软件源代码：

from PIL import Image

import numpy as np

import scipy.io as sio

name = str(input("Please input the image name: "))

im = np.array(Image.open(name).convert('L'))

#im = np.array(Image.open("2.pic\_hd.jpg").convert('L'))

im[ im > 100 ] = 255

im[ im != 255 ] = 0

#im = Image.fromarray(im)

#im.save('7\_pic.png')

# 图片的宽度和高度

img\_size = im.size

#print("图片宽度和高度分别是{}".format(img\_size))

'''

裁剪：传入一个元组作为参数

元组里的元素分别是：（距离图片左边界距离x， 距离图片上边界距离y，距离图片左边界距离+裁剪框宽度x+w，距离图片上边界距离+裁剪框高度y+h）

'''

# 截取图片中一块宽是200和高是120的

M = int(input("Please input the printing type (1 or 2): "))

x = np.zeros(shape = (12))

y = np.zeros(shape = (12))

w = 240

h = np.zeros(shape = (12))

high = np.zeros(shape = (12,240),dtype = int)

#region = np.zeros(shape = (12,80,240))

#district = np.zeros(shape = (6,80,240))

name = ['I','II','III','aVR','aVL','aVF','V1','V2','V3','V4','V5','V6']

if M == 1:

#=================宽度改变================

ECG = im

line = 0

line\_extreme\_up = 0

#line\_extreme\_down = 0

before\_count\_up = 0

no\_area = 0

#before\_count\_down = 0

line1\_extreme = []

line3\_extreme = []

line\_candidate1 = []

line\_candidate2 = []

line\_candidate3 = []

line1 = np.zeros(shape = 240,dtype=int)

line2 = np.zeros(shape = 240,dtype=int)

line3 = np.zeros(shape = 240,dtype=int)

tem\_counts1 = 0

tem\_counts2 = 0

tem\_counts3 = 0

############################第一列###########################

for j in range(165 ,165 + 240):

for i in range (181,670):

if ECG[i][j] == 0 and line == 0:

#print(str(j)+ " " + str(i))

line\_candidate1.append(669 - i)

#print(line\_candidate1)

if i == 181:

if before\_count\_up == 0:

line\_extreme\_up = 1

line1\_extreme.append(j)

else :

line\_extreme\_up = 1

before\_count\_up = 1

elif (i >= 182 ) and (ECG[i][j] != 0) and (ECG[i-1][j] == 0) and (line == 0) :

line = line + 1

continue

if ECG[i][j] == 0 and line == 1:

line\_candidate2.append(669 - i)

'''

if len(line\_candidate2) < 1:

line\_candidate2.append(635 - i)

elif (line\_candidate2[len(line\_candidate2)-1] - (635 - i)) == 1:

line\_candidate2.append(635 - i)

'''

elif (ECG[i][j] != 0) and (ECG[i-1][j] == 0) and (line == 1) :

line = line + 1

continue

if ECG[i][j] == 0 and line == 2:

line\_candidate3.append(669 - i)

'''

if len(line\_candidate3) < 1:

line\_candidate3.append(669 - i)

elif (line\_candidate3[len(line\_candidate3)-1] - (669 - i)) == 1:

line\_candidate3.append(669 - i)

'''

elif (ECG[i][j] != 0) and (ECG[i-1][j] == 0) and (line == 2):

line = 0

break

x = j - 165

#print(x)

#print(line\_candidate1)

#print(line\_candidate2)

#print(line\_candidate3)

tem\_counts1 = np.bincount(line1)

tem\_counts2 = np.bincount(line2)

tem\_counts3 = np.bincount(line3)

#返回众数

tem\_counts1 = np.argmax(tem\_counts1)

tem\_counts2 = np.argmax(tem\_counts2)

tem\_counts3 = np.argmax(tem\_counts3)

if line\_candidate2 == []:

line\_candidate2 = line\_candidate1

line\_candidate3 = line\_candidate1

elif line\_candidate3 == []:

line\_candidate3 = line\_candidate2

if ((x == 0) or (x == 1)) :

if (line\_extreme\_up == 1 and before\_count\_up == 0): #and (begin == 0) :

line1[x] = np.max(line\_candidate1)

line2[x] = np.min(line\_candidate2)

line3[x] = np.min(line\_candidate3)

before\_count\_up = 1

#before\_count\_down = 0

line\_extreme\_up = 0

#line\_extreme\_down = 0

#begin = 1

elif (line\_extreme\_up == 0 and before\_count\_up == 1): #and (begin == 1):

line1[x] = 669 - 181

line2[x] = np.min(line\_candidate1)

line3[x] = np.min(line\_candidate2)

before\_count\_up = 1

before\_count\_down = 0

line\_extreme\_up = 0

line\_extreme\_down = 0

elif (line\_extreme\_up == 1 and before\_count\_up == 1): #and (begin == 1) :

line1[x] = np.max(line\_candidate1)

line2[x] = np.min(line\_candidate2)

line3[x] = np.min(line\_candidate3)

before\_count\_up = 0

#before\_count\_down = 0

line\_extreme\_up = 0

#line\_extreme\_down = 0

#begin = 0

elif (line\_extreme\_up == 0 and before\_count\_up == 0):

line1[x] = np.min(line\_candidate1)

line2[x] = np.min(line\_candidate2)

line3[x] = np.min(line\_candidate3)

else:

if (line\_extreme\_up == 1 and before\_count\_up == 0):

line1[x] = 669 - 181

if (line2[x-2]-line2[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line2[x] = np.max(line\_candidate2)

elif (line2[x-2]-line2[x-1] > 0):# or (High[x-3]-High[x-1] > 0):

line2[x] = np.min(line\_candidate2)

elif (line2[x-2]-line2[x-1] == 0):

if (line2[x-3]-line2[x-2] < 0):

line2[x] = np.max(line\_candidate2)

elif (line2[x-3]-line2[x-2] > 0):

line2[x] = np.min(line\_candidate2)

else :

line2[x] = np.min(line\_candidate2)

if (line3[x-2]-line3[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line3[x] = np.max(line\_candidate3)

elif (line3[x-2]-line3[x-1] > 0):

line3[x] = np.min(line\_candidate3)

elif (line3[x-2]-line3[x-1] == 0):

if (line3[x-3]-line3[x-2] < 0):

line3[x] = np.max(line\_candidate3)

elif (line3[x-3]-line3[x-2] > 0):

line3[x] = np.min(line\_candidate3)

else :

line3[x] = np.min(line\_candidate3)

if (line3[x-1] >= line3[x] + 120):

if (line3[x-2]-line3[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line3[x] = np.max(line\_candidate2)

elif (line3[x-2]-line3[x-1] > 0):# or (High[x-3]-High[x-1] > 0):

line3[x] = np.min(line\_candidate2)

elif (line3[x-2]-line3[x-1] == 0):

if (line3[x-3]-line3[x-2] < 0):

line3[x] = np.min(line\_candidate2)

else :

line3[x] = np.min(line\_candidate2)

#if line1[x] == (669 - 181 - 25) or

if (line2[x-1] >= line2[x] + 120):

if (line2[x-2]-line2[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line2[x] = np.max(line\_candidate1)

elif (line2[x-2]-line2[x-1] > 0):# or (High[x-3]-High[x-1] > 0):

line2[x] = np.min(line\_candidate1)

elif (line2[x-2]-line2[x-1] == 0):

if (line2[x-3]-line2[x-2] < 0):

line2[x] = np.min(line\_candidate1)

else :

line2[x] = np.min(line\_candidate1)

if (line2[x] == line1[x]):

if (line2[x-2]-line2[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line2[x] = np.max(line\_candidate2)

elif (line2[x-2]-line2[x-1] > 0):# or (High[x-3]-High[x-1] > 0):

line2[x] = np.min(line\_candidate2)

elif (line2[x-2]-line2[x-1] == 0):

if (line2[x-3]-line2[x-2] < 0):

line2[x] = np.min(line\_candidate2)

else :

line2[x] = np.min(line\_candidate2)

if (line3[x] == line2[x]):

if (line3[x-2]-line3[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line3[x] = np.max(line\_candidate3)

elif (line3[x-2]-line3[x-1] > 0):# or (High[x-3]-High[x-1] > 0):

line3[x] = np.min(line\_candidate3)

elif (line3[x-2]-line3[x-1] == 0):

if (line3[x-3]-line3[x-2] < 0):

line3[x] = np.min(line\_candidate3)

else :

line3[x] = np.min(line\_candidate3)

before\_count\_up = 1

#before\_count\_down = 1

line\_extreme\_up = 0

#line\_extreme\_down = 0

elif (line\_extreme\_up == 0 and before\_count\_up == 1):

line1[x] = 669 - 181

if (line2[x-2]-line2[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line2[x] = np.max(line\_candidate1)

elif (line2[x-2]-line2[x-1] > 0):# or (High[x-3]-High[x-1] > 0):

line2[x] = np.min(line\_candidate1)

elif (line2[x-2]-line2[x-1] == 0):

if (line2[x-3]-line2[x-2] < 0):

line2[x] = np.max(line\_candidate1)

elif (line2[x-3]-line2[x-2] > 0):

line2[x] = np.min(line\_candidate1)

else :

line2[x] = np.min(line\_candidate1)

if (line3[x-2]-line3[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line3[x] = np.max(line\_candidate2)

elif (line3[x-2]-line3[x-1] > 0):# or (High[x-3]-High[x-1] > 0):

line3[x] = np.min(line\_candidate2)

elif (line3[x-2]-line3[x-1] == 0):

if (line3[x-3]-line3[x-2] < 0):

line3[x] = np.max(line\_candidate2)

elif (line3[x-3]-line3[x-2] > 0):

line3[x] = np.min(line\_candidate2)

else :

line3[x] = np.min(line\_candidate2)

if (line3[x-1] >= line3[x] + 120):

if (line3[x-2]-line3[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line3[x] = np.max(line\_candidate1)

elif (line3[x-2]-line3[x-1] > 0):# or (High[x-3]-High[x-1] > 0):

line3[x] = np.min(line\_candidate1)

elif (line3[x-2]-line3[x-1] == 0):

if (line3[x-3]-line3[x-2] < 0):

line3[x] = np.min(line\_candidate1)

else :

line3[x] = np.min(line\_candidate1)

'''

if (line2[x] == line1[x]):

if (line2[x-2]-line2[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line2[x] = np.max(line\_candidate2)

elif (line2[x-2]-line2[x-1] > 0):# or (High[x-3]-High[x-1] > 0):

line2[x] = np.min(line\_candidate2)

elif (line2[x-2]-line2[x-1] == 0):

if (line2[x-3]-line2[x-2] < 0):

line2[x] = np.min(line\_candidate2)

else :

line2[x] = np.min(line\_candidate2)

'''

if (line3[x] == line2[x]):

if (line3[x-2]-line3[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line3[x] = np.max(line\_candidate2)

elif (line3[x-2]-line3[x-1] > 0):# or (High[x-3]-High[x-1] > 0):

line3[x] = np.min(line\_candidate2)

elif (line3[x-2]-line3[x-1] == 0):

if (line3[x-3]-line3[x-2] < 0):

line3[x] = np.min(line\_candidate2)

else :

line3[x] = np.min(line\_candidate2)

before\_count\_up = 1

#before\_count\_down = 0

line\_extreme\_up = 0

#line\_extreme\_down = 0

elif (line\_extreme\_up == 1 and before\_count\_up == 1):

line1[x] = 669 - 181

if (line2[x-2]-line2[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line2[x] = np.max(line\_candidate2)

elif (line2[x-2]-line2[x-1] > 0):# or (High[x-3]-High[x-1] > 0):

line2[x] = np.min(line\_candidate2)

elif (line2[x-2]-line2[x-1] == 0):

if (line2[x-3]-line2[x-2] < 0):

line2[x] = np.max(line\_candidate2)

elif (line2[x-3]-line2[x-2] > 0):

line2[x] = np.min(line\_candidate2)

else :

line2[x] = np.min(line\_candidate2)

if (line3[x-2]-line3[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line3[x] = np.max(line\_candidate3)

elif (line3[x-2]-line3[x-1] > 0):# or (High[x-3]-High[x-1] > 0):

line3[x] = np.min(line\_candidate3)

elif (line3[x-2]-line3[x-1] == 0):

if (line3[x-3]-line3[x-2] < 0):

line3[x] = np.max(line\_candidate3)

elif (line3[x-3]-line3[x-2] > 0):

line3[x] = np.min(line\_candidate3)

else :

line3[x] = np.min(line\_candidate3)

if (line3[x-1] >= line3[x] + 120):

if (line3[x-2]-line3[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line3[x] = np.max(line\_candidate2)

elif (line3[x-2]-line3[x-1] > 0):# or (High[x-3]-High[x-1] > 0):

line3[x] = np.min(line\_candidate2)

elif (line3[x-2]-line3[x-1] == 0):

if (line3[x-3]-line3[x-2] < 0):

line3[x] = np.min(line\_candidate2)

else :

line3[x] = np.min(line\_candidate2)

#if line1[x] == (669 - 181 - 25) or

if (line2[x-1] >= line2[x] + 120):

if (line2[x-2]-line2[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line2[x] = np.max(line\_candidate1)

elif (line2[x-2]-line2[x-1] > 0):# or (High[x-3]-High[x-1] > 0):

line2[x] = np.min(line\_candidate1)

elif (line2[x-2]-line2[x-1] == 0):

if (line2[x-3]-line2[x-2] < 0):

line2[x] = np.min(line\_candidate1)

else :

line2[x] = np.min(line\_candidate1)

if (line2[x] == line1[x]):

if (line2[x-2]-line2[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line2[x] = np.max(line\_candidate2)

elif (line2[x-2]-line2[x-1] > 0):# or (High[x-3]-High[x-1] > 0):

line2[x] = np.min(line\_candidate2)

elif (line2[x-2]-line2[x-1] == 0):

if (line2[x-3]-line2[x-2] < 0):

line2[x] = np.min(line\_candidate2)

else :

line2[x] = np.min(line\_candidate2)

if (line3[x] == line2[x]):

if (line3[x-2]-line3[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line3[x] = np.max(line\_candidate3)

elif (line3[x-2]-line3[x-1] > 0):# or (High[x-3]-High[x-1] > 0):

line3[x] = np.min(line\_candidate3)

elif (line3[x-2]-line3[x-1] == 0):

if (line3[x-3]-line3[x-2] < 0):

line3[x] = np.min(line\_candidate3)

else :

line3[x] = np.min(line\_candidate3)

line\_extreme\_up = 0

#line\_extreme\_down = 0

before\_count\_up = 0

#before\_count\_down = 1

elif (line\_extreme\_up == 0 and before\_count\_up == 0):

if (line1[x-1] - line2[x-1] >=0 ) and (line1[x-1] - line2[x-1] < (len(line\_candidate1) + 7)) and ((np.min(line\_candidate1) - np.max(line\_candidate2)) >= 50) :

line1[x] = np.min(line\_candidate1)

line2[x] = np.max(line\_candidate1)

#print(line1[x-1] - line2[x-1])

#print(len(line\_candidate1))

if (line3[x-2]-line3[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line3[x] = np.max(line\_candidate2)

elif (line3[x-2]-line3[x-1] > 0):# or (High[x-3]-High[x-1] > 0):

line3[x] = np.min(line\_candidate2)

elif (line3[x-2]-line3[x-1] == 0):

if (line3[x-3]-line3[x-2] < 0):

line3[x] = np.max(line\_candidate2)

elif (line3[x-3]-line3[x-2] > 0):

line3[x] = np.min(line\_candidate2)

else :

line3[x] = np.min(line\_candidate2)

if (line3[x-1] >= line3[x] + 120 ):

if (line3[x-2]-line3[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line3[x] = np.max(line\_candidate1)

elif (line3[x-2]-line3[x-1] > 0):# or (High[x-3]-High[x-1] > 0):

line3[x] = np.min(line\_candidate1)

elif (line3[x-2]-line3[x-1] == 0):

if (line3[x-3]-line3[x-2] < 0):

line3[x] = np.min(line\_candidate1)

else :

line3[x] = np.min(line\_candidate1)

if (line3[x] == line2[x]):

if (line3[x-2]-line3[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line3[x] = np.max(line\_candidate2)

elif (line3[x-2]-line3[x-1] > 0):# or (High[x-3]-High[x-1] > 0):

line3[x] = np.min(line\_candidate2)

elif (line3[x-2]-line3[x-1] == 0):

if (line3[x-3]-line3[x-2] < 0):

line3[x] = np.min(line\_candidate2)

else :

line3[x] = np.min(line\_candidate2)

elif (line2[x-1] - line3[x-1] >=0 ) and (line2[x-1] - line3[x-1] < (len(line\_candidate2) + 7)) and ((np.min(line\_candidate2) - np.max(line\_candidate3)) >= 50) :

