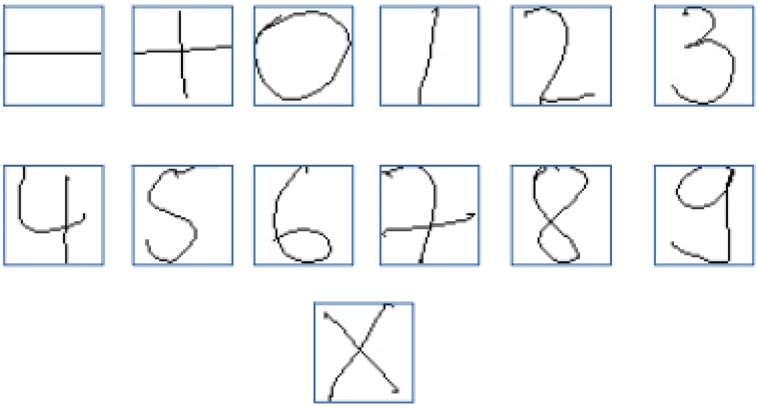
Datset Used: Handwritten math symbols dataset

- This Dataset consists of 45x45 jpg files
- This dataset consists of the handwritten digits and mathematical sysmbols such as addition (+), subtraction (-), multiplication (x), etc.
- Link for the dataset - https://www.kaggle.com/xainano/handwrittenmathsymbols (https://www.kaggle.com/xainano/handwrittenmathsymbols)



This project we have divided into three parts:

1) Feature Extraction

2) Training the data using CNN

3) Testing the Model

Feature Extraction

We have used contour extraction method to obtain features from the images. In this method we perform the following steps:

1. We first take negative of an image.
2. Convert the each image into a binary image
3. To find contours use ‘findContour’ function.
4. For features, we draw the bounding rectangle of contour using ‘boundingRect’ function.
5. Resize the image to 28x28.
6. Reshape it to 784x1.
7. Give the corresponding label to it. We have assigned for 0–9 images same label as their digit, for — assign label 10, for + assign label 11, for times assign label 12

In [1]:

```
# Importing the Required Libraries

import numpy as np
import cv2
from PIL import Image
from matplotlib import pyplot as plt
%matplotlib inline
import os
from os import listdir
from os.path import isfile, join
import pandas as pd
```

In [2]:

```
## Function to perform Feature Extration on Images from the folder.

def load_images_from_folder(folder):
    train_data = []

    # Load each image file from entire directory
    for filename in os.listdir(folder):
        # Reading an image
        img = cv2.imread(os.path.join(folder,filename),cv2.IMREAD_GRAYSCALE)

        # Negative of an image.
        img = ~img

        if img is not None:
            # Convert the image to a binary image
            ret, thresh = cv2.threshold(img, 127,255, cv2.THRESH_BINARY)

            # Find Contour
            ctrs, ret = cv2.findContours(thresh,cv2.RETR_EXTERNAL,cv2.CHAIN_APPROX_NONE
)

            # Construct bounding rectangle
            cnt = sorted(ctrs, key=lambda ctr: cv2.boundingRect(ctr)[0])

            # Resizing the image to 28x28
            w = int(28)
            h = int(28)
            maxi = 0

            for c in cnt:
                x,y,w,h = cv2.boundingRect(c)
                maxi = max(w*h,maxi)
                if (maxi == w*h):
                    x_max = x
                    y_max = y
                    w_max = w
                    h_max = h

            im_crop = thresh[y_max:y_max+h_max+10, x_max:x_max+w_max+10]
            im_resize = cv2.resize(im_crop,(28,28))

            # Reshaping the image to 784x1
            im_resize = np.reshape(im_resize,(784,1))
            train_data.append(im_resize)
    return train_data
```

In [21]:

```
data = []
```

In [22]:

```
#assign '-' = 10
data = load_images_from_folder('train images/-')

for i in range(0,len(data)):
    data[i] = np.append(data[i],['10'])

print(len(data))
```

```
4152
```

In [23]:

```
#assign '+' = 11
data11=load_images_from_folder('train images/+')

for i in range(0,len(data11)):
    data11[i] = np.append(data11[i],['11'])

data = np.concatenate((data,data11))

print(len(data))
```

```
8184
```

In [24]:

```
data0 = load_images_from_folder('train images/0')
for i in range(0,len(data0)):
    data0[i] = np.append(data0[i],['0'])

data = np.concatenate((data,data0))
print(len(data))
```

```
12018
```

In [25]:

```
data1 = load_images_from_folder('train images/1')

for i in range(0,len(data1)):
    data1[i] = np.append(data1[i],['1'])

data = np.concatenate((data,data1))
print(len(data))
```

```
16074
```

In [26]:

```
data2 = load_images_from_folder('train images/2')

for i in range(0,len(data2)):
    data2[i] = np.append(data2[i],['2'])

data = np.concatenate((data,data2))
print(len(data))
```

```
20334
```

In [27]:

```
data3 = load_images_from_folder('train images/3')

for i in range(0,len(data3)):
    data3[i] = np.append(data3[i],['3'])

data = np.concatenate((data,data3))
print(len(data))
```

```
23850
```

In [28]:

```
data4 = load_images_from_folder('train images/4')

for i in range(0,len(data4)):
    data4[i] = np.append(data4[i],['4'])

data=np.concatenate((data,data4))
print(len(data))
```

```
27882
```

In [29]:

```
data5 = load_images_from_folder('train images/5')

for i in range(0,len(data5)):
    data5[i] = np.append(data5[i],['5'])

data = np.concatenate((data,data5))
print(len(data))
```

```
31426
```

In [30]:

```
data6 = load_images_from_folder('train images/6')

for i in range(0,len(data6)):
    data6[i] = np.append(data6[i],['6'])

data = np.concatenate((data,data6))
print(len(data))
```

```
34543
```

In [31]:

```
data7 = load_images_from_folder('train images/7')

for i in range(0,len(data7)):
    data7[i] = np.append(data7[i],['7'])

data = np.concatenate((data,data7))
print(len(data))
```

```
37451
```

In [32]:

```
data8 = load_images_from_folder('train images/8')

for i in range(0,len(data8)):
    data8[i] = np.append(data8[i],['8'])

data = np.concatenate((data,data8))
print(len(data))
```

```
40518
```

In [33]:

```
data9 = load_images_from_folder('train images/9')

for i in range(0,len(data9)):
    data9[i] = np.append(data9[i],['9'])

data = np.concatenate((data,data9))
print(len(data))
```

```
44254
```

In [34]:

```
#assign * = 12
data12=load_images_from_folder('train images/times')

for i in range(0,len(data12)):
    data12[i] = np.append(data12[i],['12'])

data = np.concatenate((data,data12))
print(len(data))
```

```
47504
```

