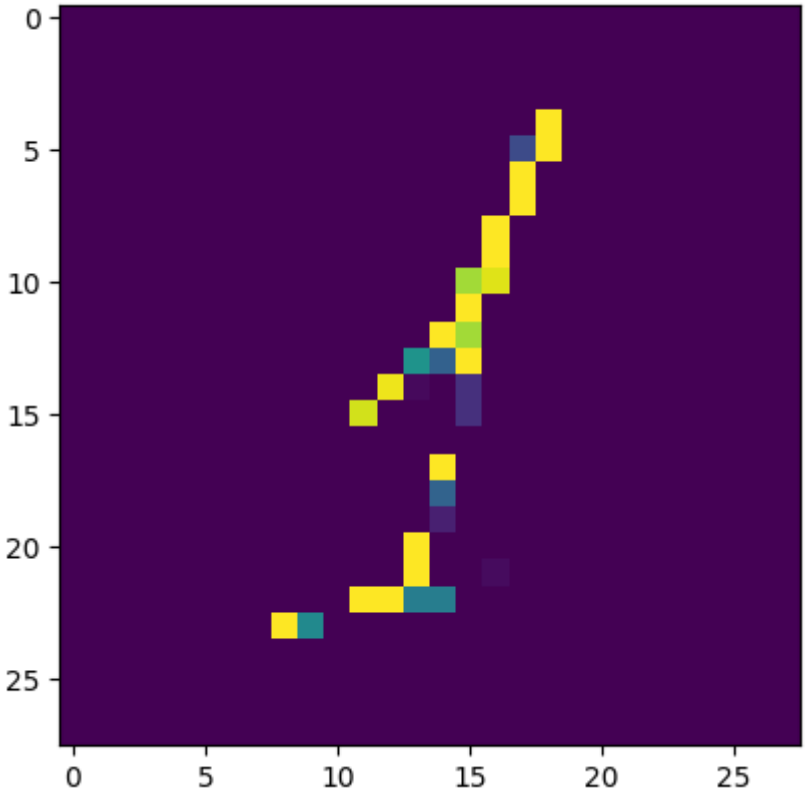
In [11]:
```
from scipy.ndimage.measurements import center_of_mass
import math
import cv2
import numpy as np
from IPython.display import Image, display
import matplotlib.pyplot as plt

def getBestShift(img):
    cy,cx = center_of_mass(img)

    rows,cols = img.shape
    shiftx = np.round(cols/2.0-cx).astype(int)
    shifty = np.round(rows/2.0-cy).astype(int)

    return shiftx,shifty

def shift(img,sx,sy):
    rows,cols = img.shape
    M = np.float32([[1,0,sx],[0,1,sy]])
    shifted = cv2.warpAffine(img,M,(cols,rows))
    return shifted

def rec_digit(img_path):
    display(Image(img_path))
    img = cv2.imread(img_path, cv2.IMREAD_GRAYSCALE)
    gray = 255-img
  # применяем пороговую обработку
    (thresh, gray) = cv2.threshold(gray, 128, 255, cv2.THRESH_BINARY | cv2.THRES
```

```
  # удаляем нулевые строки и столбцы
    while np.sum(gray[0]) == 0:
        gray = gray[1:]
    while np.sum(gray[:,0]) == 0:
        gray = np.delete(gray,0,1)
    while np.sum(gray[-1]) == 0:
        gray = gray[:-1]
    while np.sum(gray[:,-1]) == 0:
        gray = np.delete(gray,-1,1)
    rows,cols = gray.shape

  # изменяем размер, чтобы помещалось в box 20x20 пикселей
    if rows > cols:
        factor = 20.0/rows
        rows = 20
        cols = int(round(cols*factor))
        gray = cv2.resize(gray, (cols,rows))
    else:
        factor = 20.0/cols
        cols = 20
        rows = int(round(rows*factor))
        gray = cv2.resize(gray, (cols, rows))

  # расширяем до размера 28x28
    colsPadding = (int(math.ceil((28-cols)/2.0)),int(math.floor((28-cols)/2.0)))
    rowsPadding = (int(math.ceil((28-rows)/2.0)),int(math.floor((28-rows)/2.0)))
    gray = np.lib.pad(gray,(rowsPadding,colsPadding),'constant')

  # сдвигаем центр масс
    shiftx,shifty = getBestShift(gray)
    shifted = shift(gray,shiftx,shifty)
    gray = shifted

    cv2.imwrite('gray'+ img_path, gray)
    img = gray / 255.0
    img = np.array(img).reshape(-1, 28, 28, 1)

    out2 = str(np.argmax(model1.predict(img)))
    out = str(np.argmax(model.predict(img)))
    plt.imshow(img.squeeze(0))
    plt.show()

    print(f"Распознанная цифра первой моделью: {out}")
    print(f"Распознанная цифра второй моделью: {out2}")
```

```
C:\Users\Daniel\AppData\Local\Temp\ipykernel_27780\621413984.py:1: DeprecationW
arning: Please use `center_of_mass` from the `scipy.ndimage` namespace, the `sc
ipy.ndimage.measurements` namespace is deprecated.
  from scipy.ndimage.measurements import center_of_mass
```