#print(line2[x-1] - line3[x-1])

#print(len(line\_candidate1))

if (line1[x-2]-line1[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line1[x] = np.max(line\_candidate1)

elif (line1[x-2]-line1[x-1] > 0):# or (High[x-3]-High[x-1] > 0):

line1[x] = np.min(line\_candidate1)

elif (line1[x-2]-line1[x-1] == 0):

if (line1[x-3]-line1[x-2] < 0):

line1[x] = np.max(line\_candidate1)

elif (line1[x-3]-line1[x-2] > 0):

line1[x] = np.min(line\_candidate1)

else :

line1[x] = np.min(line\_candidate1)

line2[x] = np.min(line\_candidate2)

line3[x] = np.max(line\_candidate2)

else:

if (line1[x-2]-line1[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line1[x] = np.max(line\_candidate1)

elif (line1[x-2]-line1[x-1] > 0):# or (High[x-3]-High[x-1] > 0):

line1[x] = np.min(line\_candidate1)

elif (line1[x-2]-line1[x-1] == 0):

if (line1[x-3]-line1[x-2] < 0):

line1[x] = np.max(line\_candidate1)

elif (line1[x-3]-line1[x-2] > 0):

line1[x] = np.min(line\_candidate1)

else :

line1[x] = np.min(line\_candidate1)

if (line2[x-2]-line2[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line2[x] = np.max(line\_candidate2)

elif (line2[x-2]-line2[x-1] > 0):# or (High[x-3]-High[x-1] > 0):

line2[x] = np.min(line\_candidate2)

elif (line2[x-2]-line2[x-1] == 0):

if (line2[x-3]-line2[x-2] < 0):

line2[x] = np.max(line\_candidate2)

elif (line2[x-3]-line2[x-2] > 0):

line2[x] = np.min(line\_candidate2)

else :

line2[x] = np.min(line\_candidate2)

if (line3[x-2]-line3[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line3[x] = np.max(line\_candidate3)

elif (line3[x-2]-line3[x-1] > 0):# or (High[x-3]-High[x-1] > 0):

line3[x] = np.min(line\_candidate3)

elif (line3[x-2]-line3[x-1] == 0):

if (line3[x-3]-line3[x-2] < 0):

line3[x] = np.max(line\_candidate3)

elif (line3[x-3]-line3[x-2] > 0):

line3[x] = np.min(line\_candidate3)

else :

line3[x] = np.min(line\_candidate3)

#if (line2[x] == np.max(line\_candidate1)) or (line2[x] == np.min(line\_candidate1)) or

if (line3[x-1] >= line3[x] + 120 ):

if (line3[x-2]-line3[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line3[x] = np.max(line\_candidate2)

elif (line3[x-2]-line3[x-1] > 0):# or (High[x-3]-High[x-1] > 0):

line3[x] = np.min(line\_candidate2)

elif (line3[x-2]-line3[x-1] == 0):

if (line3[x-3]-line3[x-2] < 0):

line3[x] = np.min(line\_candidate2)

else :

line3[x] = np.min(line\_candidate2)

#if (line1[x] == 669 - 181) or

if (line2[x-1] >= line2[x] + 120 ):

if (line2[x-2]-line2[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line2[x] = np.max(line\_candidate1)

elif (line2[x-2]-line2[x-1] > 0):# or (High[x-3]-High[x-1] > 0):

line2[x] = np.min(line\_candidate1)

elif (line2[x-2]-line2[x-1] == 0):

if (line2[x-3]-line2[x-2] < 0):

line2[x] = np.min(line\_candidate1)

else :

line2[x] = np.min(line\_candidate1)

if (line2[x] == line1[x]):

if (line2[x-2]-line2[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line2[x] = np.max(line\_candidate2)

elif (line2[x-2]-line2[x-1] > 0):# or (High[x-3]-High[x-1] > 0):

line2[x] = np.min(line\_candidate2)

elif (line2[x-2]-line2[x-1] == 0):

if (line2[x-3]-line2[x-2] < 0):

line2[x] = np.min(line\_candidate2)

else :

line2[x] = np.min(line\_candidate2)

if (line3[x] == line2[x]):

if (line3[x-2]-line3[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line3[x] = np.max(line\_candidate3)

elif (line3[x-2]-line3[x-1] > 0):# or (High[x-3]-High[x-1] > 0):

line3[x] = np.min(line\_candidate3)

elif (line3[x-2]-line3[x-1] == 0):

if (line3[x-3]-line3[x-2] < 0):

line3[x] = np.min(line\_candidate3)

else :

line3[x] = np.min(line\_candidate3)

before\_count\_up = 0

before\_count\_down = 0

line\_extreme\_up = 0

line\_extreme\_down = 0

line = 0

tem\_counts1 = 0

tem\_counts2 = 0

tem\_counts3 = 0

line\_candidate1 = []

line\_candidate2 = []

line\_candidate3 = []

#sio.savemat("line1.mat",{"line1":line1})

#sio.savemat("line2.mat",{"line2":line2})

#sio.savemat("line3.mat",{"line3":line3})

high[0] = line1

high[1] = line2

high[2] = line3

##第二列##

#ECG = im

line = 0

line\_extreme\_up = 0

#line\_extreme\_down = 0

before\_count\_up = 0

no\_area = 0

#before\_count\_down = 0

line1\_extreme = []

line3\_extreme = []

line\_candidate1 = []

line\_candidate2 = []

line\_candidate3 = []

line1 = np.zeros(shape = 240,dtype=int)

line2 = np.zeros(shape = 240,dtype=int)

line3 = np.zeros(shape = 240,dtype=int)

tem\_counts1 = 0

tem\_counts2 = 0

tem\_counts3 = 0

for j in range(165 + 300 ,165 + 300 + 240):

for i in range (181,670):

if ECG[i][j] == 0 and line == 0:

#print(str(j)+ " " + str(i))

line\_candidate1.append(669 - i)

if i == 181:

if before\_count\_up == 0:

line\_extreme\_up = 1

line1\_extreme.append(j)

else :

line\_extreme\_up = 1

before\_count\_up = 1

'''

if len(line\_candidate1) < 1:

line\_candidate1.append(635 - i)

elif (line\_candidate1[len(line\_candidate1)-1] - (635 - i)) == 1:

line\_candidate1.append(635 - i)

'''

elif (i >= 182 ) and (ECG[i][j] != 0) and (ECG[i-1][j] == 0) and (line == 0) :

line = line + 1

continue

if ECG[i][j] == 0 and line == 1:

line\_candidate2.append(669 - i)

'''

if len(line\_candidate2) < 1:

line\_candidate2.append(635 - i)

elif (line\_candidate2[len(line\_candidate2)-1] - (635 - i)) == 1:

line\_candidate2.append(635 - i)

'''

elif (ECG[i][j] != 0) and (ECG[i-1][j] == 0) and (line == 1) :

line = line + 1

continue

if ECG[i][j] == 0 and line == 2:

line\_candidate3.append(669 - i)

'''

if len(line\_candidate3) < 1:

line\_candidate3.append(669 - i)

elif (line\_candidate3[len(line\_candidate3)-1] - (669 - i)) == 1:

line\_candidate3.append(669 - i)

'''

elif (ECG[i][j] != 0) and (ECG[i-1][j] == 0) and (line == 2):

line = 0

break

x = j - 165 - 300

#print(x)

#print(line\_candidate1)

#print(line\_candidate2)

#print(line\_candidate3)

tem\_counts1 = np.bincount(line1)

tem\_counts2 = np.bincount(line2)

tem\_counts3 = np.bincount(line3)

#返回众数

tem\_counts1 = np.argmax(tem\_counts1)

tem\_counts2 = np.argmax(tem\_counts2)

tem\_counts3 = np.argmax(tem\_counts3)

if line\_candidate2 == []:

line\_candidate2 = line\_candidate1

line\_candidate3 = line\_candidate1

elif line\_candidate3 == []:

line\_candidate3 = line\_candidate2

if ((x == 0) or (x == 1)) :

if (line\_extreme\_up == 1 and before\_count\_up == 0): #and (begin == 0) :

line1[x] = np.max(line\_candidate1)

line2[x] = np.min(line\_candidate2)

line3[x] = np.min(line\_candidate3)

before\_count\_up = 1

#before\_count\_down = 0

line\_extreme\_up = 0

#line\_extreme\_down = 0

#begin = 1

elif (line\_extreme\_up == 0 and before\_count\_up == 1): #and (begin == 1):

line1[x] = 669 - 181

line2[x] = np.min(line\_candidate1)

line3[x] = np.min(line\_candidate2)

before\_count\_up = 1

before\_count\_down = 0

line\_extreme\_up = 0

line\_extreme\_down = 0

elif (line\_extreme\_up == 1 and before\_count\_up == 1): #and (begin == 1) :

line1[x] = np.max(line\_candidate1)

line2[x] = np.min(line\_candidate2)

line3[x] = np.min(line\_candidate3)

before\_count\_up = 0

#before\_count\_down = 0

line\_extreme\_up = 0

#line\_extreme\_down = 0

#begin = 0

elif (line\_extreme\_up == 0 and before\_count\_up == 0):

line1[x] = np.min(line\_candidate1)

line2[x] = np.min(line\_candidate2)

line3[x] = np.min(line\_candidate3)

else:

if (line\_extreme\_up == 1 and before\_count\_up == 0):

line1[x] = 669 - 181

if (line2[x-2]-line2[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line2[x] = np.max(line\_candidate2)

elif (line2[x-2]-line2[x-1] > 0):# or (High[x-3]-High[x-1] > 0):

line2[x] = np.min(line\_candidate2)

elif (line2[x-2]-line2[x-1] == 0):

if (line2[x-3]-line2[x-2] < 0):

line2[x] = np.max(line\_candidate2)

elif (line2[x-3]-line2[x-2] > 0):

line2[x] = np.min(line\_candidate2)

else :

line2[x] = np.min(line\_candidate2)

if (line3[x-2]-line3[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line3[x] = np.max(line\_candidate3)

elif (line3[x-2]-line3[x-1] > 0):

line3[x] = np.min(line\_candidate3)

elif (line3[x-2]-line3[x-1] == 0):

if (line3[x-3]-line3[x-2] < 0):

line3[x] = np.max(line\_candidate3)

elif (line3[x-3]-line3[x-2] > 0):

line3[x] = np.min(line\_candidate3)

else :

line3[x] = np.min(line\_candidate3)

if (line3[x-1] >= line3[x] + 120):

if (line3[x-2]-line3[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line3[x] = np.max(line\_candidate2)

elif (line3[x-2]-line3[x-1] > 0):# or (High[x-3]-High[x-1] > 0):

line3[x] = np.min(line\_candidate2)

elif (line3[x-2]-line3[x-1] == 0):

if (line3[x-3]-line3[x-2] < 0):

line3[x] = np.min(line\_candidate2)

else :

line3[x] = np.min(line\_candidate2)

#if line1[x] == (669 - 181 - 25) or

if (line2[x-1] >= line2[x] + 120):

if (line2[x-2]-line2[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line2[x] = np.max(line\_candidate1)

elif (line2[x-2]-line2[x-1] > 0):# or (High[x-3]-High[x-1] > 0):

line2[x] = np.min(line\_candidate1)

elif (line2[x-2]-line2[x-1] == 0):

if (line2[x-3]-line2[x-2] < 0):

line2[x] = np.min(line\_candidate1)

else :

line2[x] = np.min(line\_candidate1)

if (line2[x] == line1[x]):

if (line2[x-2]-line2[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line2[x] = np.max(line\_candidate2)

elif (line2[x-2]-line2[x-1] > 0):# or (High[x-3]-High[x-1] > 0):

line2[x] = np.min(line\_candidate2)

elif (line2[x-2]-line2[x-1] == 0):

if (line2[x-3]-line2[x-2] < 0):

line2[x] = np.min(line\_candidate2)

else :

line2[x] = np.min(line\_candidate2)

if (line3[x] == line2[x]):

if (line3[x-2]-line3[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line3[x] = np.max(line\_candidate3)

elif (line3[x-2]-line3[x-1] > 0):# or (High[x-3]-High[x-1] > 0):

line3[x] = np.min(line\_candidate3)

elif (line3[x-2]-line3[x-1] == 0):

if (line3[x-3]-line3[x-2] < 0):

line3[x] = np.min(line\_candidate3)

else :

line3[x] = np.min(line\_candidate3)

before\_count\_up = 1

#before\_count\_down = 1

line\_extreme\_up = 0

#line\_extreme\_down = 0

elif (line\_extreme\_up == 0 and before\_count\_up == 1):

line1[x] = 669 - 181

if (line2[x-2]-line2[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line2[x] = np.max(line\_candidate1)

elif (line2[x-2]-line2[x-1] > 0):# or (High[x-3]-High[x-1] > 0):

line2[x] = np.min(line\_candidate1)

elif (line2[x-2]-line2[x-1] == 0):

if (line2[x-3]-line2[x-2] < 0):

line2[x] = np.max(line\_candidate1)

elif (line2[x-3]-line2[x-2] > 0):

line2[x] = np.min(line\_candidate1)

else :

line2[x] = np.min(line\_candidate1)

if (line3[x-2]-line3[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line3[x] = np.max(line\_candidate2)

elif (line3[x-2]-line3[x-1] > 0):# or (High[x-3]-High[x-1] > 0):

line3[x] = np.min(line\_candidate2)

elif (line3[x-2]-line3[x-1] == 0):

if (line3[x-3]-line3[x-2] < 0):

line3[x] = np.max(line\_candidate2)

elif (line3[x-3]-line3[x-2] > 0):

line3[x] = np.min(line\_candidate2)

else :

line3[x] = np.min(line\_candidate2)

if (line3[x-1] >= line3[x] + 120):

if (line3[x-2]-line3[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line3[x] = np.max(line\_candidate1)

elif (line3[x-2]-line3[x-1] > 0):# or (High[x-3]-High[x-1] > 0):

line3[x] = np.min(line\_candidate1)

elif (line3[x-2]-line3[x-1] == 0):

if (line3[x-3]-line3[x-2] < 0):

line3[x] = np.min(line\_candidate1)

else :

line3[x] = np.min(line\_candidate1)

'''

if (line2[x] == line1[x]):

if (line2[x-2]-line2[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line2[x] = np.max(line\_candidate2)

elif (line2[x-2]-line2[x-1] > 0):# or (High[x-3]-High[x-1] > 0):

line2[x] = np.min(line\_candidate2)

elif (line2[x-2]-line2[x-1] == 0):

if (line2[x-3]-line2[x-2] < 0):

line2[x] = np.min(line\_candidate2)

else :

line2[x] = np.min(line\_candidate2)