In [35]:

```
#assign ( = 13
data13 = load_images_from_folder('train images/(')

for i in range(0,len(data13)):
    data13[i] = np.append(data13[i],['13'])

data = np.concatenate((data, data13))
print(len(data))
```

```
51768
```

In [36]:

```
#assign ) = 14
data14 = load_images_from_folder('train images/)')

for i in range(0,len(data14)):
    data14[i] = np.append(data14[i],['14'])

data = np.concatenate((data, data1))
print(len(data))
```

```
55824
```

In [37]:

```
#assign [ = 15
data15 = load_images_from_folder('train images/[')

for i in range(0,len(data15)):
    data15[i] = np.append(data15[i],['15'])

data = np.concatenate((data, data15))
print(len(data))
```

```
56602
```

In [38]:

```
#assign ] = 16
data16 = load_images_from_folder('train images/]')

for i in range(0,len(data16)):
    data16[i] = np.append(data16[i],['16'])

data = np.concatenate((data, data16))
print(len(data))
```

```
57382
```

In [39]:

```
#assign '=' = 17
data17 = load_images_from_folder('train images/=')

for i in range(0,len(data17)):
    data17[i] = np.append(data17[i],['17'])

data = np.concatenate((data, data17))
print(len(data))
```

```
61394
```

In [40]:

```
#assign x = 18
data18 = load_images_from_folder('train images/X')

for i in range(0,len(data18)):
    data18[i] = np.append(data18[i],['18'])

data = np.concatenate((data, data18))
print(len(data))
```

```
65536
```

In [41]:

```
#assign y = 17
data19 = load_images_from_folder('train images/y')

for i in range(0,len(data19)):
    data19[i] = np.append(data19[i],['19'])

data = np.concatenate((data, data19))
print(len(data))
```

```
69761
```

In [42]:

```
# Add all the features to the 'train_final.csv' file
data_frame = pd.DataFrame(data,index = None)
data_frame.to_csv('train.csv',index = False)
```

Training the data using CNN

In [3]:

```
import pandas as pd
import numpy as np
import pickle

df_train = pd.read_csv('train.csv', index_col = False)
labels = df_train[['784']]

df_train.drop(df_train.columns[[784]], axis=1, inplace = True)
df_train.head()
```

Out[3]:

	0	1	2	3	4	5	6	7	8	9	...	774	775	776	777	778	779	7
0	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	
1	255	255	255	255	255	255	255	255	255	255	...	0	0	0	0	0	0	
2	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	
3	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	
4	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	

5 rows × 784 columns

In [4]:

```
np.random.seed(1212)
import cv2
import keras
from keras.models import Model
from keras.layers import *
from keras import optimizers
from keras.layers import Input, Dense
from keras.models import Sequential
from keras.layers import Dense
from keras.layers import Dropout
from keras.layers import Flatten
from keras.layers.convolutional import Conv2D
from keras.layers.convolutional import MaxPooling2D
from keras.datasets import mnist
from keras.utils import np_utils
from keras import backend as K
from keras.models import model_from_json
K.set_image_dim_ordering('th')
```

```
Using TensorFlow backend.
C:\Users\PRTIK\Anaconda3\lib\site-packages\tensorflow\python\framework\dty
pes.py:526: FutureWarning: Passing (type, 1) or '1type' as a synonym of ty
pe is deprecated; in a future version of numpy, it will be understood as
(type, (1,)) / '(1,)type'.
  _np_qint8 = np.dtype([("qint8", np.int8, 1)])
C:\Users\PRTIK\Anaconda3\lib\site-packages\tensorflow\python\framework\dty
pes.py:527: FutureWarning: Passing (type, 1) or '1type' as a synonym of ty
pe is deprecated; in a future version of numpy, it will be understood as
(type, (1,)) / '(1,)type'.
  _np_quint8 = np.dtype([("quint8", np.uint8, 1)])
C:\Users\PRTIK\Anaconda3\lib\site-packages\tensorflow\python\framework\dty
pes.py:528: FutureWarning: Passing (type, 1) or '1type' as a synonym of ty
pe is deprecated; in a future version of numpy, it will be understood as
(type, (1,)) / '(1,)type'.
  _np_qint16 = np.dtype([("qint16", np.int16, 1)])
C:\Users\PRTIK\Anaconda3\lib\site-packages\tensorflow\python\framework\dty
pes.py:529: FutureWarning: Passing (type, 1) or '1type' as a synonym of ty
pe is deprecated; in a future version of numpy, it will be understood as
(type, (1,)) / '(1,)type'.
  _np_quint16 = np.dtype([("quint16", np.uint16, 1)])
C:\Users\PRTIK\Anaconda3\lib\site-packages\tensorflow\python\framework\dty
pes.py:530: FutureWarning: Passing (type, 1) or '1type' as a synonym of ty
pe is deprecated; in a future version of numpy, it will be understood as
(type, (1,)) / '(1,)type'.
  _np_qint32 = np.dtype([("qint32", np.int32, 1)])
C:\Users\PRTIK\Anaconda3\lib\site-packages\tensorflow\python\framework\dty
pes.py:535: FutureWarning: Passing (type, 1) or '1type' as a synonym of ty
pe is deprecated; in a future version of numpy, it will be understood as
(type, (1,)) / '(1,)type'.
  np_resource = np.dtype([("resource", np.ubyte, 1)])
```

In [5]:

```
labels = np.array(labels)

from keras.utils.np_utils import to_categorical
cat = to_categorical(labels, num_classes = 20)

print(cat[0])

df_train.head()
```

```
[0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
```

Out[5]:

	0	1	2	3	4	5	6	7	8	9	...	774	775	776	777	778	779	7
0	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	
1	255	255	255	255	255	255	255	255	255	255	...	0	0	0	0	0	0	
2	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	
3	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	
4	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	

5 rows × 784 columns

In [6]:

```
l = []
for i in range(69761):
    l.append(np.array(df_train[i:i+1]).reshape(1,28,28))

np.random.seed(7)
```