In [12]:
```
rec_digit("C:\\Users\\Daniel\\Downloads\\11.png")
```

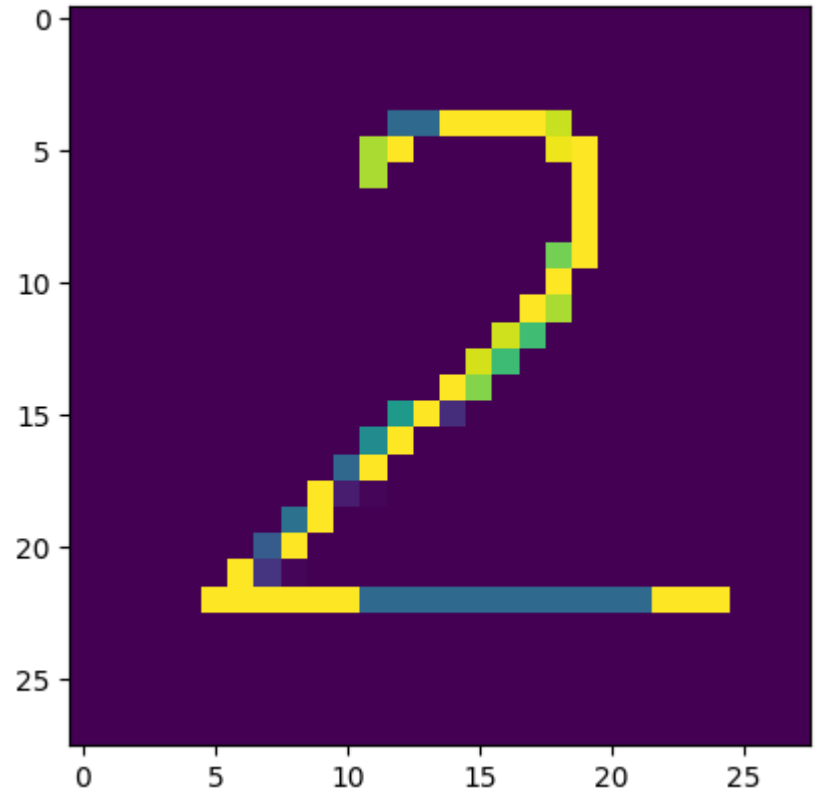
```
1/1 [==============================] - 0s 67ms/step
1/1 [==============================] - 0s 43ms/step
```




```
Распознанная цифра первой моделью: 1
Распознанная цифра второй моделью: 1
```