'''

if (line3[x] == line2[x]):

if (line3[x-2]-line3[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line3[x] = np.max(line\_candidate2)

elif (line3[x-2]-line3[x-1] > 0):# or (High[x-3]-High[x-1] > 0):

line3[x] = np.min(line\_candidate2)

elif (line3[x-2]-line3[x-1] == 0):

if (line3[x-3]-line3[x-2] < 0):

line3[x] = np.min(line\_candidate2)

else :

line3[x] = np.min(line\_candidate2)

before\_count\_up = 1

#before\_count\_down = 0

line\_extreme\_up = 0

#line\_extreme\_down = 0

elif (line\_extreme\_up == 1 and before\_count\_up == 1):

line1[x] = 669 - 181

if (line2[x-2]-line2[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line2[x] = np.max(line\_candidate2)

elif (line2[x-2]-line2[x-1] > 0):# or (High[x-3]-High[x-1] > 0):

line2[x] = np.min(line\_candidate2)

elif (line2[x-2]-line2[x-1] == 0):

if (line2[x-3]-line2[x-2] < 0):

line2[x] = np.max(line\_candidate2)

elif (line2[x-3]-line2[x-2] > 0):

line2[x] = np.min(line\_candidate2)

else :

line2[x] = np.min(line\_candidate2)

if (line3[x-2]-line3[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line3[x] = np.max(line\_candidate3)

elif (line3[x-2]-line3[x-1] > 0):# or (High[x-3]-High[x-1] > 0):

line3[x] = np.min(line\_candidate3)

elif (line3[x-2]-line3[x-1] == 0):

if (line3[x-3]-line3[x-2] < 0):

line3[x] = np.max(line\_candidate3)

elif (line3[x-3]-line3[x-2] > 0):

line3[x] = np.min(line\_candidate3)

else :

line3[x] = np.min(line\_candidate3)

if (line3[x-1] >= line3[x] + 120):

if (line3[x-2]-line3[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line3[x] = np.max(line\_candidate2)

elif (line3[x-2]-line3[x-1] > 0):# or (High[x-3]-High[x-1] > 0):

line3[x] = np.min(line\_candidate2)

elif (line3[x-2]-line3[x-1] == 0):

if (line3[x-3]-line3[x-2] < 0):

line3[x] = np.min(line\_candidate2)

else :

line3[x] = np.min(line\_candidate2)

#if line1[x] == (669 - 181 - 25) or

if (line2[x-1] >= line2[x] + 120):

if (line2[x-2]-line2[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line2[x] = np.max(line\_candidate1)

elif (line2[x-2]-line2[x-1] > 0):# or (High[x-3]-High[x-1] > 0):

line2[x] = np.min(line\_candidate1)

elif (line2[x-2]-line2[x-1] == 0):

if (line2[x-3]-line2[x-2] < 0):

line2[x] = np.min(line\_candidate1)

else :

line2[x] = np.min(line\_candidate1)

if (line2[x] == line1[x]):

if (line2[x-2]-line2[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line2[x] = np.max(line\_candidate2)

elif (line2[x-2]-line2[x-1] > 0):# or (High[x-3]-High[x-1] > 0):

line2[x] = np.min(line\_candidate2)

elif (line2[x-2]-line2[x-1] == 0):

if (line2[x-3]-line2[x-2] < 0):

line2[x] = np.min(line\_candidate2)

else :

line2[x] = np.min(line\_candidate2)

if (line3[x] == line2[x]):

if (line3[x-2]-line3[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line3[x] = np.max(line\_candidate3)

elif (line3[x-2]-line3[x-1] > 0):# or (High[x-3]-High[x-1] > 0):

line3[x] = np.min(line\_candidate3)

elif (line3[x-2]-line3[x-1] == 0):

if (line3[x-3]-line3[x-2] < 0):

line3[x] = np.min(line\_candidate3)

else :

line3[x] = np.min(line\_candidate3)

line\_extreme\_up = 0

#line\_extreme\_down = 0

before\_count\_up = 0

#before\_count\_down = 1

elif (line\_extreme\_up == 0 and before\_count\_up == 0):

if (line1[x-1] - line2[x-1] >=0 ) and (line1[x-1] - line2[x-1] < (len(line\_candidate1) + 7)) and ((np.min(line\_candidate1) - np.max(line\_candidate2)) >= 50) :

line1[x] = np.min(line\_candidate1)

line2[x] = np.max(line\_candidate1)

#print(line\_candidate1)

#print(line\_candidate2)

#print(line\_candidate3)

#print(line1[x-1] - line2[x-1])

#print(len(line\_candidate1))

if (line3[x-2]-line3[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line3[x] = np.max(line\_candidate2)

elif (line3[x-2]-line3[x-1] > 0):# or (High[x-3]-High[x-1] > 0):

line3[x] = np.min(line\_candidate2)

elif (line3[x-2]-line3[x-1] == 0):

if (line3[x-3]-line3[x-2] < 0):

line3[x] = np.max(line\_candidate2)

elif (line3[x-3]-line3[x-2] > 0):

line3[x] = np.min(line\_candidate2)

else :

line3[x] = np.min(line\_candidate2)

#if (line2[x] == np.max(line\_candidate1)) or (line2[x] == np.min(line\_candidate1)) or

if (line3[x-1] >= line3[x] + 120 ):

if (line3[x-2]-line3[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line3[x] = np.max(line\_candidate1)

elif (line3[x-2]-line3[x-1] > 0):# or (High[x-3]-High[x-1] > 0):

line3[x] = np.min(line\_candidate1)

elif (line3[x-2]-line3[x-1] == 0):

if (line3[x-3]-line3[x-2] < 0):

line3[x] = np.max(line\_candidate1)

elif (line3[x-3]-line3[x-2] > 0):

line3[x] = np.min(line\_candidate1)

else :

line3[x] = np.min(line\_candidate1)

if (line3[x] == line2[x]):

if (line3[x-2]-line3[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line3[x] = np.max(line\_candidate2)

elif (line3[x-2]-line3[x-1] > 0):# or (High[x-3]-High[x-1] > 0):

line3[x] = np.min(line\_candidate2)

elif (line3[x-2]-line3[x-1] == 0):

if (line3[x-3]-line3[x-2] < 0):

line3[x] = np.min(line\_candidate2)

else :

line3[x] = np.min(line\_candidate2)

elif (line2[x-1] - line3[x-1] >=0 ) and (line2[x-1] - line3[x-1] < (len(line\_candidate2) + 7)) and ((np.min(line\_candidate2) - np.max(line\_candidate3)) >= 50) :

#print(line2[x-1] - line3[x-1])

#print(len(line\_candidate1))

if (line1[x-2]-line1[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line1[x] = np.max(line\_candidate1)

elif (line1[x-2]-line1[x-1] > 0):# or (High[x-3]-High[x-1] > 0):

line1[x] = np.min(line\_candidate1)

elif (line1[x-2]-line1[x-1] == 0):

if (line1[x-3]-line1[x-2] < 0):

line1[x] = np.max(line\_candidate1)

elif (line1[x-3]-line1[x-2] > 0):

line1[x] = np.min(line\_candidate1)

else :

line1[x] = np.min(line\_candidate1)

line2[x] = np.min(line\_candidate2)

line3[x] = np.max(line\_candidate2)

else:

if (line1[x-2]-line1[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line1[x] = np.max(line\_candidate1)

elif (line1[x-2]-line1[x-1] > 0):# or (High[x-3]-High[x-1] > 0):

line1[x] = np.min(line\_candidate1)

elif (line1[x-2]-line1[x-1] == 0):

if (line1[x-3]-line1[x-2] < 0):

line1[x] = np.max(line\_candidate1)

elif (line1[x-3]-line1[x-2] > 0):

line1[x] = np.min(line\_candidate1)

else :

line1[x] = np.min(line\_candidate1)

if (line2[x-2]-line2[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line2[x] = np.max(line\_candidate2)

elif (line2[x-2]-line2[x-1] > 0):# or (High[x-3]-High[x-1] > 0):

line2[x] = np.min(line\_candidate2)

elif (line2[x-2]-line2[x-1] == 0):

if (line2[x-3]-line2[x-2] < 0):

line2[x] = np.max(line\_candidate2)

elif (line2[x-3]-line2[x-2] > 0):

line2[x] = np.min(line\_candidate2)

else :

line2[x] = np.min(line\_candidate2)

if (line3[x-2]-line3[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line3[x] = np.max(line\_candidate3)

elif (line3[x-2]-line3[x-1] > 0):# or (High[x-3]-High[x-1] > 0):

line3[x] = np.min(line\_candidate3)

elif (line3[x-2]-line3[x-1] == 0):

if (line3[x-3]-line3[x-2] < 0):

line3[x] = np.max(line\_candidate3)

elif (line3[x-3]-line3[x-2] > 0):

line3[x] = np.min(line\_candidate3)

else :

line3[x] = np.min(line\_candidate3)

#if (line2[x] == np.max(line\_candidate1)) or (line2[x] == np.min(line\_candidate1)) or

if (line3[x-1] >= line3[x] + 120 ):

if (line3[x-2]-line3[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line3[x] = np.max(line\_candidate2)

elif (line3[x-2]-line3[x-1] > 0):# or (High[x-3]-High[x-1] > 0):

line3[x] = np.min(line\_candidate2)

elif (line3[x-2]-line3[x-1] == 0):

if (line3[x-3]-line3[x-2] < 0):

line3[x] = np.min(line\_candidate2)

else :

line3[x] = np.min(line\_candidate2)

#if (line1[x] == 669 - 181) or

if (line2[x-1] >= line2[x] + 120 ):

if (line2[x-2]-line2[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line2[x] = np.max(line\_candidate1)

elif (line2[x-2]-line2[x-1] > 0):# or (High[x-3]-High[x-1] > 0):

line2[x] = np.min(line\_candidate1)

elif (line2[x-2]-line2[x-1] == 0):

if (line2[x-3]-line2[x-2] < 0):

line2[x] = np.min(line\_candidate1)

else :

line2[x] = np.min(line\_candidate1)

if (line2[x] == line1[x]):

if (line2[x-2]-line2[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line2[x] = np.max(line\_candidate2)

elif (line2[x-2]-line2[x-1] > 0):# or (High[x-3]-High[x-1] > 0):

line2[x] = np.min(line\_candidate2)

elif (line2[x-2]-line2[x-1] == 0):

if (line2[x-3]-line2[x-2] < 0):

line2[x] = np.min(line\_candidate2)

else :

line2[x] = np.min(line\_candidate2)

if (line3[x] == line2[x]):

if (line3[x-2]-line3[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line3[x] = np.max(line\_candidate3)

elif (line3[x-2]-line3[x-1] > 0):# or (High[x-3]-High[x-1] > 0):

line3[x] = np.min(line\_candidate3)

elif (line3[x-2]-line3[x-1] == 0):

if (line3[x-3]-line3[x-2] < 0):

line3[x] = np.min(line\_candidate3)

else :

line3[x] = np.min(line\_candidate3)

before\_count\_up = 0

before\_count\_down = 0

line\_extreme\_up = 0

line\_extreme\_down = 0

line = 0

tem\_counts1 = 0

tem\_counts2 = 0

tem\_counts3 = 0

line\_candidate1 = []

line\_candidate2 = []

line\_candidate3 = []

high[3] = line1

high[4] = line2

high[5] = line3

#sio.savemat("line4.mat",{"line4":line1})

#sio.savemat("line5.mat",{"line5":line2})

#sio.savemat("line6.mat",{"line6":line3})

##第三列##

#ECG = im

line = 0

line\_extreme\_up = 0

#line\_extreme\_down = 0

before\_count\_up = 0

no\_area = 0

#before\_count\_down = 0

line1\_extreme = []

line3\_extreme = []

line\_candidate1 = []

line\_candidate2 = []

line\_candidate3 = []

line1 = np.zeros(shape = 240,dtype=int)

line2 = np.zeros(shape = 240,dtype=int)

line3 = np.zeros(shape = 240,dtype=int)

tem\_counts1 = 0

tem\_counts2 = 0

tem\_counts3 = 0

for j in range(165 + 300 \* 2 ,165 + 300 \* 2 + 240):

for i in range (181,670):

if ECG[i][j] == 0 and line == 0:

#print(str(j)+ " " + str(i))

line\_candidate1.append(669 - i)

if i == 181:

if before\_count\_up == 0:

line\_extreme\_up = 1

line1\_extreme.append(j)

else :

line\_extreme\_up = 1

before\_count\_up = 1

'''

if len(line\_candidate1) < 1:

line\_candidate1.append(635 - i)

elif (line\_candidate1[len(line\_candidate1)-1] - (635 - i)) == 1:

line\_candidate1.append(635 - i)

'''

elif (i >= 182 ) and(ECG[i][j] != 0) and (ECG[i-1][j] == 0) and (line == 0) :

line = line + 1

continue

if ECG[i][j] == 0 and line == 1:

line\_candidate2.append(669 - i)

'''

if len(line\_candidate2) < 1:

line\_candidate2.append(635 - i)

elif (line\_candidate2[len(line\_candidate2)-1] - (635 - i)) == 1:

line\_candidate2.append(635 - i)

'''

elif (ECG[i][j] != 0) and (ECG[i-1][j] == 0) and (line == 1) :

line = line + 1

continue

if ECG[i][j] == 0 and line == 2:

line\_candidate3.append(669 - i)

'''

if len(line\_candidate3) < 1:

line\_candidate3.append(669 - i)

elif (line\_candidate3[len(line\_candidate3)-1] - (669 - i)) == 1:

line\_candidate3.append(669 - i)

'''

elif (ECG[i][j] != 0) and (ECG[i-1][j] == 0) and (line == 2):

line = 0

break

x = j - 165 - 300 \* 2

#print(x)

#print(line\_candidate1)

#print(line\_candidate2)

#print(line\_candidate3)

tem\_counts1 = np.bincount(line1)

tem\_counts2 = np.bincount(line2)

tem\_counts3 = np.bincount(line3)

#返回众数

tem\_counts1 = np.argmax(tem\_counts1)

tem\_counts2 = np.argmax(tem\_counts2)

tem\_counts3 = np.argmax(tem\_counts3)

if line\_candidate2 == []:

line\_candidate2 = line\_candidate1

line\_candidate3 = line\_candidate1

elif line\_candidate3 == []:

line\_candidate3 = line\_candidate2

if ((x == 0) or (x == 1)) :

if (line\_extreme\_up == 1 and before\_count\_up == 0): #and (begin == 0) :

line1[x] = np.max(line\_candidate1)

line2[x] = np.min(line\_candidate2)

line3[x] = np.min(line\_candidate3)

before\_count\_up = 1

#before\_count\_down = 0

line\_extreme\_up = 0

#line\_extreme\_down = 0

#begin = 1

elif (line\_extreme\_up == 0 and before\_count\_up == 1): #and (begin == 1):

line1[x] = 669 - 181

line2[x] = np.min(line\_candidate1)

line3[x] = np.min(line\_candidate2)

before\_count\_up = 1

before\_count\_down = 0

line\_extreme\_up = 0

line\_extreme\_down = 0

elif (line\_extreme\_up == 1 and before\_count\_up == 1): #and (begin == 1) :

line1[x] = np.max(line\_candidate1)

line2[x] = np.min(line\_candidate2)

line3[x] = np.min(line\_candidate3)

before\_count\_up = 0

#before\_count\_down = 0

line\_extreme\_up = 0

#line\_extreme\_down = 0

#begin = 0

elif (line\_extreme\_up == 0 and before\_count\_up == 0):

line1[x] = np.min(line\_candidate1)

line2[x] = np.min(line\_candidate2)

line3[x] = np.min(line\_candidate3)

else:

if (line\_extreme\_up == 1 and before\_count\_up == 0):

line1[x] = 669 - 181

if (line2[x-2]-line2[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line2[x] = np.max(line\_candidate2)

elif (line2[x-2]-line2[x-1] > 0):# or (High[x-3]-High[x-1] > 0):

line2[x] = np.min(line\_candidate2)

elif (line2[x-2]-line2[x-1] == 0):

if (line2[x-3]-line2[x-2] < 0):

line2[x] = np.max(line\_candidate2)

elif (line2[x-3]-line2[x-2] > 0):

line2[x] = np.min(line\_candidate2)

else :

line2[x] = np.min(line\_candidate2)

if (line3[x-2]-line3[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line3[x] = np.max(line\_candidate3)

elif (line3[x-2]-line3[x-1] > 0):

line3[x] = np.min(line\_candidate3)

elif (line3[x-2]-line3[x-1] == 0):

if (line3[x-3]-line3[x-2] < 0):

line3[x] = np.max(line\_candidate3)

elif (line3[x-3]-line3[x-2] > 0):

line3[x] = np.min(line\_candidate3)

else :

line3[x] = np.min(line\_candidate3)

if (line3[x-1] >= line3[x] + 120):

if (line3[x-2]-line3[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line3[x] = np.max(line\_candidate2)

elif (line3[x-2]-line3[x-1] > 0):# or (High[x-3]-High[x-1] > 0):

line3[x] = np.min(line\_candidate2)

elif (line3[x-2]-line3[x-1] == 0):

if (line3[x-3]-line3[x-2] < 0):

line3[x] = np.min(line\_candidate2)

else :

line3[x] = np.min(line\_candidate2)