In [7]:

```
model = Sequential()

model.add(Conv2D(30, (5, 5), input_shape=(1 , 28, 28), activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))

model.add(Conv2D(15, (3, 3), activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))

model.add(Dropout(0.2))

model.add(Flatten())

model.add(Dense(128, activation='relu'))
model.add(Dense(50, activation='relu'))
model.add(Dense(20, activation='softmax'))

# Compile model
model.compile(loss = 'categorical_crossentropy', optimizer = 'adam', metrics = ['accuracy'])

model.fit(np.array(l), cat, epochs = 100, batch_size = 200, shuffle = True, verbose = 1)
```

```
WARNING:tensorflow:From C:\Users\PRTIK\Anaconda3\lib\site-packages\tensorf
low\python\framework\op_def_library.py:263: colocate_with (from tensorflo
w.python.framework.ops) is deprecated and will be removed in a future vers
ion.
Instructions for updating:
Colocations handled automatically by placer.
WARNING:tensorflow:From C:\Users\PRTIK\Anaconda3\lib\site-packages\keras\b
ackend\tensorflow_backend.py:3445: calling dropout (from tensorflow.pytho
n.ops.nn_ops) with keep_prob is deprecated and will be removed in a future
version.
Instructions for updating:
Please use `rate` instead of `keep_prob`. Rate should be set to `rate = 1
- keep_prob`.
WARNING:tensorflow:From C:\Users\PRTIK\Anaconda3\lib\site-packages\tensorf
low\python\ops\math_ops.py:3066: to_int32 (from tensorflow.python.ops.math
_ops) is deprecated and will be removed in a future version.
Instructions for updating:
Use tf.cast instead.
Epoch 1/100
69761/69761 [==============================] - 66s 946us/step - loss: 1.37
11 - acc: 0.6373
Epoch 2/100
69761/69761 [==============================] - 66s 939us/step - loss: 0.39
56 - acc: 0.8634
Epoch 3/100
69761/69761 [==============================] - 65s 935us/step - loss: 0.29
15 - acc: 0.8943
Epoch 4/100
69761/69761 [==============================] - 63s 900us/step - loss: 0.23
95 - acc: 0.9108
Epoch 5/100
69761/69761 [==============================] - 65s 930us/step - loss: 0.21
31 - acc: 0.9184
Epoch 6/100
69761/69761 [==============================] - 66s 942us/step - loss: 0.18
53 - acc: 0.9270
Epoch 7/100
69761/69761 [==============================] - 65s 933us/step - loss: 0.17
61 - acc: 0.9304
Epoch 8/100
69761/69761 [==============================] - 65s 937us/step - loss: 0.16
58 - acc: 0.9340
Epoch 9/100
69761/69761 [==============================] - 66s 941us/step - loss: 0.15
46 - acc: 0.9385
Epoch 10/100
69761/69761 [==============================] - 65s 933us/step - loss: 0.14
63 - acc: 0.94052s - loss: 0.146
Epoch 11/100
69761/69761 [==============================] - 65s 938us/step - loss: 0.14
45 - acc: 0.9405
Epoch 12/100
69761/69761 [==============================] - 65s 933us/step - loss: 0.13
71 - acc: 0.9440
Epoch 13/100
69761/69761 [==============================] - 65s 937us/step - loss: 0.13
15 - acc: 0.9451
Epoch 14/100
69761/69761 [==============================] - 65s 933us/step - loss: 0.13
09 - acc: 0.9453
Epoch 15/100
```

```
69761/69761 [==============================] - 65s 930us/step - loss: 0.12
61 - acc: 0.9464
Epoch 16/100
69761/69761 [==============================] - 65s 937us/step - loss: 0.12
64 - acc: 0.9483
Epoch 17/100
69761/69761 [==============================] - 65s 932us/step - loss: 0.12
04 - acc: 0.9493
Epoch 18/100
69761/69761 [==============================] - 65s 934us/step - loss: 0.11
97 - acc: 0.9493
Epoch 19/100
69761/69761 [==============================] - 65s 933us/step - loss: 0.11
67 - acc: 0.9516
Epoch 20/100
69761/69761 [==============================] - 66s 941us/step - loss: 0.11
76 - acc: 0.9504
Epoch 21/100
69761/69761 [==============================] - 65s 933us/step - loss: 0.11
12 - acc: 0.9529
Epoch 22/100
69761/69761 [==============================] - 65s 935us/step - loss: 0.11
04 - acc: 0.9529
Epoch 23/100
69761/69761 [==============================] - 65s 937us/step - loss: 0.10
83 - acc: 0.9539
Epoch 24/100
69761/69761 [==============================] - 64s 920us/step - loss: 0.10
89 - acc: 0.9541
Epoch 25/100
69761/69761 [==============================] - 63s 904us/step - loss: 0.10
87 - acc: 0.9534
Epoch 26/100
69761/69761 [==============================] - 61s 874us/step - loss: 0.10
46 - acc: 0.9555
Epoch 27/100
69761/69761 [==============================] - 61s 878us/step - loss: 0.10
77 - acc: 0.9541
Epoch 28/100
69761/69761 [==============================] - 63s 906us/step - loss: 0.10
44 - acc: 0.9548
Epoch 29/100
69761/69761 [==============================] - 66s 948us/step - loss: 0.10
12 - acc: 0.9571
Epoch 30/100
69761/69761 [==============================] - 66s 941us/step - loss: 0.09
88 - acc: 0.9571
Epoch 31/100
69761/69761 [==============================] - 64s 919us/step - loss: 0.10
09 - acc: 0.9569
Epoch 32/100
69761/69761 [==============================] - 64s 918us/step - loss: 0.09
76 - acc: 0.9577
Epoch 33/100
69761/69761 [==============================] - 64s 924us/step - loss: 0.09
80 - acc: 0.9578
Epoch 34/100
69761/69761 [==============================] - 67s 956us/step - loss: 0.09
81 - acc: 0.9583
Epoch 35/100
69761/69761 [==============================] - 66s 944us/step - loss: 0.09
```

```
75 - acc: 0.9587
Epoch 36/100
69761/69761 [==============================] - 64s 921us/step - loss: 0.09
73 - acc: 0.9585
Epoch 37/100
69761/69761 [==============================] - 66s 947us/step - loss: 0.09
58 - acc: 0.9594
Epoch 38/100
69761/69761 [==============================] - 70s 1ms/step - loss: 0.0954
- acc: 0.9590
Epoch 39/100
69761/69761 [==============================] - 65s 933us/step - loss: 0.09
38 - acc: 0.9605
Epoch 40/100
69761/69761 [==============================] - 65s 925us/step - loss: 0.09
47 - acc: 0.9604
Epoch 41/100
69761/69761 [==============================] - 64s 918us/step - loss: 0.09
07 - acc: 0.9618
Epoch 42/100
69761/69761 [==============================] - 66s 948us/step - loss: 0.09
08 - acc: 0.9612
Epoch 43/100
69761/69761 [==============================] - 65s 933us/step - loss: 0.09