In [13]:
```
rec_digit("C:\\Users\\Daniel\\Downloads\\2.png")
```

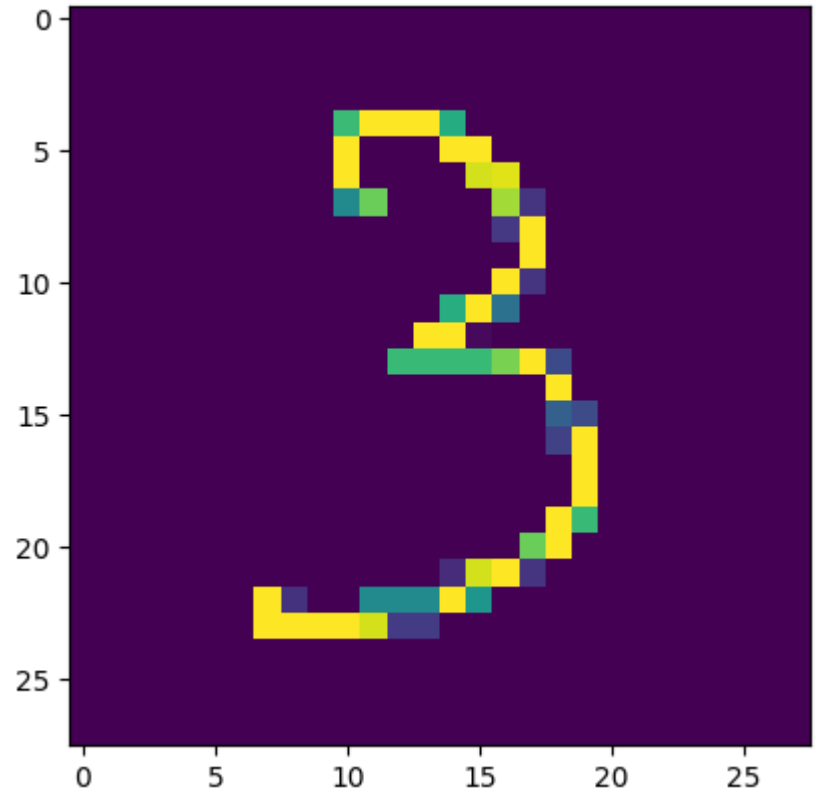
```
1/1 [==============================] - 0s 17ms/step
1/1 [==============================] - 0s 17ms/step
```




```
Распознанная цифра первой моделью: 2
Распознанная цифра второй моделью: 2
```

In [14]:
```
rec_digit("C:\\Users\\Daniel\\Downloads\\3.png")
```

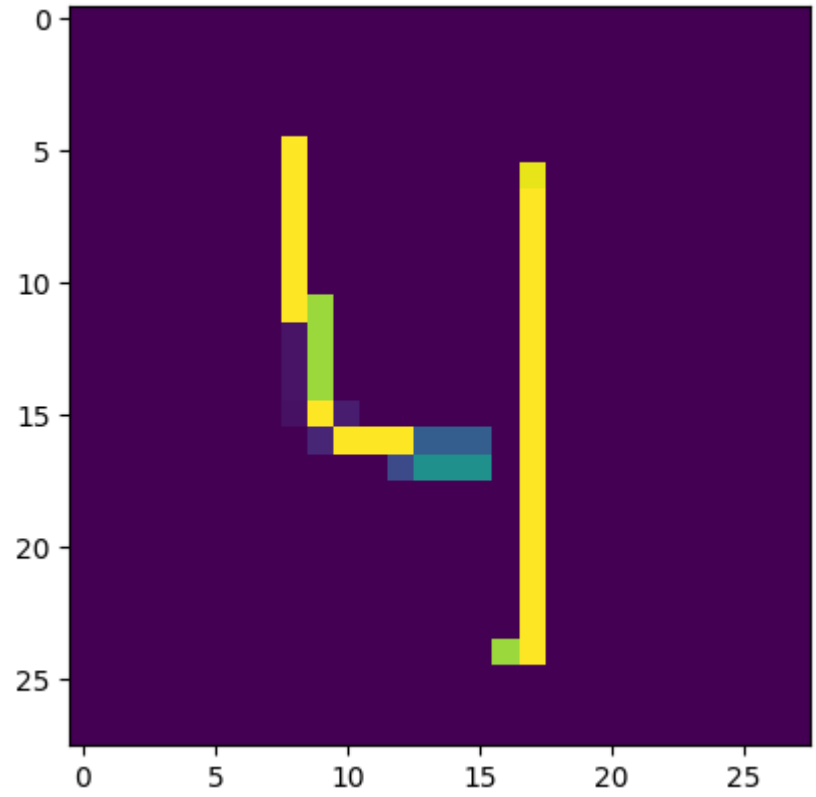
```
1/1 [==============================] - 0s 16ms/step
1/1 [==============================] - 0s 16ms/step
```




```
Распознанная цифра первой моделью: 3
Распознанная цифра второй моделью: 3
```