#if line1[x] == (669 - 181 - 25) or

if (line2[x-1] >= line2[x] + 120):

if (line2[x-2]-line2[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line2[x] = np.max(line\_candidate1)

elif (line2[x-2]-line2[x-1] > 0):# or (High[x-3]-High[x-1] > 0):

line2[x] = np.min(line\_candidate1)

elif (line2[x-2]-line2[x-1] == 0):

if (line2[x-3]-line2[x-2] < 0):

line2[x] = np.min(line\_candidate1)

else :

line2[x] = np.min(line\_candidate1)

if (line2[x] == line1[x]):

if (line2[x-2]-line2[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line2[x] = np.max(line\_candidate2)

elif (line2[x-2]-line2[x-1] > 0):# or (High[x-3]-High[x-1] > 0):

line2[x] = np.min(line\_candidate2)

elif (line2[x-2]-line2[x-1] == 0):

if (line2[x-3]-line2[x-2] < 0):

line2[x] = np.min(line\_candidate2)

else :

line2[x] = np.min(line\_candidate2)

if (line3[x] == line2[x]):

if (line3[x-2]-line3[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line3[x] = np.max(line\_candidate3)

elif (line3[x-2]-line3[x-1] > 0):# or (High[x-3]-High[x-1] > 0):

line3[x] = np.min(line\_candidate3)

elif (line3[x-2]-line3[x-1] == 0):

if (line3[x-3]-line3[x-2] < 0):

line3[x] = np.min(line\_candidate3)

else :

line3[x] = np.min(line\_candidate3)

before\_count\_up = 1

#before\_count\_down = 1

line\_extreme\_up = 0

#line\_extreme\_down = 0

elif (line\_extreme\_up == 0 and before\_count\_up == 1):

line1[x] = 669 - 181

if (line2[x-2]-line2[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line2[x] = np.max(line\_candidate1)

elif (line2[x-2]-line2[x-1] > 0):# or (High[x-3]-High[x-1] > 0):

line2[x] = np.min(line\_candidate1)

elif (line2[x-2]-line2[x-1] == 0):

if (line2[x-3]-line2[x-2] < 0):

line2[x] = np.max(line\_candidate1)

elif (line2[x-3]-line2[x-2] > 0):

line2[x] = np.min(line\_candidate1)

else :

line2[x] = np.min(line\_candidate1)

if (line3[x-2]-line3[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line3[x] = np.max(line\_candidate2)

elif (line3[x-2]-line3[x-1] > 0):# or (High[x-3]-High[x-1] > 0):

line3[x] = np.min(line\_candidate2)

elif (line3[x-2]-line3[x-1] == 0):

if (line3[x-3]-line3[x-2] < 0):

line3[x] = np.max(line\_candidate2)

elif (line3[x-3]-line3[x-2] > 0):

line3[x] = np.min(line\_candidate2)

else :

line3[x] = np.min(line\_candidate2)

if (line3[x-1] >= line3[x] + 120):

if (line3[x-2]-line3[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line3[x] = np.max(line\_candidate1)

elif (line3[x-2]-line3[x-1] > 0):# or (High[x-3]-High[x-1] > 0):

line3[x] = np.min(line\_candidate1)

elif (line3[x-2]-line3[x-1] == 0):

if (line3[x-3]-line3[x-2] < 0):

line3[x] = np.min(line\_candidate1)

else :

line3[x] = np.min(line\_candidate1)

'''

if (line2[x] == line1[x]):

if (line2[x-2]-line2[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line2[x] = np.max(line\_candidate2)

elif (line2[x-2]-line2[x-1] > 0):# or (High[x-3]-High[x-1] > 0):

line2[x] = np.min(line\_candidate2)

elif (line2[x-2]-line2[x-1] == 0):

if (line2[x-3]-line2[x-2] < 0):

line2[x] = np.min(line\_candidate2)

else :

line2[x] = np.min(line\_candidate2)

'''

if (line3[x] == line2[x]):

if (line3[x-2]-line3[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line3[x] = np.max(line\_candidate2)

elif (line3[x-2]-line3[x-1] > 0):# or (High[x-3]-High[x-1] > 0):

line3[x] = np.min(line\_candidate2)

elif (line3[x-2]-line3[x-1] == 0):

if (line3[x-3]-line3[x-2] < 0):

line3[x] = np.min(line\_candidate2)

else :

line3[x] = np.min(line\_candidate2)

before\_count\_up = 1

#before\_count\_down = 0

line\_extreme\_up = 0

#line\_extreme\_down = 0

elif (line\_extreme\_up == 1 and before\_count\_up == 1):

line1[x] = 669 - 181

if (line2[x-2]-line2[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line2[x] = np.max(line\_candidate2)

elif (line2[x-2]-line2[x-1] > 0):# or (High[x-3]-High[x-1] > 0):

line2[x] = np.min(line\_candidate2)

elif (line2[x-2]-line2[x-1] == 0):

if (line2[x-3]-line2[x-2] < 0):

line2[x] = np.max(line\_candidate2)

elif (line2[x-3]-line2[x-2] > 0):

line2[x] = np.min(line\_candidate2)

else :

line2[x] = np.min(line\_candidate2)

if (line3[x-2]-line3[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line3[x] = np.max(line\_candidate3)

elif (line3[x-2]-line3[x-1] > 0):# or (High[x-3]-High[x-1] > 0):

line3[x] = np.min(line\_candidate3)

elif (line3[x-2]-line3[x-1] == 0):

if (line3[x-3]-line3[x-2] < 0):

line3[x] = np.max(line\_candidate3)

elif (line3[x-3]-line3[x-2] > 0):

line3[x] = np.min(line\_candidate3)

else :

line3[x] = np.min(line\_candidate3)

if (line3[x-1] >= line3[x] + 120):

if (line3[x-2]-line3[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line3[x] = np.max(line\_candidate2)

elif (line3[x-2]-line3[x-1] > 0):# or (High[x-3]-High[x-1] > 0):

line3[x] = np.min(line\_candidate2)

elif (line3[x-2]-line3[x-1] == 0):

if (line3[x-3]-line3[x-2] < 0):

line3[x] = np.min(line\_candidate2)

else :

line3[x] = np.min(line\_candidate2)

#if line1[x] == (669 - 181 - 25) or

if (line2[x-1] >= line2[x] + 120):

if (line2[x-2]-line2[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line2[x] = np.max(line\_candidate1)

elif (line2[x-2]-line2[x-1] > 0):# or (High[x-3]-High[x-1] > 0):

line2[x] = np.min(line\_candidate1)

elif (line2[x-2]-line2[x-1] == 0):

if (line2[x-3]-line2[x-2] < 0):

line2[x] = np.min(line\_candidate1)

else :

line2[x] = np.min(line\_candidate1)

if (line2[x] == line1[x]):

if (line2[x-2]-line2[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line2[x] = np.max(line\_candidate2)

elif (line2[x-2]-line2[x-1] > 0):# or (High[x-3]-High[x-1] > 0):

line2[x] = np.min(line\_candidate2)

elif (line2[x-2]-line2[x-1] == 0):

if (line2[x-3]-line2[x-2] < 0):

line2[x] = np.min(line\_candidate2)

else :

line2[x] = np.min(line\_candidate2)

if (line3[x] == line2[x]):

if (line3[x-2]-line3[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line3[x] = np.max(line\_candidate3)

elif (line3[x-2]-line3[x-1] > 0):# or (High[x-3]-High[x-1] > 0):

line3[x] = np.min(line\_candidate3)

elif (line3[x-2]-line3[x-1] == 0):

if (line3[x-3]-line3[x-2] < 0):

line3[x] = np.min(line\_candidate3)

else :

line3[x] = np.min(line\_candidate3)

line\_extreme\_up = 0

#line\_extreme\_down = 0

before\_count\_up = 0

#before\_count\_down = 1

elif (line\_extreme\_up == 0 and before\_count\_up == 0):

if (line1[x-1] - line2[x-1] >=0 ) and (line1[x-1] - line2[x-1] < (len(line\_candidate1) + 7)) and ((np.min(line\_candidate1) - np.max(line\_candidate2)) >= 50) :

line1[x] = np.min(line\_candidate1)

line2[x] = np.max(line\_candidate1)

#print(line1[x-1] - line2[x-1])

#print(len(line\_candidate1))

if (line3[x-2]-line3[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line3[x] = np.max(line\_candidate2)

elif (line3[x-2]-line3[x-1] > 0):# or (High[x-3]-High[x-1] > 0):

line3[x] = np.min(line\_candidate2)

elif (line3[x-2]-line3[x-1] == 0):

if (line3[x-3]-line3[x-2] < 0):

line3[x] = np.max(line\_candidate2)

elif (line3[x-3]-line3[x-2] > 0):

line3[x] = np.min(line\_candidate2)

else :

line3[x] = np.min(line\_candidate2)

if (line3[x-1] >= line3[x] + 120):

if (line3[x-2]-line3[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line3[x] = np.max(line\_candidate1)

elif (line3[x-2]-line3[x-1] > 0):# or (High[x-3]-High[x-1] > 0):

line3[x] = np.min(line\_candidate1)

elif (line3[x-2]-line3[x-1] == 0):

if (line3[x-3]-line3[x-2] > 0):

line3[x] = np.min(line\_candidate1)

else :

line3[x] = np.min(line\_candidate1)

if (line3[x] == line2[x]):

if (line3[x-2]-line3[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line3[x] = np.max(line\_candidate2)

elif (line3[x-2]-line3[x-1] > 0):# or (High[x-3]-High[x-1] > 0):

line3[x] = np.min(line\_candidate2)

elif (line3[x-2]-line3[x-1] == 0):

if (line3[x-3]-line3[x-2] < 0):

line3[x] = np.min(line\_candidate2)

else :

line3[x] = np.min(line\_candidate2)

elif (line2[x-1] - line3[x-1] >=0 ) and (line2[x-1] - line3[x-1] < (len(line\_candidate2) + 7)): #and ((np.min(line\_candidate2) - np.max(line\_candidate3)) >= 50) :

#print(line2[x-1] - line3[x-1])

#print(len(line\_candidate1))

if (line1[x-2]-line1[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line1[x] = np.max(line\_candidate1)

elif (line1[x-2]-line1[x-1] > 0):# or (High[x-3]-High[x-1] > 0):

line1[x] = np.min(line\_candidate1)

elif (line1[x-2]-line1[x-1] == 0):

if (line1[x-3]-line1[x-2] < 0):

line1[x] = np.max(line\_candidate1)

elif (line1[x-3]-line1[x-2] > 0):

line1[x] = np.min(line\_candidate1)

else :

line1[x] = np.min(line\_candidate1)

line2[x] = np.min(line\_candidate2)

line3[x] = np.max(line\_candidate2)

else:

if (line1[x-2]-line1[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line1[x] = np.max(line\_candidate1)

elif (line1[x-2]-line1[x-1] > 0):# or (High[x-3]-High[x-1] > 0):

line1[x] = np.min(line\_candidate1)

elif (line1[x-2]-line1[x-1] == 0):

if (line1[x-3]-line1[x-2] < 0):

line1[x] = np.max(line\_candidate1)

elif (line1[x-3]-line1[x-2] > 0):

line1[x] = np.min(line\_candidate1)

else :

line1[x] = np.min(line\_candidate1)

if (line2[x-2]-line2[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line2[x] = np.max(line\_candidate2)

elif (line2[x-2]-line2[x-1] > 0):# or (High[x-3]-High[x-1] > 0):

line2[x] = np.min(line\_candidate2)

elif (line2[x-2]-line2[x-1] == 0):

if (line2[x-3]-line2[x-2] < 0):

line2[x] = np.max(line\_candidate2)

elif (line2[x-3]-line2[x-2] > 0):

line2[x] = np.min(line\_candidate2)

else :

line2[x] = np.min(line\_candidate2)

if (line3[x-2]-line3[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line3[x] = np.max(line\_candidate3)

elif (line3[x-2]-line3[x-1] > 0):# or (High[x-3]-High[x-1] > 0):

line3[x] = np.min(line\_candidate3)

elif (line3[x-2]-line3[x-1] == 0):

if (line3[x-3]-line3[x-2] < 0):

line3[x] = np.max(line\_candidate3)

elif (line3[x-3]-line3[x-2] > 0):

line3[x] = np.min(line\_candidate3)

else :

line3[x] = np.min(line\_candidate3)

#if (line2[x] == np.max(line\_candidate1)) or (line2[x] == np.min(line\_candidate1)) or

if (line3[x-1] >= line3[x] + 120):

if (line3[x-2]-line3[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line3[x] = np.max(line\_candidate2)

elif (line3[x-2]-line3[x-1] > 0):# or (High[x-3]-High[x-1] > 0):

line3[x] = np.min(line\_candidate2)

elif (line3[x-2]-line3[x-1] == 0):

if (line3[x-3]-line3[x-2] > 0):

line3[x] = np.min(line\_candidate2)

else :

line3[x] = np.min(line\_candidate2)

#if line1[x] == (669 - 181 - 25) or

if (line2[x-1] >= line2[x] + 120):

if (line2[x-2]-line2[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line2[x] = np.max(line\_candidate1)

elif (line2[x-2]-line2[x-1] > 0):# or (High[x-3]-High[x-1] > 0):

line2[x] = np.min(line\_candidate1)

elif (line2[x-2]-line2[x-1] == 0):

if (line2[x-3]-line2[x-2] < 0):

line2[x] = np.min(line\_candidate1)

else :

line2[x] = np.min(line\_candidate1)

if (line2[x] == line1[x]):

if (line2[x-2]-line2[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line2[x] = np.max(line\_candidate2)

elif (line2[x-2]-line2[x-1] > 0):# or (High[x-3]-High[x-1] > 0):

line2[x] = np.min(line\_candidate2)

elif (line2[x-2]-line2[x-1] == 0):

if (line2[x-3]-line2[x-2] < 0):

line2[x] = np.min(line\_candidate2)

else :

line2[x] = np.min(line\_candidate2)

if (line3[x] == line2[x]):

if (line3[x-2]-line3[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line3[x] = np.max(line\_candidate3)

elif (line3[x-2]-line3[x-1] > 0):# or (High[x-3]-High[x-1] > 0):

line3[x] = np.min(line\_candidate3)

elif (line3[x-2]-line3[x-1] == 0):

if (line3[x-3]-line3[x-2] < 0):

line3[x] = np.min(line\_candidate3)

else :

line3[x] = np.min(line\_candidate3)

before\_count\_up = 0

before\_count\_down = 0

line\_extreme\_up = 0

line\_extreme\_down = 0

line = 0

tem\_counts1 = 0

tem\_counts2 = 0

tem\_counts3 = 0

line\_candidate1 = []

line\_candidate2 = []

line\_candidate3 = []

#sio.savemat("line7.mat",{"line7":line1})

#sio.savemat("line8.mat",{"line8":line2})

#sio.savemat("line9.mat",{"line9":line3})

high[6] = line1

high[7] = line2

high[8] = line3

##第四列##

#ECG = im

line = 0

line\_extreme\_up = 0

#line\_extreme\_down = 0

before\_count\_up = 0

no\_area = 0

#before\_count\_down = 0

line1\_extreme = []

line3\_extreme = []

line\_candidate1 = []

line\_candidate2 = []

line\_candidate3 = []

line1 = np.zeros(shape = 240,dtype=int)

line2 = np.zeros(shape = 240,dtype=int)

line3 = np.zeros(shape = 240,dtype=int)

tem\_counts1 = 0

tem\_counts2 = 0

tem\_counts3 = 0

for j in range(165 + 300 \* 3 ,165 + 300 \* 3 + 240):

for i in range (181 + 25 ,670):

if ECG[i][j] == 0 and line == 0:

#print(str(j)+ " " + str(i))

line\_candidate1.append(669 - i)

if i == 181 + 25:

if before\_count\_up == 0:

line\_extreme\_up = 1

line1\_extreme.append(j)

else :

line\_extreme\_up = 1

before\_count\_up = 1

'''

if len(line\_candidate1) < 1:

line\_candidate1.append(635 - i)

elif (line\_candidate1[len(line\_candidate1)-1] - (635 - i)) == 1:

line\_candidate1.append(635 - i)