26 - acc: 0.9620
Epoch 44/100
69761/69761 [==============================] - 64s 920us/step - loss: 0.08
98 - acc: 0.9624
Epoch 45/100
69761/69761 [==============================] - 64s 920us/step - loss: 0.08
62 - acc: 0.9633
Epoch 46/100
69761/69761 [==============================] - 64s 922us/step - loss: 0.08
97 - acc: 0.9621
Epoch 47/100
69761/69761 [==============================] - 65s 925us/step - loss: 0.08
72 - acc: 0.9635
Epoch 48/100
69761/69761 [==============================] - 64s 920us/step - loss: 0.09
05 - acc: 0.9625
Epoch 49/100
69761/69761 [==============================] - 64s 921us/step - loss: 0.08
81 - acc: 0.9632
Epoch 50/100
69761/69761 [==============================] - 65s 927us/step - loss: 0.08
47 - acc: 0.9650
Epoch 51/100
69761/69761 [==============================] - 64s 921us/step - loss: 0.08
46 - acc: 0.9646
Epoch 52/100
69761/69761 [==============================] - 64s 922us/step - loss: 0.08
84 - acc: 0.9635
Epoch 53/100
69761/69761 [==============================] - 64s 919us/step - loss: 0.08
58 - acc: 0.9645
Epoch 54/100
69761/69761 [==============================] - 64s 920us/step - loss: 0.08
20 - acc: 0.9657
Epoch 55/100
69761/69761 [==============================] - 64s 920us/step - loss: 0.08
63 - acc: 0.9638
```

```
Epoch 56/100
69761/69761 [==============================] - 64s 919us/step - loss: 0.08
44 - acc: 0.9646
Epoch 57/100
69761/69761 [==============================] - 65s 936us/step - loss: 0.08
33 - acc: 0.9651
Epoch 58/100
69761/69761 [==============================] - 64s 920us/step - loss: 0.08
32 - acc: 0.9661
Epoch 59/100
69761/69761 [==============================] - 64s 919us/step - loss: 0.08
25 - acc: 0.9653
Epoch 60/100
69761/69761 [==============================] - 64s 919us/step - loss: 0.08
24 - acc: 0.9660
Epoch 61/100
69761/69761 [==============================] - 67s 958us/step - loss: 0.08
10 - acc: 0.9667
Epoch 62/100
69761/69761 [==============================] - 65s 927us/step - loss: 0.08
33 - acc: 0.9658
Epoch 63/100
69761/69761 [==============================] - 64s 921us/step - loss: 0.07
98 - acc: 0.9671
Epoch 64/100
69761/69761 [==============================] - 65s 938us/step - loss: 0.08
15 - acc: 0.9666
Epoch 65/100
69761/69761 [==============================] - 65s 927us/step - loss: 0.08
10 - acc: 0.9669
Epoch 66/100
69761/69761 [==============================] - 64s 922us/step - loss: 0.08
24 - acc: 0.9660
Epoch 67/100
69761/69761 [==============================] - 64s 919us/step - loss: 0.07
90 - acc: 0.9673
Epoch 68/100
69761/69761 [==============================] - 65s 925us/step - loss: 0.08
14 - acc: 0.9668
Epoch 69/100
69761/69761 [==============================] - 64s 922us/step - loss: 0.07
90 - acc: 0.9680
Epoch 70/100
69761/69761 [==============================] - 64s 920us/step - loss: 0.08
00 - acc: 0.9674
Epoch 71/100
69761/69761 [==============================] - 64s 924us/step - loss: 0.08
07 - acc: 0.9672
Epoch 72/100
69761/69761 [==============================] - 65s 927us/step - loss: 0.07
72 - acc: 0.9680
Epoch 73/100
69761/69761 [==============================] - 64s 919us/step - loss: 0.07
85 - acc: 0.9683
Epoch 74/100
69761/69761 [==============================] - 64s 921us/step - loss: 0.07
85 - acc: 0.9675
Epoch 75/100
69761/69761 [==============================] - 58s 836us/step - loss: 0.07
60 - acc: 0.9687
Epoch 76/100
```

```
69761/69761 [==============================] - 52s 743us/step - loss: 0.07
81 - acc: 0.9680
Epoch 77/100
69761/69761 [==============================] - 47s 677us/step - loss: 0.07
55 - acc: 0.9691
Epoch 78/100
69761/69761 [==============================] - 45s 640us/step - loss: 0.07
93 - acc: 0.9678
Epoch 79/100
69761/69761 [==============================] - 44s 628us/step - loss: 0.07
91 - acc: 0.9681
Epoch 80/100
69761/69761 [==============================] - 43s 614us/step - loss: 0.07
66 - acc: 0.9682
Epoch 81/100
69761/69761 [==============================] - 43s 623us/step - loss: 0.07
52 - acc: 0.9696
Epoch 82/100
69761/69761 [==============================] - 42s 603us/step - loss: 0.07
58 - acc: 0.9693
Epoch 83/100
69761/69761 [==============================] - 41s 594us/step - loss: 0.07
40 - acc: 0.9697
Epoch 84/100
69761/69761 [==============================] - 41s 589us/step - loss: 0.07
36 - acc: 0.9698
Epoch 85/100
69761/69761 [==============================] - 41s 590us/step - loss: 0.07
41 - acc: 0.9695
Epoch 86/100
69761/69761 [==============================] - 42s 608us/step - loss: 0.07
45 - acc: 0.9699
Epoch 87/100
69761/69761 [==============================] - 43s 618us/step - loss: 0.07
19 - acc: 0.9711
Epoch 88/100
69761/69761 [==============================] - 42s 600us/step - loss: 0.07
22 - acc: 0.9707
Epoch 89/100
69761/69761 [==============================] - 41s 591us/step - loss: 0.07
77 - acc: 0.9689
Epoch 90/100
69761/69761 [==============================] - 41s 591us/step - loss: 0.07
44 - acc: 0.9699
Epoch 91/100
69761/69761 [==============================] - 46s 665us/step - loss: 0.07
20 - acc: 0.9709
Epoch 92/100
69761/69761 [==============================] - 42s 603us/step - loss: 0.07
11 - acc: 0.9709
Epoch 93/100
69761/69761 [==============================] - 41s 594us/step - loss: 0.07
15 - acc: 0.9708
Epoch 94/100
69761/69761 [==============================] - 41s 589us/step - loss: 0.07
42 - acc: 0.9706
Epoch 95/100
69761/69761 [==============================] - 45s 648us/step - loss: 0.07
21 - acc: 0.9706
Epoch 96/100
69761/69761 [==============================] - 42s 609us/step - loss: 0.07
```

```
10 - acc: 0.9709
Epoch 97/100
69761/69761 [==============================] - 44s 631us/step - loss: 0.07
11 - acc: 0.9714
Epoch 98/100
69761/69761 [==============================] - 44s 634us/step - loss: 0.07
43 - acc: 0.9704
Epoch 99/100
69761/69761 [==============================] - 44s 628us/step - loss: 0.06
97 - acc: 0.9717
Epoch 100/100
69761/69761 [==============================] - 41s 588us/step - loss: 0.07
09 - acc: 0.9707
```