In [15]:
```
rec_digit("C:\\Users\\Daniel\\Downloads\\4.png")
```

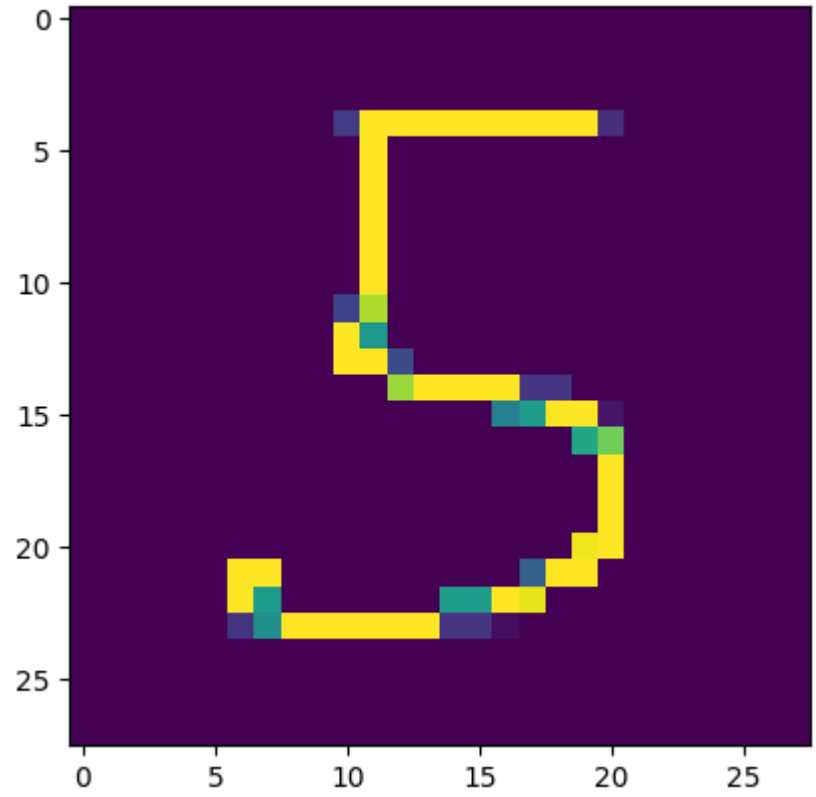
```
1/1 [==============================] - 0s 16ms/step
1/1 [==============================] - 0s 16ms/step
```




```
Распознанная цифра первой моделью: 4
Распознанная цифра второй моделью: 4
```

In [16]:
```
rec_digit("C:\\Users\\Daniel\\Downloads\\5.png")
```

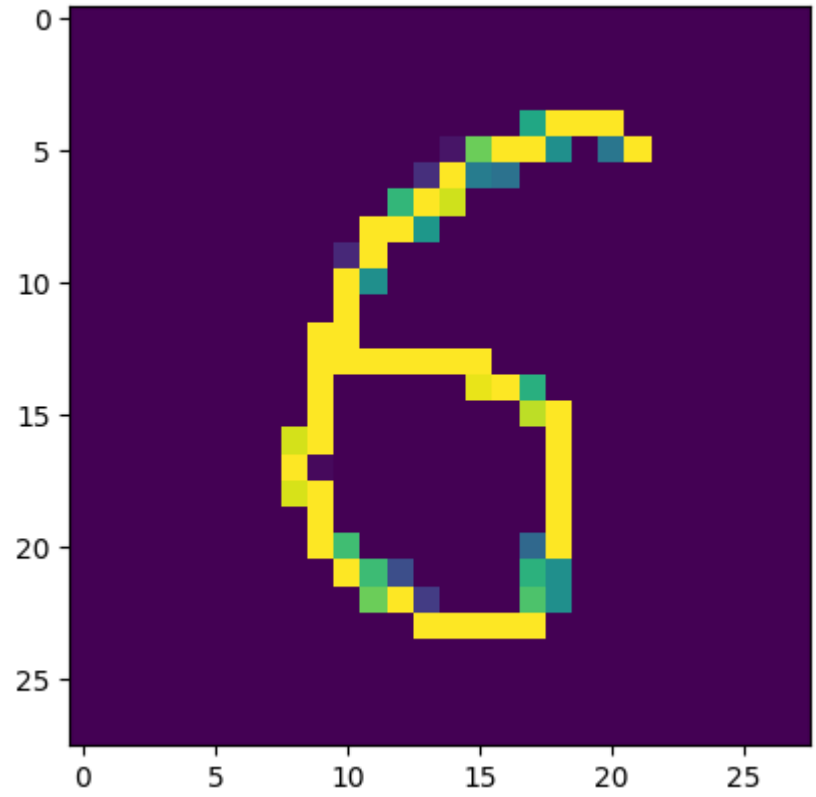
```
1/1 [==============================] - 0s 16ms/step
1/1 [==============================] - 0s 82ms/step
```




```
Распознанная цифра первой моделью: 5
Распознанная цифра второй моделью: 5
```