'''

elif (i >= 182 + 25 ) and (ECG[i][j] != 0) and (ECG[i-1][j] == 0) and (line == 0) :

line = line + 1

continue

if ECG[i][j] == 0 and line == 1:

line\_candidate2.append(669 - i)

'''

if len(line\_candidate2) < 1:

line\_candidate2.append(635 - i)

elif (line\_candidate2[len(line\_candidate2)-1] - (635 - i)) == 1:

line\_candidate2.append(635 - i)

'''

elif (ECG[i][j] != 0) and (ECG[i-1][j] == 0) and (line == 1) :

line = line + 1

continue

if ECG[i][j] == 0 and line == 2:

line\_candidate3.append(669 - i)

'''

if len(line\_candidate3) < 1:

line\_candidate3.append(669 - i)

elif (line\_candidate3[len(line\_candidate3)-1] - (669 - i)) == 1:

line\_candidate3.append(669 - i)

'''

elif (ECG[i][j] != 0) and (ECG[i-1][j] == 0) and (line == 2):

line = 0

break

x = j - 165 - 300 \* 3

#print(x)

#print(line\_candidate1)

#print(line\_candidate2)

#print(line\_candidate3)

tem\_counts1 = np.bincount(line1)

tem\_counts2 = np.bincount(line2)

tem\_counts3 = np.bincount(line3)

#返回众数

tem\_counts1 = np.argmax(tem\_counts1)

tem\_counts2 = np.argmax(tem\_counts2)

tem\_counts3 = np.argmax(tem\_counts3)

if line\_candidate2 == []:

line\_candidate2 = line\_candidate1

line\_candidate3 = line\_candidate1

elif line\_candidate3 == []:

line\_candidate3 = line\_candidate2

if ((x == 0) or (x == 1)) :

if (line\_extreme\_up == 1 and before\_count\_up == 0): #and (begin == 0) :

line1[x] = np.max(line\_candidate1)

line2[x] = np.min(line\_candidate2)

line3[x] = np.min(line\_candidate3)

before\_count\_up = 1

#before\_count\_down = 0

line\_extreme\_up = 0

#line\_extreme\_down = 0

#begin = 1

elif (line\_extreme\_up == 0 and before\_count\_up == 1): #and (begin == 1):

line1[x] = 669 - 181 - 25

line2[x] = np.min(line\_candidate1)

line3[x] = np.min(line\_candidate2)

before\_count\_up = 1

#before\_count\_down = 0

line\_extreme\_up = 0

#line\_extreme\_down = 0

elif (line\_extreme\_up == 1 and before\_count\_up == 1): #and (begin == 1) :

line1[x] = np.max(line\_candidate1)

line2[x] = np.min(line\_candidate2)

line3[x] = np.min(line\_candidate3)

before\_count\_up = 0

#before\_count\_down = 0

line\_extreme\_up = 0

#line\_extreme\_down = 0

#begin = 0

elif (line\_extreme\_up == 0 and before\_count\_up == 0):

line1[x] = np.min(line\_candidate1)

line2[x] = np.min(line\_candidate2)

line3[x] = np.min(line\_candidate3)

else :

if (line\_extreme\_up == 1 and before\_count\_up == 0):

line1[x] = 669 - 181 - 25

if (line2[x-2]-line2[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line2[x] = np.max(line\_candidate2)

elif (line2[x-2]-line2[x-1] > 0):# or (High[x-3]-High[x-1] > 0):

line2[x] = np.min(line\_candidate2)

elif (line2[x-2]-line2[x-1] == 0):

if (line2[x-3]-line2[x-2] < 0):

line2[x] = np.max(line\_candidate2)

elif (line2[x-3]-line2[x-2] > 0):

line2[x] = np.min(line\_candidate2)

else :

line2[x] = np.min(line\_candidate2)

if (line3[x-2]-line3[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line3[x] = np.max(line\_candidate3)

elif (line3[x-2]-line3[x-1] > 0):

line3[x] = np.min(line\_candidate3)

elif (line3[x-2]-line3[x-1] == 0):

if (line3[x-3]-line3[x-2] < 0):

line3[x] = np.max(line\_candidate3)

elif (line3[x-3]-line3[x-2] > 0):

line3[x] = np.min(line\_candidate3)

else :

line3[x] = np.min(line\_candidate3)

if (line3[x-1] >= line3[x] + 120):

if (line3[x-2]-line3[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line3[x] = np.max(line\_candidate2)

elif (line3[x-2]-line3[x-1] > 0):# or (High[x-3]-High[x-1] > 0):

line3[x] = np.min(line\_candidate2)

elif (line3[x-2]-line3[x-1] == 0):

if (line3[x-3]-line3[x-2] < 0):

line3[x] = np.min(line\_candidate2)

else :

line3[x] = np.min(line\_candidate2)

#if line1[x] == (669 - 181 - 25) or

if (line2[x-1] >= line2[x] + 120):

if (line2[x-2]-line2[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line2[x] = np.max(line\_candidate1)

elif (line2[x-2]-line2[x-1] > 0):# or (High[x-3]-High[x-1] > 0):

line2[x] = np.min(line\_candidate1)

elif (line2[x-2]-line2[x-1] == 0):

if (line2[x-3]-line2[x-2] < 0):

line2[x] = np.min(line\_candidate1)

else :

line2[x] = np.min(line\_candidate1)

if (line2[x] == line1[x]):

if (line2[x-2]-line2[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line2[x] = np.max(line\_candidate2)

elif (line2[x-2]-line2[x-1] > 0):# or (High[x-3]-High[x-1] > 0):

line2[x] = np.min(line\_candidate2)

elif (line2[x-2]-line2[x-1] == 0):

if (line2[x-3]-line2[x-2] < 0):

line2[x] = np.min(line\_candidate2)

else :

line2[x] = np.min(line\_candidate2)

if (line3[x] == line2[x]):

if (line3[x-2]-line3[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line3[x] = np.max(line\_candidate3)

elif (line3[x-2]-line3[x-1] > 0):# or (High[x-3]-High[x-1] > 0):

line3[x] = np.min(line\_candidate3)

elif (line3[x-2]-line3[x-1] == 0):

if (line3[x-3]-line3[x-2] < 0):

line3[x] = np.min(line\_candidate3)

else :

line3[x] = np.min(line\_candidate3)

before\_count\_up = 1

#before\_count\_down = 1

line\_extreme\_up = 0

#line\_extreme\_down = 0

elif (line\_extreme\_up == 0 and before\_count\_up == 1):

#print(j,i)

#print(line\_candidate1)

#print(line\_candidate2)

#print(line\_candidate3)

line1[x] = 669 - 181 -25

if (line2[x-2]-line2[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line2[x] = np.max(line\_candidate1)

elif (line2[x-2]-line2[x-1] > 0):# or (High[x-3]-High[x-1] > 0):

line2[x] = np.min(line\_candidate1)

elif (line2[x-2]-line2[x-1] == 0):

if (line2[x-3]-line2[x-2] < 0):

line2[x] = np.max(line\_candidate1)

elif (line2[x-3]-line2[x-2] > 0):

line2[x] = np.min(line\_candidate1)

else :

line2[x] = np.min(line\_candidate1)

if (line3[x-2]-line3[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line3[x] = np.max(line\_candidate2)

elif (line3[x-2]-line3[x-1] > 0):# or (High[x-3]-High[x-1] > 0):

line3[x] = np.min(line\_candidate2)

elif (line3[x-2]-line3[x-1] == 0):

if (line3[x-3]-line3[x-2] < 0):

line3[x] = np.max(line\_candidate2)

elif (line3[x-3]-line3[x-2] > 0):

line3[x] = np.min(line\_candidate2)

else :

line3[x] = np.min(line\_candidate2)

if (line3[x-1] >= line3[x] + 120):

if (line3[x-2]-line3[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line3[x] = np.max(line\_candidate1)

elif (line3[x-2]-line3[x-1] > 0):# or (High[x-3]-High[x-1] > 0):

line3[x] = np.min(line\_candidate1)

elif (line3[x-2]-line3[x-1] == 0):

if (line3[x-3]-line3[x-2] < 0):

line3[x] = np.min(line\_candidate1)

else :

line3[x] = np.min(line\_candidate1)

if (line3[x] == line2[x]):

if (line3[x-2]-line3[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line3[x] = np.max(line\_candidate2)

elif (line3[x-2]-line3[x-1] > 0):# or (High[x-3]-High[x-1] > 0):

line3[x] = np.min(line\_candidate2)

elif (line3[x-2]-line3[x-1] == 0):

if (line3[x-3]-line3[x-2] < 0):

line3[x] = np.min(line\_candidate2)

else :

line3[x] = np.min(line\_candidate2)

before\_count\_up = 1

#before\_count\_down = 0

line\_extreme\_up = 0

#line\_extreme\_down = 0

elif (line\_extreme\_up == 1 and before\_count\_up == 1):

line1[x] = 669 - 181 - 25

if (line2[x-2]-line2[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line2[x] = np.max(line\_candidate2)

elif (line2[x-2]-line2[x-1] > 0):# or (High[x-3]-High[x-1] > 0):

line2[x] = np.min(line\_candidate2)

elif (line2[x-2]-line2[x-1] == 0):

if (line2[x-3]-line2[x-2] < 0):

line2[x] = np.max(line\_candidate2)

elif (line2[x-3]-line2[x-2] > 0):

line2[x] = np.min(line\_candidate2)

else :

line2[x] = np.min(line\_candidate2)

if (line3[x-2]-line3[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line3[x] = np.max(line\_candidate3)

elif (line3[x-2]-line3[x-1] > 0):# or (High[x-3]-High[x-1] > 0):

line3[x] = np.min(line\_candidate3)

elif (line3[x-2]-line3[x-1] == 0):

if (line3[x-3]-line3[x-2] < 0):

line3[x] = np.max(line\_candidate3)

elif (line3[x-3]-line3[x-2] > 0):

line3[x] = np.min(line\_candidate3)

else :

line3[x] = np.min(line\_candidate3)

if (line3[x-1] >= line3[x] + 120):

if (line3[x-2]-line3[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line3[x] = np.max(line\_candidate2)

elif (line3[x-2]-line3[x-1] > 0):# or (High[x-3]-High[x-1] > 0):

line3[x] = np.min(line\_candidate2)

elif (line3[x-2]-line3[x-1] == 0):

if (line3[x-3]-line3[x-2] < 0):

line3[x] = np.min(line\_candidate2)

else :

line3[x] = np.min(line\_candidate2)

#if line1[x] == (669 - 181 - 25) or

if (line2[x-1] >= line2[x] + 120):

if (line2[x-2]-line2[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line2[x] = np.max(line\_candidate1)

elif (line2[x-2]-line2[x-1] > 0):# or (High[x-3]-High[x-1] > 0):

line2[x] = np.min(line\_candidate1)

elif (line2[x-2]-line2[x-1] == 0):

if (line2[x-3]-line2[x-2] < 0):

line2[x] = np.min(line\_candidate1)

else :

line2[x] = np.min(line\_candidate1)

if (line2[x] == line1[x]):

if (line2[x-2]-line2[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line2[x] = np.max(line\_candidate2)

elif (line2[x-2]-line2[x-1] > 0):# or (High[x-3]-High[x-1] > 0):

line2[x] = np.min(line\_candidate2)

elif (line2[x-2]-line2[x-1] == 0):

if (line2[x-3]-line2[x-2] < 0):

line2[x] = np.min(line\_candidate2)

else :

line2[x] = np.min(line\_candidate2)

if (line3[x] == line2[x]):

if (line3[x-2]-line3[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line3[x] = np.max(line\_candidate3)

elif (line3[x-2]-line3[x-1] > 0):# or (High[x-3]-High[x-1] > 0):

line3[x] = np.min(line\_candidate3)

elif (line3[x-2]-line3[x-1] == 0):

if (line3[x-3]-line3[x-2] < 0):

line3[x] = np.min(line\_candidate3)

else :

line3[x] = np.min(line\_candidate3)

line\_extreme\_up = 0

#line\_extreme\_down = 0

before\_count\_up = 0

#before\_count\_down = 1

elif (line\_extreme\_up == 0 and before\_count\_up == 0):

if (line1[x-1] - line2[x-1] >=0 ) and (line1[x-1] - line2[x-1] < (len(line\_candidate1) + 7)) and ((np.min(line\_candidate1) - np.max(line\_candidate2)) >= 50) :

line1[x] = np.min(line\_candidate1)

line2[x] = np.max(line\_candidate1)

#print(line1[x-1] - line2[x-1])

#print(len(line\_candidate1))

if (line3[x-2]-line3[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line3[x] = np.max(line\_candidate2)

elif (line3[x-2]-line3[x-1] > 0):# or (High[x-3]-High[x-1] > 0):

line3[x] = np.min(line\_candidate2)

elif (line3[x-2]-line3[x-1] == 0):

if (line3[x-3]-line3[x-2] < 0):

line3[x] = np.max(line\_candidate2)

elif (line3[x-3]-line3[x-2] > 0):

line3[x] = np.min(line\_candidate2)

else :

line3[x] = np.min(line\_candidate2)

#if (line2[x] == np.max(line\_candidate1)) or (line2[x] == np.min(line\_candidate1)) or

if (line3[x-1] >= line3[x] + 120):

if (line3[x-2]-line3[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line3[x] = np.max(line\_candidate1)

elif (line3[x-2]-line3[x-1] > 0):# or (High[x-3]-High[x-1] > 0):

line3[x] = np.min(line\_candidate1)

elif (line3[x-2]-line3[x-1] == 0):

if (line3[x-3]-line3[x-2] < 0):

line3[x] = np.min(line\_candidate1)

else :

line3[x] = np.min(line\_candidate1)

if (line3[x] == line2[x]):

if (line3[x-2]-line3[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line3[x] = np.max(line\_candidate2)

elif (line3[x-2]-line3[x-1] > 0):# or (High[x-3]-High[x-1] > 0):

line3[x] = np.min(line\_candidate2)

elif (line3[x-2]-line3[x-1] == 0):

if (line3[x-3]-line3[x-2] < 0):

line3[x] = np.min(line\_candidate2)

else :

line3[x] = np.min(line\_candidate2)

elif (line2[x-1] - line3[x-1] >=0 ) and (line2[x-1] - line3[x-1] < (len(line\_candidate2) + 7)) and ((np.min(line\_candidate2) - np.max(line\_candidate3)) >= 50) :

#print(line2[x-1] - line3[x-1])

#print(len(line\_candidate1))

if (line1[x-2]-line1[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line1[x] = np.max(line\_candidate1)

elif (line1[x-2]-line1[x-1] > 0):# or (High[x-3]-High[x-1] > 0):

line1[x] = np.min(line\_candidate1)

elif (line1[x-2]-line1[x-1] == 0):

if (line1[x-3]-line1[x-2] < 0):

line1[x] = np.max(line\_candidate1)

elif (line1[x-3]-line1[x-2] > 0):

line1[x] = np.min(line\_candidate1)

else :

line1[x] = np.min(line\_candidate1)

line2[x] = np.min(line\_candidate2)

line3[x] = np.max(line\_candidate2)

else:

if (line1[x-2]-line1[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line1[x] = np.max(line\_candidate1)

elif (line1[x-2]-line1[x-1] > 0):# or (High[x-3]-High[x-1] > 0):

line1[x] = np.min(line\_candidate1)

elif (line1[x-2]-line1[x-1] == 0):

if (line1[x-3]-line1[x-2] < 0):

line1[x] = np.max(line\_candidate1)

elif (line1[x-3]-line1[x-2] > 0):

line1[x] = np.min(line\_candidate1)

else :

line1[x] = np.min(line\_candidate1)

if (line2[x-2]-line2[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line2[x] = np.max(line\_candidate2)

elif (line2[x-2]-line2[x-1] > 0):# or (High[x-3]-High[x-1] > 0):

line2[x] = np.min(line\_candidate2)

elif (line2[x-2]-line2[x-1] == 0):

if (line2[x-3]-line2[x-2] < 0):

line2[x] = np.max(line\_candidate2)

elif (line2[x-3]-line2[x-2] > 0):

line2[x] = np.min(line\_candidate2)

else :

line2[x] = np.min(line\_candidate2)

if (line3[x-2]-line3[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line3[x] = np.max(line\_candidate3)

elif (line3[x-2]-line3[x-1] > 0):# or (High[x-3]-High[x-1] > 0):

line3[x] = np.min(line\_candidate3)

elif (line3[x-2]-line3[x-1] == 0):

if (line3[x-3]-line3[x-2] < 0):

line3[x] = np.max(line\_candidate3)

elif (line3[x-3]-line3[x-2] > 0):

line3[x] = np.min(line\_candidate3)

else :

line3[x] = np.min(line\_candidate3)