Out[7]:

```
<keras.callbacks.History at 0x1231acb3198>
```

In [9]:

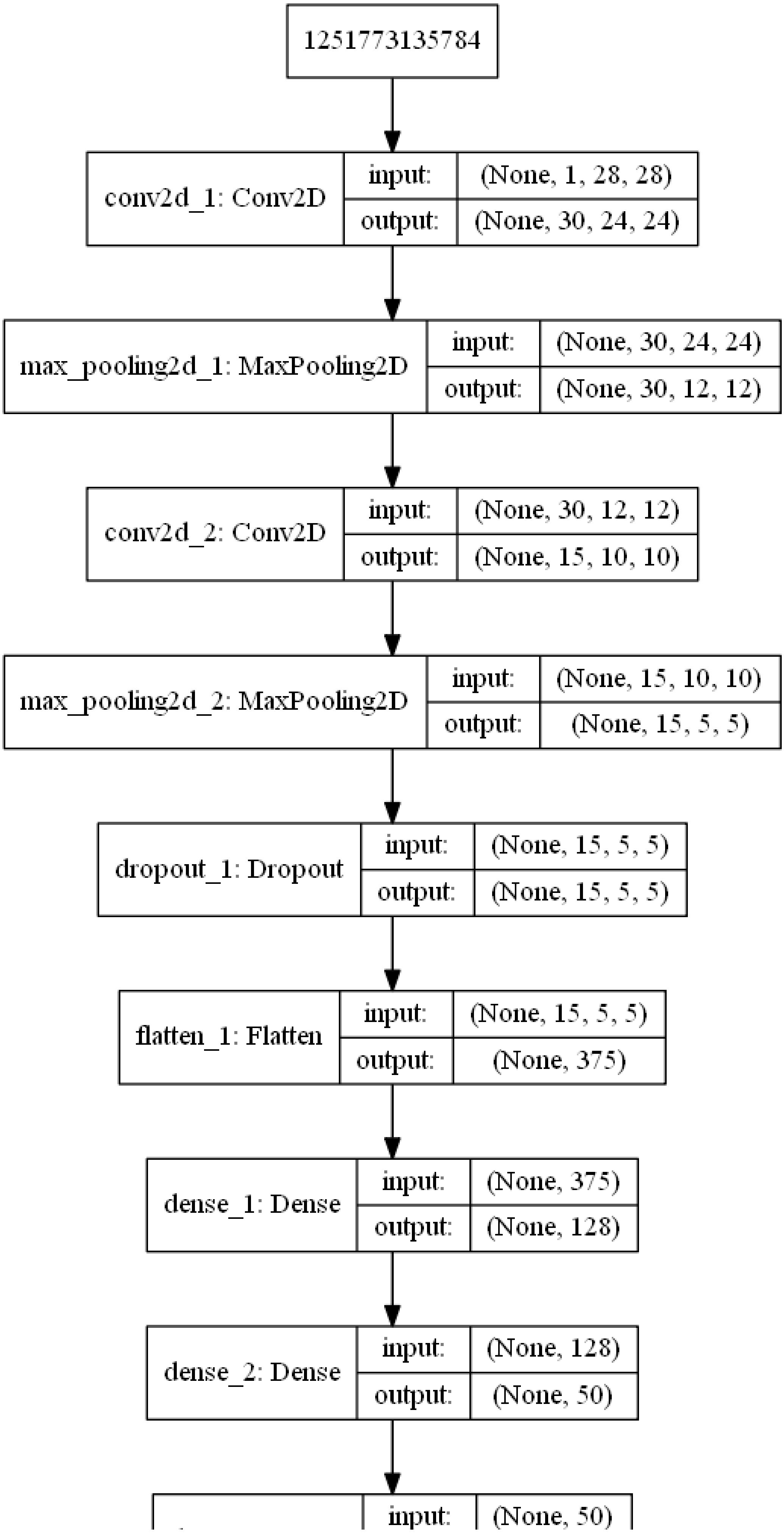
```
model_json = model.to_json()
with open("model_final.json", "w") as json_file:
    json_file.write(model_json)

# serialize weights to HDF5
model.save_weights("model_final.h5")
```

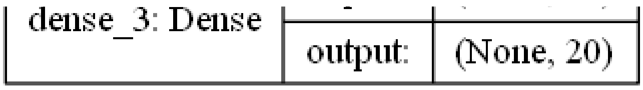
In [10]:

```
print(model.summary())

import pydotplus
import keras.utils
keras.utils.vis_utils.pydot = pydotplus
keras.utils.plot_model(model, to_file='model_plot.png', show_shapes=True)
#plot_model(model, to_file='model_plot.png', show_shapes=True, show_layer_names=True)
from IPython.display import Image
Image(filename = 'model_plot.png')
```

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #   
=================================================================
conv2d_1 (Conv2D)            (None, 30, 24, 24)        780       
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 30, 12, 12)        0         
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 15, 10, 10)        4065      
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 15, 5, 5)          0         
_________________________________________________________________
dropout_1 (Dropout)          (None, 15, 5, 5)          0         
_________________________________________________________________
flatten_1 (Flatten)          (None, 375)               0         
_________________________________________________________________
dense_1 (Dense)              (None, 128)               48128     
_________________________________________________________________
dense_2 (Dense)              (None, 50)                6450      
_________________________________________________________________
dense_3 (Dense)              (None, 20)                1020      
=================================================================
Total params: 60,443
Trainable params: 60,443
Non-trainable params: 0
_________________________________________________________________
None
```

Out[10]:


1251773135784
conv2d_1: Conv2D
input: (None, 1, 28, 28)
output: (None, 30, 24, 24)
max_pooling2d_1: MaxPooling2D
input: (None, 30, 24, 24)
output: (None, 30, 12, 12)
conv2d_2: Conv2D
input: (None, 30, 12, 12)
output: (None, 15, 10, 10)
max_pooling2d_2: MaxPooling2D
input: (None, 15, 10, 10)
output: (None, 15, 5, 5)
dropout_1: Dropout
input: (None, 15, 5, 5)
output: (None, 15, 5, 5)
flatten_1: Flatten
input: (None, 15, 5, 5)
output: (None, 375)
dense_1: Dense
input: (None, 375)
output: (None, 128)
dense_2: Dense
input: (None, 128)
output: (None, 50)
input: (None, 50)



Testing the Model

In [20]:

```
json_file = open('model_final_hes.json', 'r')
loaded_model_json = json_file.read()
json_file.close()
loaded_model_hes = model_from_json(loaded_model_json)

# load weights into new model
loaded_model_hes.load_weights("model_final_hes.h5")
```

In [11]:

```
json_file = open('model_final.json', 'r')
loaded_model_json = json_file.read()
json_file.close()
loaded_model = model_from_json(loaded_model_json)