In [17]:
```
rec_digit("C:\\Users\\Daniel\\Downloads\\6.png")
```

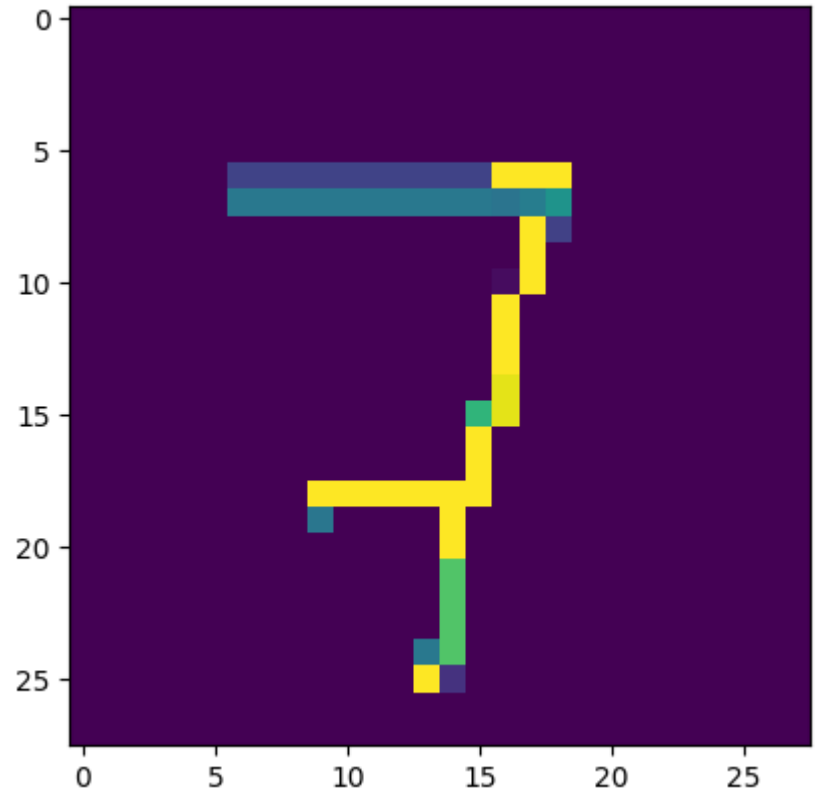
```
1/1 [==============================] - 0s 16ms/step
1/1 [==============================] - 0s 10ms/step
```




```
Распознанная цифра первой моделью: 6
Распознанная цифра второй моделью: 6
```