#if (line2[x] == np.max(line\_candidate1)) or (line2[x] == np.min(line\_candidate1)) or

if (line3[x-1] >= line3[x] + 120):

if (line3[x-2]-line3[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line3[x] = np.max(line\_candidate2)

elif (line3[x-2]-line3[x-1] > 0):# or (High[x-3]-High[x-1] > 0):

line3[x] = np.min(line\_candidate2)

elif (line3[x-2]-line3[x-1] == 0):

if (line3[x-3]-line3[x-2] < 0):

line3[x] = np.min(line\_candidate2)

else :

line3[x] = np.min(line\_candidate2)

#if line1[x] == (669 - 181 - 25) or

if (line2[x-1] >= line2[x] + 120):

if (line2[x-2]-line2[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line2[x] = np.max(line\_candidate1)

elif (line2[x-2]-line2[x-1] > 0):# or (High[x-3]-High[x-1] > 0):

line2[x] = np.min(line\_candidate1)

elif (line2[x-2]-line2[x-1] == 0):

if (line2[x-3]-line2[x-2] < 0):

line2[x] = np.min(line\_candidate1)

else :

line2[x] = np.min(line\_candidate1)

if (line2[x] == line1[x]):

if (line2[x-2]-line2[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line2[x] = np.max(line\_candidate2)

elif (line2[x-2]-line2[x-1] > 0):# or (High[x-3]-High[x-1] > 0):

line2[x] = np.min(line\_candidate2)

elif (line2[x-2]-line2[x-1] == 0):

if (line2[x-3]-line2[x-2] < 0):

line2[x] = np.min(line\_candidate2)

else :

line2[x] = np.min(line\_candidate2)

if (line3[x] == line2[x]):

if (line3[x-2]-line3[x-1] < 0):# or (High[x-3]-High[x-1] < 0):

line3[x] = np.max(line\_candidate3)

elif (line3[x-2]-line3[x-1] > 0):# or (High[x-3]-High[x-1] > 0):

line3[x] = np.min(line\_candidate3)

elif (line3[x-2]-line3[x-1] == 0):

if (line3[x-3]-line3[x-2] < 0):

line3[x] = np.min(line\_candidate3)

else :

line3[x] = np.min(line\_candidate3)

line = 0

tem\_counts1 = 0

tem\_counts2 = 0

tem\_counts3 = 0

line\_candidate1 = []

line\_candidate2 = []

line\_candidate3 = []

#sio.savemat("line10.mat",{"line10":line1})

#sio.savemat("line11.mat",{"line11":line2})

#sio.savemat("line12.mat",{"line12":line3})

high[9] = line1

high[10] = line2

high[11] = line3

######################################################################################

V1 = 0

V2 = 0

V3 = 0

V4 = 0

V5 = 0

V6 = 0

Positive = 0

uptype = 0

downtype = 0

curve = 0

T1\_sum = 0

T2\_sum = 0

T1\_final = float(0)

T2\_final = float(0)

I\_minus = 0

II\_minus = 0

III\_minus = 0

area = 0

area\_sum = 0

area\_final = float(0)

square = 0

aVL\_max = 0

PvN = float(0)

for k in range(12):

Q\_final = 0

S\_final = 0

A = 0

A1 = 0

A2 = 0

B = 0

C = 0

High = np.zeros(shape =120)

print('Picture ' + name[k] + ':')

#sio.savemat("line"+str(k)+".mat",{"line"+str(k):High})

High = high[k]

A1 = np.max(High)

A2 = np.min(High)

counts = np.bincount(High)

#返回众数

A = np.argmax(counts)

#print("Up or down: " + str(B)+"," + str(C))

#G = - A2

#M = A1 - base

R\_index1 = 0

R\_index2 = 0

#print(High)

down = 0

#print(base)

if (A - A2) > (A1 - A):

#print('down')

downtype = 1

if name[k] == 'II':

II = 0

if name[k] == 'III':

III = 0

if name[k] == 'V1':

V1 = 0

if name[k] == 'V2':

V2 = 0

if name[k] == 'V3':

V3 = 0

if name[k] == 'V4':

V4 = 0

if name[k] == 'V5':

V5 = 0

if name[k] == 'V6':

V6 = 0

if k>6 :

continue#break

#print(np.where(High==1))

#print(High\_extreme\_down)

#print(C)

R\_index = int(np.where(High==np.min(High))[0][0])

while R\_index <= 50 or R\_index >=200:

High[R\_index] = A

New\_index = int(np.where(High==np.min(High))[0][0])

#print("New\_index"+ str(New\_index))

R\_index = New\_index

#print(R\_index)

R\_index1 = R\_index - 2

R\_index2 = R\_index + 2

if R\_index1 - 15 >= 0 :

downrange = R\_index1 - 15

else:

downrange = 0

if R\_index2 + 15 <= 239 :

uprange = R\_index2 + 15

else:

uprange = 239

Base = np.bincount(High[R\_index1-30:R\_index2+30])

base = np.argmax(Base)

#print(base)

#print(R\_index1)

#print(R\_index2)

'''''''''''''''

down

Find S extreme

'''''''''''''''

S\_index = 0

for i in range(R\_index2, uprange):

if (i + 2 <= 239) and (i - 2 >= 0 ):

if (High[i-2]<High[i]) and (High[i+2]<High[i]):

S\_index = i

break

elif ((High[i-2]<High[i]) and (High[i+2]<=High[i])) or ((High[i-2]<=High[i]) and (High[i+2]<High[i])):

S\_index = i

break

elif (High[i-2]<=High[i]) and (High[i+2]<=High[i]):

S\_index = i

break

elif ((High[i-2]<High[i]) and (High[i+1]<High[i])) or ((High[i-1]<High[i]) and (High[i+2]<High[i])):

S\_index = i

break

elif (((High[i-2]<=High[i]) and (High[i+1]<High[i])) or ((High[i-2]<High[i]) and (High[i+1]<=High[i]))) or (((High[i-1]<=High[i]) and (High[i+2]<High[i])) or ((High[i-1]<High[i]) and (High[i+2]<=High[i]))):

S\_index = i

break

elif (((High[i-2]<=High[i]) and (High[i+1]<=High[i])) or ((High[i-1]<=High[i]) and (High[i+2]<=High[i]))):

S\_index = i

break

elif ((High[i-1]<High[i]) and (High[i+1]<High[i])):

S\_index = i

break

elif ((High[i-1]<=High[i]) and (High[i+1]<High[i])) or ((High[i-1]<High[i]) and (High[i+1]<=High[i])):

S\_index = i

break

elif ((High[i-1]<=High[i]) and (High[i+1]<=High[i])):

S\_index = i

break

elif ((i + 1 <= 239) and (i - 1 >= 0 )):

if ((High[i-1]<High[i]) and (High[i+1]<High[i])):

S\_index = i

break

elif ((High[i-1]<=High[i]) and (High[i+1]<High[i])) or ((High[i-1]<High[i]) and (High[i+1]<=High[i])):

S\_index = i

break

elif ((High[i-1]<=High[i]) and (High[i+1]<=High[i])):

S\_index = i

break

#if len(S\_index)>3:

# S = int(np.min(S\_index))

#else:

# S = int(np.median(S\_index))

'''''''''''''''

up

Find S\_final

'''''''''''''''

S\_final = 0

for i in range(S\_index, S\_index + 15 ):

#print("i = "+ str(i))

F = abs(High[i+1] - High[i]) + abs(High[i+2] - High[i+1]) + abs(High[i+3] - High[i+2])

#print(F)

if (F <= 1) and ( abs(base - High[i]) <= 10 ):

S\_final = i

break

if S\_final == 0:

for i in range(S\_index, S\_index + 15 ):

F = abs(High[i+1] - High[i]) + abs(High[i+2] - High[i+1]) + abs(High[i+3] - High[i+2])

#print(F)

if (F <= 2) and ( abs(base - High[i]) <= 10 ):

S\_final = i

break

if S\_final == 0:

for i in range(S\_index, S\_index + 15 ):

F = abs(High[i+1] - High[i]) + abs(High[i+2] - High[i+1]) + abs(High[i+3] - High[i+2])

#print(F)

if (F <= 3) and ( abs(base - High[i]) <= 10 ):

S\_final = i

break

#print(S\_final)

#print('Hello again')

break

'''''''''''''''

down

Find Q extreme

'''''''''''''''

Q\_index = 0

for i in range(R\_index1, downrange, -1):

#print(str(i) + ", " + str(High[i]))

if (i + 2 <= 239) and (i - 2 >= 0 ):

if (abs(High[i] - High[S\_index]) < 20):

if (High[i-2]<High[i]) and (High[i+2]<High[i]):

Q\_index = i

break

elif ((High[i-2]<High[i]) and (High[i+2]<=High[i])) or ((High[i-2]<=High[i]) and (High[i+2]<High[i])):

Q\_index = i

break

elif (High[i-2]<=High[i]) and (High[i+2]<=High[i]):

Q\_index = i

break

elif ((High[i-2]<High[i]) and (High[i+1]<High[i])) or ((High[i-1]<High[i]) and (High[i+2]<High[i])):

Q\_index = i

break

elif (((High[i-2]<=High[i]) and (High[i+1]<High[i])) or ((High[i-2]<High[i]) and (High[i+1]<=High[i]))) or (((High[i-1]<=High[i]) and (High[i+2]<High[i])) or ((High[i-1]<High[i]) and (High[i+2]<=High[i]))):

Q\_index = i

break

elif (((High[i-2]<=High[i]) and (High[i+1]<=High[i])) or ((High[i-1]<=High[i]) and (High[i+2]<=High[i]))):

Q\_index = i

break

if ((High[i-1]<High[i]) and (High[i+1]<High[i])):

Q\_index = i

break

elif ((High[i-1]<=High[i]) and (High[i+1]<High[i])) or ((High[i-1]<High[i]) and (High[i+1]<=High[i])):

Q\_index = i

break

elif ((High[i-1]<=High[i]) and (High[i+1]<=High[i])):

Q\_index = i

break

elif (i + 1 <= 239) and (i - 1 >= 0 ):

if (abs(High[i] - High[S\_index]) < 20):

if ((High[i-1]<High[i]) and (High[i+1]<High[i])):

Q\_index = i

break

elif ((High[i-1]<=High[i]) and (High[i+1]<High[i])) or ((High[i-1]<High[i]) and (High[i+1]<=High[i])):

Q\_index = i

break

elif ((High[i-1]<=High[i]) and (High[i+1]<=High[i])):

Q\_index = i

break

#print("Q\_index: " + str(Q\_index))

#if len(Q\_index)>3:

# Q = int(np.max(Q\_index))

#else:

# Q = int(np.median(Q\_index))

'''''''''''''''

down

Find Q\_final

'''''''''''''''

Q\_final = 0

D = 0

for i in range(Q\_index, Q\_index - 15 ,-1):

D = abs(High[i] - High[i-1]) + abs(High[i-1] - High[i-2]) + abs(High[i-2] - High[i-3])

if High[i] <= base or D <= 1:

Q\_final = i

break

'''

D = abs(High[i] - High[i-1]) + abs(High[i-1] - High[i-2]) + abs(High[i-2] - High[i-3])

if (D <= 1) and ( abs(High[S\_final] - High[i]) <= 20 ):

Q\_final = i

#print('Hello')

break

elif (D <= 2) and ( abs(High[S\_final] - High[i]) <= 20 ):

Q\_final = i

#print('Hello')

break

#print("Q\_final: "+ str(Q\_final))

'''

'''''''''''''''

down

Find Qs

'''''''''''''''

#find QS extreme

R = 0

num1 = 0

num2 = 0

up = 0

Qs = 0

for i in range(Q\_final , Q\_final - 20 ,-1):

R = abs(High[i] - High[i-1]) + abs(High[i-1] - High[i-2]) + abs(High[i-2] - High[i-3])

#print (R)

#print (num)

if name[k] == 'aVR':

if (High[i-2]-High[i]<0 or High[i-1]-High[i]<=0) :#and (High[i-3]-High[i-1]<0 or High[i-2]-High[i-1]<=0):

num1 = num1 + 1

if num1 >=3:

up = 1

if High[i-1]-High[i]>0 and up == 1:

#if (High[i-2]-High[i]>0 or High[i-1]-High[i]>=0) and (High[i-3]-High[i-1]>0 or High[i-2]-High[i-1]>=0) and up==1:

num2 = num2 + 1

if num2 >=3:

Qs = i + 3

break

else :

#################小波应该为负############

if (High[i-2]-High[i]<0 or High[i-1]-High[i]<=0) :#and (High[i-3]-High[i-1]>0 or High[i-2]-High[i-1]>=0):

num1 = num1 + 1

if num1 >=3:

up = 1

#print(up)

#if (High[i-2]-High[i]<0 or High[i-1]-High[i]<=0) and (High[i-3]-High[i-1]<0 or High[i-2]-High[i-1]<=0) and up==1:

if High[i-1]-High[i]>0 and up == 1:

num2 = num2 + 1

if num2 >=3:

Qs = i + 3

break

if Qs == 0:

for i in range(Q\_final , Q\_final - 20 ,-1):

R = abs(High[i] - High[i-1]) + abs(High[i-1] - High[i-2]) + abs(High[i-2] - High[i-3])

#print (R)

#print (num)

if name[k] == 'aVR':

if (High[i-2]-High[i]<0 or High[i-1]-High[i]<=0) :#and (High[i-3]-High[i-1]<0 or High[i-2]-High[i-1]<=0):

num1 = num1 + 1

if num1 >=3:

up = 1

if High[i-1]-High[i]>0 and up == 1:

#if (High[i-2]-High[i]>0 or High[i-1]-High[i]>=0) and (High[i-3]-High[i-1]>0 or High[i-2]-High[i-1]>=0) and up==1:

num2 = num2 + 1

if num2 >=2:

Qs = i + 3

break

else :

if (High[i-2]-High[i]<0 or High[i-1]-High[i]<=0) :#and (High[i-3]-High[i-1]>0 or High[i-2]-High[i-1]>=0):

num1 = num1 + 1

if num1 >=3:

up = 1

#print(up)

#if (High[i-2]-High[i]<0 or High[i-1]-High[i]<=0) and (High[i-3]-High[i-1]<0 or High[i-2]-High[i-1]<=0) and up==1:

if High[i-1]-High[i]>0 and up == 1:

num2 = num2 + 1

if num2 >=2:

Qs = i + 3

break

if Qs == 0:

for i in range(Q\_final , Q\_final - 20 ,-1):

R = abs(High[i] - High[i-1]) + abs(High[i-1] - High[i-2]) + abs(High[i-2] - High[i-3])

#print (R)

#print (num)

if name[k] == 'aVR':

if (High[i-2]-High[i]<0 or High[i-1]-High[i]<=0) :#and (High[i-3]-High[i-1]<0 or High[i-2]-High[i-1]<=0):

num1 = num1 + 1

if num1 >=2:

up = 1

if High[i-1]-High[i]>0 and up == 1:

#if (High[i-2]-High[i]>0 or High[i-1]-High[i]>=0) and (High[i-3]-High[i-1]>0 or High[i-2]-High[i-1]>=0) and up==1:

num2 = num2 + 1

if num2 >=3:

Qs = i + 3

break

else :

if (High[i-2]-High[i]<0 or High[i-1]-High[i]<=0) :#and (High[i-3]-High[i-1]>0 or High[i-2]-High[i-1]>=0):

num1 = num1 + 1

if num1 >=2:

up = 1

#print(up)

#if (High[i-2]-High[i]<0 or High[i-1]-High[i]<=0) and (High[i-3]-High[i-1]<0 or High[i-2]-High[i-1]<=0) and up==1:

if High[i-1]-High[i]>0 and up == 1:

num2 = num2 + 1

if num2 >=3:

Qs = i + 3

break

if Qs == 0:

for i in range(Q\_final , Q\_final - 20 ,-1):

R = abs(High[i] - High[i-1]) + abs(High[i-1] - High[i-2]) + abs(High[i-2] - High[i-3])

#print (R)

#print (num)

if name[k] == 'aVR':

if (High[i-2]-High[i]<0 or High[i-1]-High[i]<=0) :#and (High[i-3]-High[i-1]<0 or High[i-2]-High[i-1]<=0):

num1 = num1 + 1

if num1 >=2:

up = 1

if High[i-1]-High[i]>0 and up == 1:

#if (High[i-2]-High[i]>0 or High[i-1]-High[i]>=0) and (High[i-3]-High[i-1]>0 or High[i-2]-High[i-1]>=0) and up==1:

num2 = num2 + 1

if num2 >=2:

Qs = i + 3

break

else :

if (High[i-2]-High[i]<0 or High[i-1]-High[i]<=0) :#and (High[i-3]-High[i-1]>0 or High[i-2]-High[i-1]>=0):

num1 = num1 + 1

if num1 >=2:

up = 1

#print(up)

#if (High[i-2]-High[i]<0 or High[i-1]-High[i]<=0) and (High[i-3]-High[i-1]<0 or High[i-2]-High[i-1]<=0) and up==1:

if High[i-1]-High[i]>0 and up == 1:

num2 = num2 + 1

if num2 >=2:

Qs = i + 3

break

num1 = 0

num2 = 0

up = 0

print("Qs: "+ str(Qs))

#print("Qs: "+ str(Qs))

#Qs = int(np.median(Qs))

'''''''''''''''

down

Find Qt

'''''''''''''''

K1 = 0

K2 = 0

K3 = 0

M = 0

Qt = 0

for i in range(Qs, Qs - 10 ,-1):

K1 = abs(High[i] - High[i-1]) + abs(High[i-1] - High[i-2]) + abs(High[i-2] - High[i-3]) + abs(High[i-3] - High[i-4]) #+ abs(High[i-4] - High[i-5])

K2 = abs(High[i-1] - High[i-2]) + abs(High[i-2] - High[i-3]) + abs(High[i-3] - High[i-4]) + abs(High[i-4] - High[i-5])

#K3 = abs(High[i-2] - High[i-3]) + abs(High[i-3] - High[i-4]) + abs(High[i-4] - High[i-5]) + abs(High[i-5] - High[i-6])

#print("K1:"+str(K1))

#print("K2:"+str(K2))

#print("K3:"+str(K3))

#if K >=5 :

#M = M + 1

#if M >= 5 :

if K1 <= 2 and K2 <= 1 :#and K3 <= 1:

Qt = i

#print('Hello')

break

#elif K <= 1 :

# Qt = i

# #print('Hello')

# break

'''

if i == (Qs - 9):

for i in range(Qs, Qs - 20 ,-1):

K1 = abs(High[i] - High[i-1]) + abs(High[i-1] - High[i-2]) + abs(High[i-2] - High[i-3])# + abs(High[i-3] - High[i-4])

#print(K1)

if K1 <= 1:

Qt = i

break

'''

if Qt == 0:

for i in range(Qs, Qs - 10 ,-1):

K1 = abs(High[i] - High[i-1]) + abs(High[i-1] - High[i-2]) + abs(High[i-2] - High[i-3]) + abs(High[i-3] - High[i-4]) #+ abs(High[i-4] - High[i-5])

K2 = abs(High[i-1] - High[i-2]) + abs(High[i-2] - High[i-3]) + abs(High[i-3] - High[i-4]) + abs(High[i-4] - High[i-5])

#K3 = abs(High[i-2] - High[i-3]) + abs(High[i-3] - High[i-4]) + abs(High[i-4] - High[i-5]) + abs(High[i-5] - High[i-6])

#if K >=5 :

#M = M + 1

#if M >= 5 :

if K1 <= 2 and K2 <= 2: #and K3 <= 1:

Qt = i

#print('Hello')

break

#elif K <= 1 :

# Qt = i

# #print('Hello')

# break

if Qt == 0:

for i in range(Qs, Qs - 10 ,-1):

K1 = abs(High[i] - High[i-1]) + abs(High[i-1] - High[i-2]) + abs(High[i-2] - High[i-3]) + abs(High[i-3] - High[i-4]) #+ abs(High[i-4] - High[i-5])

K2 = abs(High[i-1] - High[i-2]) + abs(High[i-2] - High[i-3]) + abs(High[i-3] - High[i-4]) + abs(High[i-4] - High[i-5])

#K3 = abs(High[i-2] - High[i-3]) + abs(High[i-3] - High[i-4]) + abs(High[i-4] - High[i-5]) + abs(High[i-5] - High[i-6])

#if K >=5 :

#M = M + 1

#if M >= 5 :

if K1 <= 3 and K2 <= 2 :#and K3 <= 2:

Qt = i

#print('Hello')

break

#elif K <= 1 :

# Qt = i

# #print('Hello')

# break

if Qt == 0:

for i in range(Qs, Qs - 10 ,-1):

K1 = abs(High[i] - High[i-1]) + abs(High[i-1] - High[i-2]) + abs(High[i-2] - High[i-3]) + abs(High[i-3] - High[i-4]) #+ abs(High[i-4] - High[i-5])

K2 = abs(High[i-1] - High[i-2]) + abs(High[i-2] - High[i-3]) + abs(High[i-3] - High[i-4]) + abs(High[i-4] - High[i-5])

#K3 = abs(High[i-2] - High[i-3]) + abs(High[i-3] - High[i-4]) + abs(High[i-4] - High[i-5]) + abs(High[i-5] - High[i-6])

if K1 <= 3 and K2 <= 3 :#and K3 <= 2:

Qt = i

break

if Qt == 0:

for i in range(Qs, Qs - 10 ,-1):

K1 = abs(High[i] - High[i-1]) + abs(High[i-1] - High[i-2]) + abs(High[i-2] - High[i-3]) + abs(High[i-3] - High[i-4]) #+ abs(High[i-4] - High[i-5])

K2 = abs(High[i-1] - High[i-2]) + abs(High[i-2] - High[i-3]) + abs(High[i-3] - High[i-4]) + abs(High[i-4] - High[i-5])

#K3 = abs(High[i-2] - High[i-3]) + abs(High[i-3] - High[i-4]) + abs(High[i-4] - High[i-5]) + abs(High[i-5] - High[i-6])

if K1 <= 4 and K2 <= 3 :#and K3 <= 2:

Qt = i

break

if Qt == 0:

for i in range(Qs, Qs - 10 ,-1):

K1 = abs(High[i] - High[i-1]) + abs(High[i-1] - High[i-2]) + abs(High[i-2] - High[i-3]) + abs(High[i-3] - High[i-4]) #+ abs(High[i-4] - High[i-5])

K2 = abs(High[i-1] - High[i-2]) + abs(High[i-2] - High[i-3]) + abs(High[i-3] - High[i-4]) + abs(High[i-4] - High[i-5])

#K3 = abs(High[i-2] - High[i-3]) + abs(High[i-3] - High[i-4]) + abs(High[i-4] - High[i-5]) + abs(High[i-5] - High[i-6])

if K1 <= 4 and K2 <= 4 :#and K3 <= 2:

Qt = i

break

if Qt == 0:

for i in range(Qs, Qs - 10 ,-1):

K1 = abs(High[i] - High[i-1]) + abs(High[i-1] - High[i-2]) + abs(High[i-2] - High[i-3]) + abs(High[i-3] - High[i-4]) #+ abs(High[i-4] - High[i-5])

K2 = abs(High[i-1] - High[i-2]) + abs(High[i-2] - High[i-3]) + abs(High[i-3] - High[i-4]) + abs(High[i-4] - High[i-5])

#K3 = abs(High[i-2] - High[i-3]) + abs(High[i-3] - High[i-4]) + abs(High[i-4] - High[i-5]) + abs(High[i-5] - High[i-6])

if K1 <= 5 and K2 <= 4 :#and K3 <= 2:

Qt = i

break

if Qt == 0:

for i in range(Qs, Qs - 10 ,-1):

K1 = abs(High[i] - High[i-1]) + abs(High[i-1] - High[i-2]) + abs(High[i-2] - High[i-3]) + abs(High[i-3] - High[i-4]) #+ abs(High[i-4] - High[i-5])

K2 = abs(High[i-1] - High[i-2]) + abs(High[i-2] - High[i-3]) + abs(High[i-3] - High[i-4]) + abs(High[i-4] - High[i-5])

#K3 = abs(High[i-2] - High[i-3]) + abs(High[i-3] - High[i-4]) + abs(High[i-4] - High[i-5]) + abs(High[i-5] - High[i-6])

if K1 <= 5 and K2 <= 5 :#and K3 <= 2:

Qt = i

break

K1 = 0

K2 = 0

K3 = 0

#up type

else:

#print('up')

uptype = 1

#R\_index = 0

#print(np.where(High==119))

#print()

if name[k] == 'II':

II = 1

if name[k] == 'III':

III = 1

if name[k] == 'V1':

V1 = 1

if name[k] == 'V2':

V2 = 1

if name[k] == 'V3':

V3 = 1

if name[k] == 'V4':

V4 = 1

if name[k] == 'V5':

V5 = 1

if name[k] == 'V6':

V6 = 1

if k > 6 :

continue#break

#elif B >= 4:

#print(High\_extreme)

#High\_extreme = High\_extreme.sort()

#print(High\_extreme)

#print(High)

# R\_index1 = int(np.where(High==np.max(High))[0][2]) #High\_extreme[2]

# R\_index2 = int(np.where(High==np.max(High))[0][3]) #High\_extreme[3]

#print(np.where(High==1))

R\_index = int(np.where(High==np.max(High))[0][0])

while R\_index <= 50 or R\_index >=200:

High[R\_index] = A

New\_index = int(np.where(High==np.max(High))[0][0])

#print(New\_index)

R\_index = New\_index

R\_index1 = R\_index - 2

R\_index2 = R\_index + 2

if R\_index1 - 15 >= 0 :

downrange = R\_index1 - 15

else:

downrange = 0

if R\_index2 + 15 <= 239 :

uprange = R\_index2 + 15

else:

uprange = 239

Base = np.bincount(High[R\_index1-30:R\_index2+30])

base = np.argmax(Base)

#print(base)

'''''''''''''''

up

Find S extreme

'''''''''''''''

S\_index = 0

#print(R\_index2)

for i in range(R\_index2, uprange):

if (i + 2 <= 239) and (i - 2 >= 0 ):

if (High[i-2]>High[i]) and (High[i+2]>High[i]):

S\_index = i

break

elif ((High[i-2]>High[i]) and (High[i+2]>=High[i])) or ((High[i-2]>=High[i]) and (High[i+2]>High[i])):

S\_index = i

break

elif (High[i-2]>=High[i]) and (High[i+2]>=High[i]):

S\_index = i

break

elif ((High[i-2]>High[i]) and (High[i+1]>High[i])) or ((High[i-1]>High[i]) and (High[i+2]>High[i])):

S\_index = i

break

elif (((High[i-2]>=High[i]) and (High[i+1]>High[i])) or ((High[i-2]>High[i]) and (High[i+1]>=High[i]))) or (((High[i-1]>=High[i]) and (High[i+2]>High[i])) or ((High[i-1]>High[i]) and (High[i+2]>=High[i]))):

S\_index = i

break

elif (((High[i-2]>=High[i]) and (High[i+1]>=High[i])) or ((High[i-1]>=High[i]) and (High[i+2]>=High[i]))):

S\_index = i

break

elif ((High[i-1]>High[i]) and (High[i+1]>High[i])):

S\_index = i

break

elif ((High[i-1]>=High[i]) and (High[i+1]>High[i])) or ((High[i-1]>High[i]) and (High[i+1]>=High[i])):

S\_index = i

break

elif ((High[i-1]>=High[i]) and (High[i+1]>=High[i])):

S\_index = i

break

elif (i + 1 <= 239) and (i - 1 >= 0 ):

if ((High[i-1]>High[i]) and (High[i+1]>High[i])):

S\_index = i

break

elif ((High[i-1]>=High[i]) and (High[i+1]>High[i])) or ((High[i-1]>High[i]) and (High[i+1]>=High[i])):

S\_index = i

break

elif ((High[i-1]>=High[i]) and (High[i+1]>=High[i])):

S\_index = i

break

#print(S\_index)

#if len(S\_index)>3:

# S = int(np.min(S\_index))

#else:

# S = int(np.median(S\_index))

'''''''''''''''

up

Find S\_final

'''''''''''''''

S\_final = 0

for i in range(S\_index , S\_index + 15 ):

#print("i = "+ str(i))

F = abs(High[i+1] - High[i]) + abs(High[i+2] - High[i+1]) + abs(High[i+3] - High[i+2])

#print(F)

if (F <= 1) and ( abs(base - High[i]) <= 10 ):

S\_final = i

#print('Hello again')

#print(S\_final)

break

if S\_final == 0:

for i in range(S\_index, S\_index + 15 ):

F = abs(High[i+1] - High[i]) + abs(High[i+2] - High[i+1]) + abs(High[i+3] - High[i+2])

#print(F)

if (F <= 2) and ( abs(base - High[i]) <= 10 ):

S\_final = i

break

if S\_final == 0:

for i in range(S\_index, S\_index + 15 ):

F = abs(High[i+1] - High[i]) + abs(High[i+2] - High[i+1]) + abs(High[i+3] - High[i+2])

#print(F)

if (F <= 3) and ( abs(base - High[i]) <= 10 ):

S\_final = i

break

'''''''''''''''

up

Find Q extreme

'''''''''''''''

Q\_index = 0

#print(R\_index1)

for i in range(R\_index1, downrange, -1):

if (i + 2 <= 239) and (i - 2 >= 0 ):

#print(str(S))

#print(str(i) + ',' + str(abs(High[i] - High[S])))

if (abs(High[i] - High[S\_index]) < 20):

if (High[i-2]>High[i]) and (High[i+2]>High[i]):

Q\_index = i

break

elif ((High[i-2]>High[i]) and (High[i+2]>=High[i])) or ((High[i-2]>=High[i]) and (High[i+2]>High[i])):

Q\_index = i

break

elif (High[i-2]>=High[i]) and (High[i+2]>=High[i]):

Q\_index = i

break

elif ((High[i-2]>High[i]) and (High[i+1]>High[i])) or ((High[i-1]>High[i]) and (High[i+2]>High[i])):

Q\_index = i

break

elif (((High[i-2]>=High[i]) and (High[i+1]>High[i])) or ((High[i-2]>High[i]) and (High[i+1]>=High[i]))) or (((High[i-1]>=High[i]) and (High[i+2]>High[i])) or ((High[i-1]>High[i]) and (High[i+2]>=High[i]))):

Q\_index = i

break

elif (((High[i-2]>=High[i]) and (High[i+1]>=High[i])) or ((High[i-1]>=High[i]) and (High[i+2]>=High[i]))):

Q\_index = i

break

elif ((High[i-1]>High[i]) and (High[i+1]>High[i])):

Q\_index = i

break

elif ((High[i-1]>=High[i]) and (High[i+1]>High[i])) or ((High[i-1]>High[i]) and (High[i+1]>=High[i])):

Q\_index = i

break

elif ((High[i-1]>=High[i]) and (High[i+1]>=High[i])):

Q\_index = i

break

elif (i + 1 <= 239) and (i - 1 >= 0 ):

if (abs(High[i] - High[S\_index]) < 20):

if ((High[i-1]>High[i]) and (High[i+1]>High[i])):

Q\_index = i

break

elif ((High[i-1]>=High[i]) and (High[i+1]>High[i])) or ((High[i-1]>High[i]) and (High[i+1]>=High[i])):

Q\_index = i

break

elif ((High[i-1]>=High[i]) and (High[i+1]>=High[i])):

Q\_index = i

break

'''''''''''''''

up

Find Q\_final

'''''''''''''''

Q\_final = 0

D = 0

for i in range(Q\_index , Q\_index - 15 ,-1):

D = abs(High[i] - High[i-1]) + abs(High[i-1] - High[i-2]) + abs(High[i-2] - High[i-3])

if High[i] >= base or D <= 1:

Q\_final = i

break

'''

D = abs(High[i] - High[i-1]) + abs(High[i-1] - High[i-2]) + abs(High[i-2] - High[i-3])

if (D <= 1) and ( abs(High[S\_final] - High[i]) <= 20 ):

Q\_final = i

#print('Hello')

break

elif (D <= 2) and ( abs(High[S\_final] - High[i]) <= 20 ):

Q\_final = i

#print('Hello')

break

'''

'''''''''''''''

up

Find Qs

'''''''''''''''

#find QS extreme

R = 0

num1 = 0

num2 = 0

up = 0

Qs = 0

for i in range(Q\_final , Q\_final - 20 ,-1):

R = abs(High[i] - High[i-1]) + abs(High[i-1] - High[i-2]) + abs(High[i-2] - High[i-3])

#print (R)

#if R >=3 :

#num = num + 1

#print (num)

if name[k] == 'aVR':

if (High[i-2]-High[i]<0 or High[i-1]-High[i]<=0): #and (High[i-3]-High[i]<0 or High[i-2]-High[i-1]<=0):

num1 = num1 + 1

if num1 >=3:

up = 1

if High[i-1]-High[i]>0 and up == 1:

#if (High[i-2]-High[i]>0 or High[i-1]-High[i]>=0) and (High[i-3]-High[i-1]>0 or High[i-2]-High[i-1]>=0) and up==1:

num2 = num2 + 1

if num2 >=3:

Qs = i + 3

break

else :

if (High[i-2]-High[i]>0 or High[i-1]-High[i]>=0) :#and (High[i-3]-High[i-1]>0 or High[i-2]-High[i-1]>=0):

num1 = num1 + 1

if num1 >=3:

up = 1

#print(i)

if High[i-1]-High[i]<0 and up == 1:

#if (High[i-2]-High[i]<0 or High[i-1]-High[i]<=0) and (High[i-3]-High[i-1]<0 or High[i-2]-High[i-1]<=0) and up==1:

num2 = num2 + 1

if num2 >=3:

Qs = i + 3

break

if Qs == 0:

for i in range(Q\_final , Q\_final - 20 ,-1):

if name[k] == 'aVR':

if (High[i-2]-High[i]<0 or High[i-1]-High[i]<=0): #and (High[i-3]-High[i]<0 or High[i-2]-High[i-1]<=0):

num1 = num1 + 1

if num1 >=2:

up = 1

if High[i-1]-High[i]>0 and up == 1:

#if (High[i-2]-High[i]>0 or High[i-1]-High[i]>=0) and (High[i-3]-High[i-1]>0 or High[i-2]-High[i-1]>=0) and up==1:

num2 = num2 + 1

if num2 >=3:

Qs = i + 3

break

else :

if (High[i-2]-High[i]>0 or High[i-1]-High[i]>=0) :#and (High[i-3]-High[i-1]>0 or High[i-2]-High[i-1]>=0):

num1 = num1 + 1

if num1 >=2:

up = 1

#print(i)

if High[i-1]-High[i]<0 and up == 1:

#if (High[i-2]-High[i]<0 or High[i-1]-High[i]<=0) and (High[i-3]-High[i-1]<0 or High[i-2]-High[i-1]<=0) and up==1:

num2 = num2 + 1

if num2 >=3:

Qs = i + 3

break

if Qs == 0:

for i in range(Q\_final , Q\_final - 20 ,-1):

if name[k] == 'aVR':

if (High[i-2]-High[i]<0 or High[i-1]-High[i]<=0): #and (High[i-3]-High[i]<0 or High[i-2]-High[i-1]<=0):

num1 = num1 + 1

if num1 >=3:

up = 1

if High[i-1]-High[i]>0 and up == 1:

#if (High[i-2]-High[i]>0 or High[i-1]-High[i]>=0) and (High[i-3]-High[i-1]>0 or High[i-2]-High[i-1]>=0) and up==1:

num2 = num2 + 1

if num2 >=2:

Qs = i + 3

break

else :

if (High[i-2]-High[i]>0 or High[i-1]-High[i]>=0) :#and (High[i-3]-High[i-1]>0 or High[i-2]-High[i-1]>=0):

num1 = num1 + 1

if num1 >=3:

up = 1

#print(i)

if High[i-1]-High[i]<0 and up == 1:

#if (High[i-2]-High[i]<0 or High[i-1]-High[i]<=0) and (High[i-3]-High[i-1]<0 or High[i-2]-High[i-1]<=0) and up==1:

num2 = num2 + 1

if num2 >=2:

Qs = i + 3

break

if Qs == 0:

for i in range(Q\_final , Q\_final - 20 ,-1):

if name[k] == 'aVR':

if (High[i-2]-High[i]<0 or High[i-1]-High[i]<=0): #and (High[i-3]-High[i]<0 or High[i-2]-High[i-1]<=0):

num1 = num1 + 1

if num1 >=2:

up = 1

if High[i-1]-High[i]>0 and up == 1:

#if (High[i-2]-High[i]>0 or High[i-1]-High[i]>=0) and (High[i-3]-High[i-1]>0 or High[i-2]-High[i-1]>=0) and up==1:

num2 = num2 + 1

if num2 >=2:

Qs = i + 3

break

else :

if (High[i-2]-High[i]>0 or High[i-1]-High[i]>=0) :#and (High[i-3]-High[i-1]>0 or High[i-2]-High[i-1]>=0):

num1 = num1 + 1

if num1 >=2:

up = 1

#print(i)

if High[i-1]-High[i]<0 and up == 1:

#if (High[i-2]-High[i]<0 or High[i-1]-High[i]<=0) and (High[i-3]-High[i-1]<0 or High[i-2]-High[i-1]<=0) and up==1:

num2 = num2 + 1

if num2 >=2:

Qs = i + 3

break

print("Qs: "+ str(Qs))

num1 = 0

num2 = 0

up = 0

#print(Qs)

#Qs = int(np.median(Qs))

'''''''''''''''

up

Find Qt

'''''''''''''''

K1 = 0

K2 = 0

K3 = 0

M = 0

Qt = 0

'''

if Q\_final < Qs:

Qt = Q\_final

Q\_final = Q\_index

'''

#else:

for i in range(Qs , Qs - 10 ,-1):

K1 = abs(High[i] - High[i-1]) + abs(High[i-1] - High[i-2]) + abs(High[i-2] - High[i-3]) + abs(High[i-3] - High[i-4]) #+ abs(High[i-4] - High[i-5])

K2 = abs(High[i-1] - High[i-2]) + abs(High[i-2] - High[i-3]) + abs(High[i-3] - High[i-4]) + abs(High[i-4] - High[i-5])

#K3 = abs(High[i-2] - High[i-3]) + abs(High[i-3] - High[i-4]) + abs(High[i-4] - High[i-5]) + abs(High[i-5] - High[i-6])

#print("K1:"+str(K1))

#print("K2:"+str(K2))

#print("K3:"+str(K3))

#if K >=5 :

#M = M + 1

#if M >= 5 :

if K1 <= 2 and K2 <= 1 :#and K3 <= 1:

Qt = i

#print('Hello')

break

#elif K <= 1 :

# Qt = i

# #print('Hello')

'''

# break

if i == (Qs - 9):

for i in range(Qs, Qs - 20 ,-1):

K1 = abs(High[i] - High[i-1]) + abs(High[i-1] - High[i-2]) + abs(High[i-2] - High[i-3])# + abs(High[i-3] - High[i-4])

#print(K1)

if K1 <= 1:

Qt = i

break

'''

if Qt == 0:

for i in range(Qs, Qs - 10 ,-1):

K1 = abs(High[i] - High[i-1]) + abs(High[i-1] - High[i-2]) + abs(High[i-2] - High[i-3]) + abs(High[i-3] - High[i-4]) #+ abs(High[i-4] - High[i-5])

K2 = abs(High[i-1] - High[i-2]) + abs(High[i-2] - High[i-3]) + abs(High[i-3] - High[i-4]) + abs(High[i-4] - High[i-5])

#K3 = abs(High[i-2] - High[i-3]) + abs(High[i-3] - High[i-4]) + abs(High[i-4] - High[i-5]) + abs(High[i-5] - High[i-6])

#if K >=5 :

#M = M + 1

#if M >= 5 :

if K1 <= 2 and K2 <= 2 :#and K3 <= 1:

Qt = i

#print('Hello')

break

#elif K <= 1 :

# Qt = i

# #print('Hello')

# break

if Qt == 0:

for i in range(Qs, Qs - 10 ,-1):

K1 = abs(High[i] - High[i-1]) + abs(High[i-1] - High[i-2]) + abs(High[i-2] - High[i-3]) + abs(High[i-3] - High[i-4]) #+ abs(High[i-4] - High[i-5])

K2 = abs(High[i-1] - High[i-2]) + abs(High[i-2] - High[i-3]) + abs(High[i-3] - High[i-4]) + abs(High[i-4] - High[i-5])

#K3 = abs(High[i-2] - High[i-3]) + abs(High[i-3] - High[i-4]) + abs(High[i-4] - High[i-5]) + abs(High[i-5] - High[i-6])

#if K >=5 :

#M = M + 1

#if M >= 5 :

if K1 <= 3 and K2 <= 2 :#and K3 <= 2:

Qt = i

#print('Hello')

break

#elif K <= 1 :

# Qt = i

# #print('Hello')

# break

if Qt == 0:

for i in range(Qs, Qs - 10 ,-1):

K1 = abs(High[i] - High[i-1]) + abs(High[i-1] - High[i-2]) + abs(High[i-2] - High[i-3]) + abs(High[i-3] - High[i-4]) #+ abs(High[i-4] - High[i-5])

K2 = abs(High[i-1] - High[i-2]) + abs(High[i-2] - High[i-3]) + abs(High[i-3] - High[i-4]) + abs(High[i-4] - High[i-5])

#K3 = abs(High[i-2] - High[i-3]) + abs(High[i-3] - High[i-4]) + abs(High[i-4] - High[i-5]) + abs(High[i-5] - High[i-6])

#if K >=5 :

#M = M + 1

#if M >= 5 :

if K1 <= 3 and K2 <= 3:# and K3 <= 2:

Qt = i

#print('Hello')

break

#elif K <= 1 :

# Qt = i

# #print('Hello')

if Qt == 0:

for i in range(Qs, Qs - 10 ,-1):

K1 = abs(High[i] - High[i-1]) + abs(High[i-1] - High[i-2]) + abs(High[i-2] - High[i-3]) + abs(High[i-3] - High[i-4]) #+ abs(High[i-4] - High[i-5])

K2 = abs(High[i-1] - High[i-2]) + abs(High[i-2] - High[i-3]) + abs(High[i-3] - High[i-4]) + abs(High[i-4] - High[i-5])

if K1 <= 4 and K2 <= 3:# and K3 <= 2:

Qt = i

break

if Qt == 0:

for i in range(Qs, Qs - 10 ,-1):

K1 = abs(High[i] - High[i-1]) + abs(High[i-1] - High[i-2]) + abs(High[i-2] - High[i-3]) + abs(High[i-3] - High[i-4]) #+ abs(High[i-4] - High[i-5])

K2 = abs(High[i-1] - High[i-2]) + abs(High[i-2] - High[i-3]) + abs(High[i-3] - High[i-4]) + abs(High[i-4] - High[i-5])

if K1 <= 4 and K2 <= 4:# and K3 <= 2:

Qt = i

break

if Qt == 0:

for i in range(Qs, Qs - 10 ,-1):

K1 = abs(High[i] - High[i-1]) + abs(High[i-1] - High[i-2]) + abs(High[i-2] - High[i-3]) + abs(High[i-3] - High[i-4]) #+ abs(High[i-4] - High[i-5])

K2 = abs(High[i-1] - High[i-2]) + abs(High[i-2] - High[i-3]) + abs(High[i-3] - High[i-4]) + abs(High[i-4] - High[i-5])

if K1 <= 5 and K2 <= 4:# and K3 <= 2:

Qt = i

break

if Qt == 0:

for i in range(Qs, Qs - 10 ,-1):

K1 = abs(High[i] - High[i-1]) + abs(High[i-1] - High[i-2]) + abs(High[i-2] - High[i-3]) + abs(High[i-3] - High[i-4]) #+ abs(High[i-4] - High[i-5])

K2 = abs(High[i-1] - High[i-2]) + abs(High[i-2] - High[i-3]) + abs(High[i-3] - High[i-4]) + abs(High[i-4] - High[i-5])

if K1 <= 5 and K2 <= 5:# and K3 <= 2:

Qt = i

break

M = 0

K1 = 0

K2 = 0

K3 = 0

###############################

######## new feature ##########

###############################

if Qs <= 0 :#or Q\_final == 0: or S\_final == 0:

for i in range(Q\_final , S\_final):

area = area + (High[i] - High[S\_final])

square =+ 1

if k == 1:

I\_max = np.max(High[Q\_final:S\_final]) - High[S\_final]

if k == 6:

PvN = (np.max(High[Q\_final:S\_final])- High[S\_final]) / (np.min(High[Q\_final:S\_final]) - High[S\_final])

V1\_max = np.max(High[Q\_final:S\_final]) - High[S\_final]

###################################################

if k == 4:

aVL\_max = np.max(High[Q\_final:S\_final]) - High[S\_final]

'''

print("Q: "+str(Q\_final)+"; S: "+str(S\_final))

T1 = S\_final - Q\_final

print("T1: "+str(T1))

if T1 >= 20 :

print('The picture ' + name[k] + ' has something wrong with QS distance.')

Positive = 1

'''

else:

if k < 6 :

print("Qt: "+str(Qt)+"; Q: "+str(Q\_final)+"; S: "+str(S\_final))

print("Q\_index: "+str(Q\_index)+"; S\_index: "+str(S\_index))

T1 = S\_final - Q\_final

T2 = Q\_final - Qt

T1\_sum = T1\_sum + T1

T2\_sum = T2\_sum + T2

curve = curve + 1

print("T1: "+str(T1))

print("T2: "+str(T2))

for i in range(Q\_final , S\_final):

area = area + (High[i] - High[S\_final])

square =+ 1

if k == 1:

I\_max = np.max(High[Q\_final:S\_final]) - High[S\_final]

if k == 6:

PvN = (np.max(High[Q\_final:S\_final])- High[S\_final]) / (np.min(High[Q\_final:S\_final]) - High[S\_final])

V1\_max = np.max(High[Q\_final:S\_final]) - High[S\_final]

if k == 4:

aVL\_max = np.max(High[Q\_final:S\_final]) - High[S\_final]

if (T1 >= 15) and (T2 <= 15):

#print('The picture ' + name[k] + ' has something wrong with QS distance and platform distance.')

Positive = 1

#elif (T1 >= 15):

#print('The picture ' + name[k] + ' has something wrong with QS distance.')

T1\_final = T1\_sum / curve

T2\_final = T2\_sum / curve

'''

print("T1\_final: "+str(T1\_final))

print("T2\_final: "+str(T2\_final))

'''

I\_minus = I\_max - V1\_max

II\_minus = aVL\_max - V1\_max

III\_minus = I\_max + aVL\_max - V1\_max

'''

print("Positive: "+str(Positive))

print("I\_max: "+str(I\_max))

print("aVL\_max: "+str(aVL\_max))

print("V1\_max: "+str(V1\_max))

print("I\_minus: "+str(I\_minus))

print("II\_minus: "+str(II\_minus))

print("III\_minus: "+str(III\_minus))

print("II: "+str(II))

print("III: "+str(III))

'''

print("V1: "+str(V1))

print("V2: "+str(V2))

print("V3: "+str(V3))

print("V4: "+str(V4))

print("V5: "+str(V5))

print("V6: "+str(V6))

if Positive == 1:

if V1==0 and V2==0 and V5==1 and V6==1 :

print ("Must B")

elif ((V1==0) + (V2==0) + (V5==1) + (V6==1))/4 >= ((V1+V2+V3+V4+V5+V6)/6):

print("Maybe B")

else :

print("Maybe A")

else:

print("None")

'''

V1\_down = 0

V2\_down = 0

V5\_up = 0

V6\_up = 0

V\_count\_up = 0

V\_count\_down = 0

'''