# load weights into new model
loaded_model.load_weights("model_final.h5")
```

In [21]:

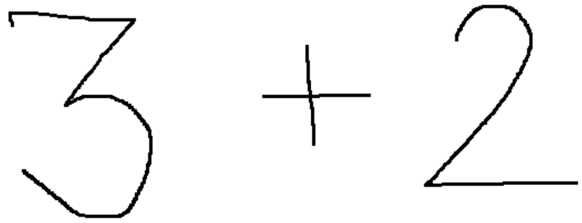
```
def handwritten_equation_solver(file_name):
    #file_name = 'C:/Users/PRTIK/Handwritten-Equation-Solver/test_images/2symbols.png'
    file_name = file_name
    img = cv2.imread(file_name, cv2.IMREAD_GRAYSCALE)

    cv2.imshow("Original Image",img)
    cv2.imshow("Negative of Original Image",~img)
    cv2.waitKey(0)
    cv2.destroyAllWindows()

    if img is not None:
        img = ~img
        ret, thresh = cv2.threshold(img, 127, 255, cv2.THRESH_BINARY)
        ctrs, ret = cv2.findContours(thresh, cv2.RETR_TREE, cv2.CHAIN_APPROX_SIMPLE)
        cnt = sorted(ctrs, key=lambda ctr: cv2.boundingRect(ctr)[0])

        w = int(28)
        h = int(28)
        train_data = []
        rects = []

        for c in cnt :
            x, y, w, h = cv2.boundingRect(c)
            rect = [x,y,w,h]
            rects.append(rect)
        #print(rects)

        bool_rect = []
        for r in rects:
            l = []
            for rec in rects:
                flag = 0
                if rec != r:
                    if r[0]<(rec[0]+rec[2]+10) and rec[0]<(r[0]+r[2]+10) and r[1]<(rec[
1]+rec[3]+10) and rec[1]<(r[1]+r[3]+10):
                        flag = 1
                    l.append(flag)
                if rec == r:
                    l.append(0)
            bool_rect.append(l)
        #print(bool_rect)

        dump_rect = []
        for i in range(0,len(cnt)):
            for j in range(0,len(cnt)):
                if bool_rect[i][j] == 1:
                    area1 = rects[i][2]*rects[i][3]
                    area2 = rects[j][2]*rects[j][3]
                    if(area1 == min(area1,area2)):
                        dump_rect.append(rects[i])
        #print(len(dump_rect))

        final_rect=[i for i in rects if i not in dump_rect]
        #print(final_rect)

        for r in final_rect:
            x = r[0]
            y = r[1]
            w = r[2]
            h = r[3]
            im_crop = thresh[y:y+h+10, x:x+w+10]


            im_resize = cv2.resize(im_crop, (28,28))
            cv2.imshow("Contour", im_resize)
            cv2.waitKey(0)
            cv2.destroyAllWindows()

            im_resize = np.reshape(im_resize,(1,28,28))
            train_data.append(im_resize)


    # Recognition and Solution of the Expression
    expression = ''

    for i in range(len(train_data)):
        train_data[i] = np.array(train_data[i])
        train_data[i] = train_data[i].reshape(1,1,28,28)
        result = loaded_model_hes.predict_classes(train_data[i])

        if(result[0] == 10):
            expression = expression + '-'

        if(result[0] == 11):
            expression = expression + '+'

        if(result[0] == 12):
            expression = expression + '*'

        if(result[0] == 0):
            expression = expression + '0'

        if(result[0] == 1):
            expression = expression + '1'

        if(result[0] == 2):
            expression = expression + '2'

        if(result[0] == 3):
            expression = expression + '3'

        if(result[0] == 4):
            expression = expression + '4'

        if(result[0] == 5):
            expression = expression + '5'

        if(result[0] == 6):
            expression = expression + '6'

        if(result[0] == 7):
            expression = expression + '7'

        if(result[0] == 8):
            expression = expression + '8'

        if(result[0] == 9):
            expression = expression + '9'

        #if(result[0] == 13):
```

```
        #    expression = expression + '/'

        if(result[0] == 13):
            expression = expression + '('

        if(result[0] == 14):
            expression = expression + ')'

        if(result[0] == 15):
            expression = expression + '['

        if(result[0] == 16):
            expression = expression + ']'

        if(result[0] == 17):
            expression = expression + '='

        if(result[0] == 18):
            expression = expression + 'x'

        if(result[0] == 19):
            expression = expression + 'y'

    print("Input Image : ")

    from IPython.display import display
    from PIL import Image
    display(Image.open(file_name))

    print("Expression recognised is : ", expression)

    answer = eval(expression)
    print("                  Answer : ", answer)
```

In [6]:

```
file_name = 'C:/Users/PRTIK/Handwritten-Equation-Solver/test_images/3+2.png'
handwritten_equation_solver(file_name = file_name)
```

```
Input Image :
```

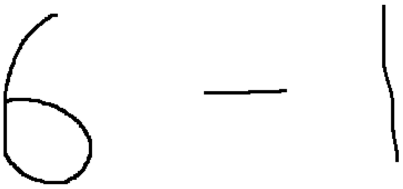


```
Expression recognised is :  3+2
                  Answer :  5
```

In [5]:

```
file_name = 'C:/Users/PRTIK/Handwritten-Equation-Solver/test_images/6-1.png'
handwritten_equation_solver(file_name = file_name)
```

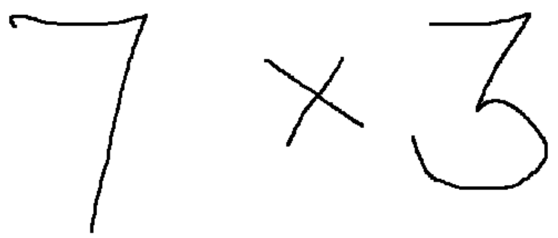
```
Input Image :
```



```
Expression recognised is :  6-1
                  Answer :  5
```

```
In [19]:
file_name = 'C:/Users/PRTIK/Handwritten-Equation-Solver/test_images/7x3.png'
handwritten_equation_solver(file_name)
```

```
Input Image :
```

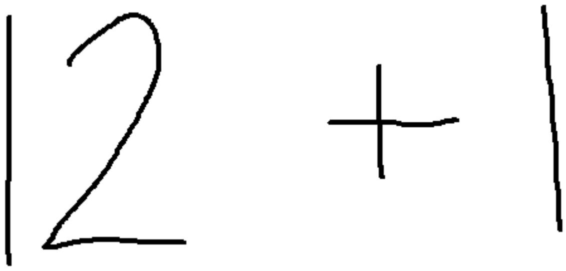


```
Expression recognised is :  7*3
                  Answer :  21
```

In [6]:

```
file_name = 'C:/Users/PRTIK/Handwritten-Equation-Solver/test_images/twelveplus1.png'
handwritten_equation_solver(file_name)
```

```
Input Image :
```

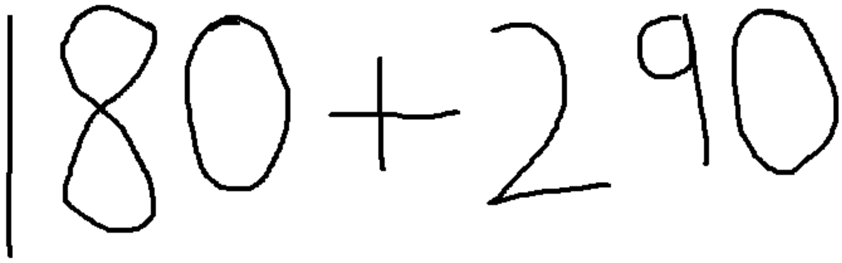


```
Expression recognised is :  12+1
                  Answer :  13
```

In [52]:

```
file_name = 'C:/Users/PRTIK/Handwritten-Equation-Solver/test_images/180+290.png'
handwritten_equation_solver(file_name)
```

```
Input Image :
```

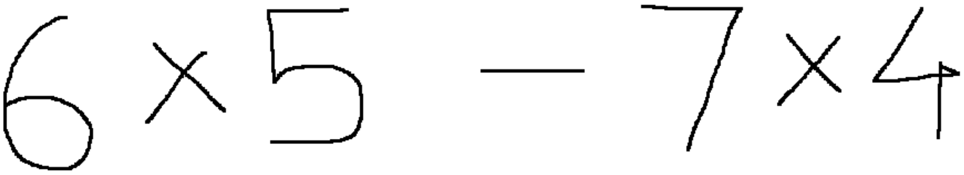


```
Expression recognised is :  180+290
                  Answer :  470
```

In [8]:

```
file_name = 'C:/Users/PRTIK/Handwritten-Equation-Solver/test_images/3symbols.png'
handwritten_equation_solver(file_name = file_name)
```

```
Input Image :
```

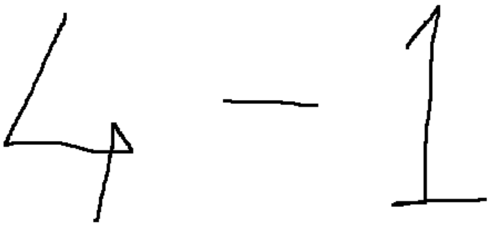


```
Expression recognised is :  6*5-7*4
                  Answer :  2
```

In [9]:

```
file_name = 'C:/Users/PRTIK/Handwritten-Equation-Solver/test_images/4-1.png'
handwritten_equation_solver(file_name = file_name)
```

```
Input Image :
```

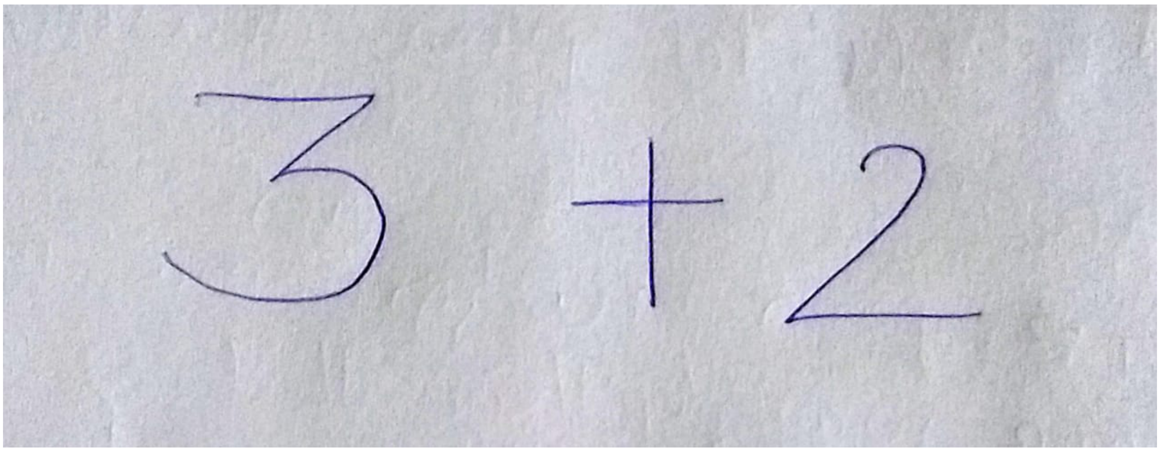


```
Expression recognised is :  4-2
                  Answer :  2
```

In [56]:

```
file_name = 'C:/Users/PRTIK/Handwritten-Equation-Solver/test_images/3+2_copy.png'
handwritten_equation_solver(file_name = file_name)
```

```
Input Image :
```

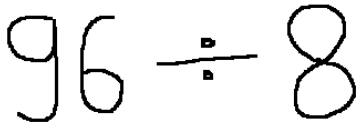


```
Expression recognised is :  3+2
                  Answer :  5
```

In [65]:

```
file_name = 'C:/Users/PRTIK/Handwritten-Equation-Solver/test_images/div_brackets.png'
handwritten_equation_solver(file_name = file_name)
```

```
Input Image :
```



```
Expression recognised is :  96/8
                  Answer :  12.0
```

Solving Linear Equations in 2 variable

In [13]:

```
def linear_equation_expression(file_name):

    file_name = file_name
    img = cv2.imread(file_name, cv2.IMREAD_GRAYSCALE)

    cv2.imshow("\n Original Image",img)
    #cv2.imshow("Negative of Original Image",~img)
    cv2.waitKey(0)
    cv2.destroyAllWindows()

    if img is not None:
        img = ~img
        ret, thresh = cv2.threshold(img, 127, 255, cv2.THRESH_BINARY)
        ctrs, ret = cv2.findContours(thresh, cv2.RETR_TREE, cv2.CHAIN_APPROX_SIMPLE)
        cnt = sorted(ctrs, key=lambda ctr: cv2.boundingRect(ctr)[0])

        w = int(28)
        h = int(28)
        train_data = []
        rects = []

        for c in cnt :
            x, y, w, h = cv2.boundingRect(c)
            rect = [x,y,w,h]
            rects.append(rect)
        #print(rects)

        bool_rect = []
        for r in rects:
            l = []
            for rec in rects:
                flag = 0
                if rec != r:
                    if r[0]<(rec[0]+rec[2]+10) and rec[0]<(r[0]+r[2]+10) and r[1]<(rec[
1]+rec[3]+10) and rec[1]<(r[1]+r[3]+10):
                        flag = 1
                    l.append(flag)
                if rec == r:
                    l.append(0)
            bool_rect.append(l)
        #print(bool_rect)