In [18]:
```
rec_digit("C:\\Users\\Daniel\\Downloads\\7.png")
```

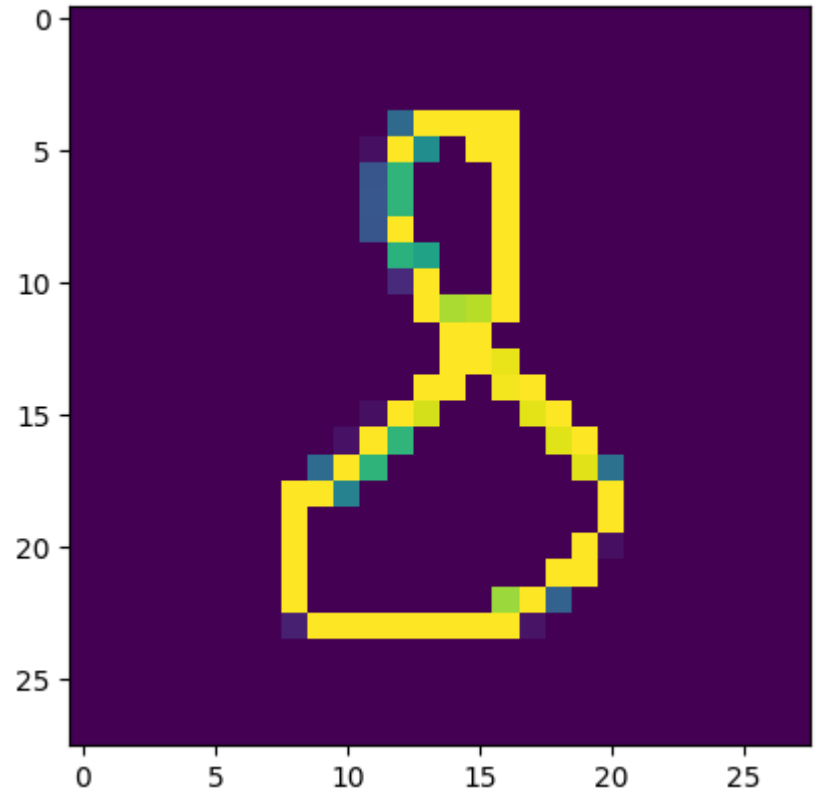
```
1/1 [==============================] - 0s 17ms/step
1/1 [==============================] - 0s 17ms/step
```




```
Распознанная цифра первой моделью: 7
Распознанная цифра второй моделью: 7
```

In [19]:
```
rec_digit("C:\\Users\\Daniel\\Downloads\\8.png")
```

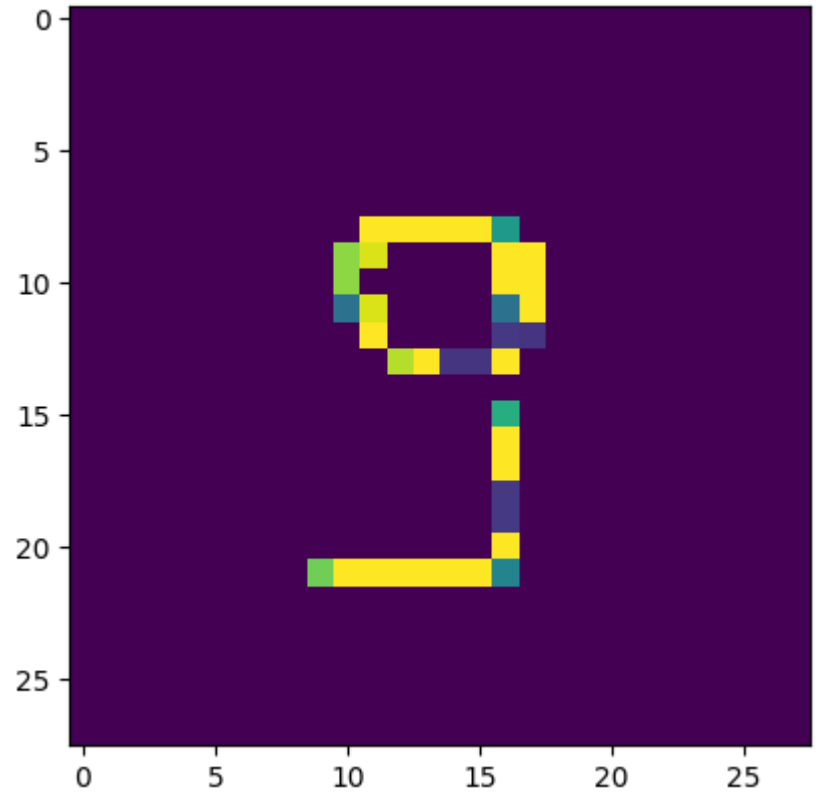
```
1/1 [==============================] - 0s 17ms/step
1/1 [==============================] - 0s 16ms/step
```




```
Распознанная цифра первой моделью: 8
Распознанная цифра второй моделью: 8
```

In [20]:
```
rec_digit("C:\\Users\\Daniel\\Downloads\\9.png")
```

```
1/1 [==============================] - 0s 16ms/step
1/1 [==============================] - 0s 17ms/step
```




```
Распознанная цифра первой моделью: 1
Распознанная цифра второй моделью: 9
```