        dump_rect = []
        for i in range(0,len(cnt)):
            for j in range(0,len(cnt)):
                if bool_rect[i][j] == 1:
                    area1 = rects[i][2]*rects[i][3]
                    area2 = rects[j][2]*rects[j][3]
                    if(area1 == min(area1,area2)):
                        dump_rect.append(rects[i])
        #print(len(dump_rect))

        final_rect=[i for i in rects if i not in dump_rect]
        #print(final_rect)

        for r in final_rect:
            x = r[0]
            y = r[1]
            w = r[2]
            h = r[3]
            im_crop = thresh[y:y+h+10, x:x+w+10]


            im_resize = cv2.resize(im_crop, (28,28))
            #cv2.imshow("Contour", im_resize)
            cv2.waitKey(0)
            cv2.destroyAllWindows()

            im_resize = np.reshape(im_resize,(1,28,28))
            train_data.append(im_resize)


    # Recognition and Solution of the Expression
    expression = ''

    for i in range(len(train_data)):
        train_data[i] = np.array(train_data[i])
        train_data[i] = train_data[i].reshape(1,1,28,28)
        result = loaded_model.predict_classes(train_data[i])

        if(result[0] == 10):
            expression = expression + '-'

        if(result[0] == 11):
            expression = expression + '+'

        if(result[0] == 12):
            expression = expression + '*'

        if(result[0] == 0):
            expression = expression + '0'

        if(result[0] == 1):
            expression = expression + '1'

        if(result[0] == 2):
            expression = expression + '2'

        if(result[0] == 3):
            expression = expression + '3'

        if(result[0] == 4):
            expression = expression + '4'

        if(result[0] == 5):
            expression = expression + '5'

        if(result[0] == 6):
            expression = expression + '6'

        if(result[0] == 7):
            expression = expression + '7'

        if(result[0] == 8):
            expression = expression + '8'

        if(result[0] == 9):
            expression = expression + '9'

        #if(result[0] == 13):
```

```
        #    expression = expression + '/'

        if(result[0] == 13):
            expression = expression + '('

        if(result[0] == 14):
            expression = expression + ')'

        if(result[0] == 15):
            expression = expression + '['

        if(result[0] == 16):
            expression = expression + ']'

        if(result[0] == 17):
            expression = expression + '='

        if(result[0] == 18):
            expression = expression + 'x'

        if(result[0] == 19):
            expression = expression + 'y'

    print("\nInput Image : ")

    from IPython.display import display
    from PIL import Image
    display(Image.open(file_name))

    print("Expression recognised is : ", expression)

    return expression
```

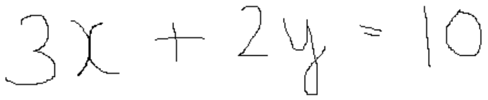
In [14]:

```
def linear_eqn_solver(eq1, eq2):

    list_eq1 = list(eq1)
    list_eq2 = list(eq2)

    equation1_A = []
    equation2_A = []

    A = []
    B = []
    X = []

    con1 = ''
    con2 = ''
    con3 = ''

    def constants_from_equation(list_eq):

        con1 = ''
        con2 = ''
        con3 = ''

        for ii in range(len(list_eq)):
            #for jj in range(2):
            if list_eq[ii] != 'x':
                con1 = con1 + list_eq[ii]

            elif list_eq[0] == 'x':
                con1 = 1
                break

            else:
                break

        for jj in range(ii+2, len(list_eq)):
            #for jj in range(2):
            if list_eq[jj] != 'y':
                con2 = con2 + list_eq[jj]

            elif list_eq[ii+2] == 'y':
                con1 = 1
                break

            else:
                break

        for kk in range(jj+2, len(list_eq)):
            con3 = con3 + list_eq[kk]

        #equation1_A.append(int(con1))
        #equation1_A.append(int(con2))

        return con1, con2, con3

    con1, con2, con3 = constants_from_equation(list_eq1)
    equation1_A.append(int(con1))
    equation1_A.append(int(con2))
    B.append(int(con3))
    con1, con2, con3 = constants_from_equation(list_eq2)
    equation2_A.append(int(con1))
    equation2_A.append(int(con2))
    B.append(int(con3))

    A.append(equation1_A)
    A.append(equation2_A)

    X = np.linalg.solve(np.array(A), np.array(B))

    #print(X)
    #print('[x, y] = ', X)

    return X
```

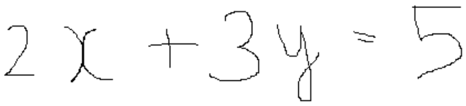
In [99]:

```
file_name = 'C:/Users/PRTIK/Handwritten-Equation-Solver/test_images/eq2_1.png'
eq1 = linear_equation_expression(file_name = file_name)

file_name = 'C:/Users/PRTIK/Handwritten-Equation-Solver/test_images/eq2_2.png'
eq2 = linear_equation_expression(file_name = file_name)

solution = linear_eqn_solver(eq1 = eq1, eq2 = eq2)

print("\nEquation 1 : ", eq1)
print("Equation 2 : ", eq2)
print("Solution   : [x, y] = ", solution)
```

```
Input Image :
```



```
Expression recognised is :  3x+2y=10

Input Image :
```

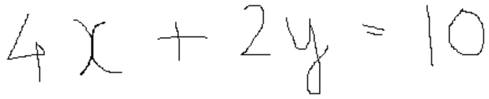


```
Expression recognised is :  2x+3y=5

Equation 1 :  3x+2y=10
Equation 2 :  2x+3y=5
Solution   : [x, y] =  [ 4. -1.]
```

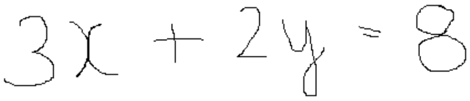
In [96]:

```
file_name = 'C:/Users/PRTIK/Handwritten-Equation-Solver/test_images/eq4_1.png'
eq1 = linear_equation_expression(file_name = file_name)

file_name = 'C:/Users/PRTIK/Handwritten-Equation-Solver/test_images/eq4_2.png'
eq2 = linear_equation_expression(file_name = file_name)

solution = linear_eqn_solver(eq1 = eq1, eq2 = eq2)

print("\nEquation 1 : ", eq1)
print("Equation 2 : ", eq2)
print("Solution   : [x, y] = ", solution)
```

```
Input Image :
```



```
Expression recognised is :  4x+2y=10

Input Image :
```



```
Expression recognised is :  3x+2y=8

Equation 1 :  4x+2y=10
Equation 2 :  3x+2y=8
Solution   : [x, y] =  [2. 1.]
```

```
In [ ]:
```