In [50]:
```
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.preprocessing.image import load_img, img_to_array
import matplotlib.pyplot as plt
import tensorflow as tf
import numpy as np
import cv2
import os
```

In [51]:
```
img = load_img(r'C:\Users\lenovo\Desktop\Training\Happy\8.jpeg', target_size=(22
img_array = img_to_array(img)  # Convert to NumPy array
print(img_array.shape)
```

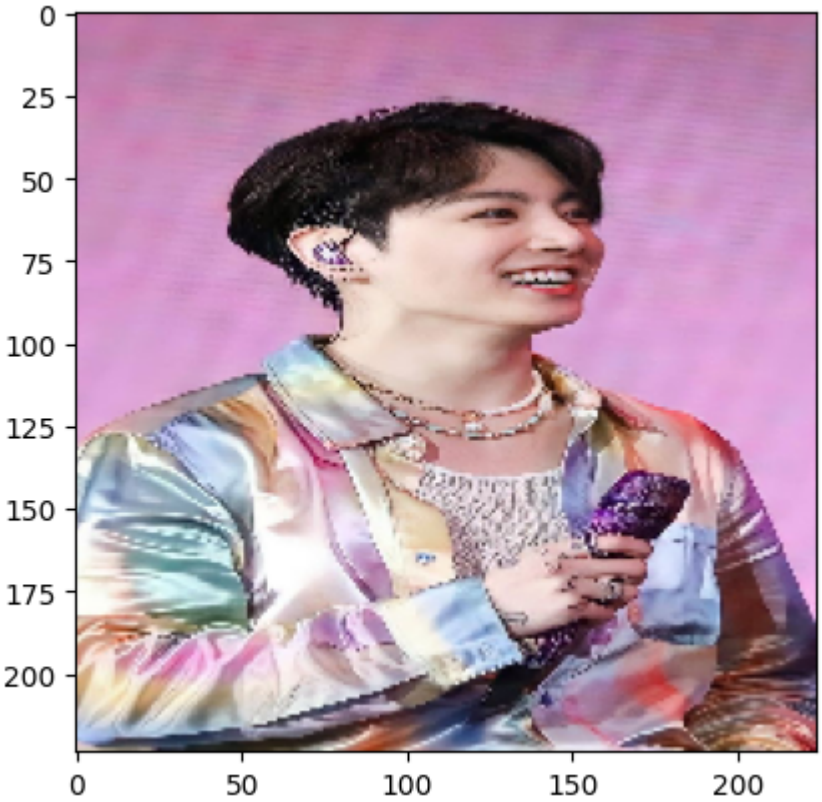
```
(224, 224, 3)
```

In [52]:
```
img
```

Out[52]:

In [53]:
```
plt.imshow(img)
```

Out[53]:
```
<matplotlib.image.AxesImage at 0x266ff646750>
```



In [54]:
```
i1 = cv2.imread(r'C:\Users\lenovo\Desktop\Training\Happy\8.jpeg')
i1
```

Out[54]:
```
array([[[170, 126, 197],
        [169, 125, 196],
        [169, 125, 196],
        ...,
        [164, 120, 187],
        [164, 120, 187],
        [164, 120, 187]],

       [[168, 124, 195],
        [168, 124, 195],
        [167, 123, 194],
        ...,
        [164, 120, 187],
        [164, 120, 187],
        [164, 120, 187]],

       [[166, 122, 193],
        [166, 122, 193],
        [166, 122, 193],
        ...,
        [163, 119, 186],
        [164, 120, 187],
        [164, 120, 187]],

       ...,

       [[204, 210, 223],
        [203, 209, 220],
        [200, 208, 221],
        ...,
        [ 22,  13, 117],
        [ 32,  23, 127],
        [ 36,  27, 131]],

       [[204, 210, 221],
        [203, 210, 219],
        [200, 209, 219],
        ...,
        [ 25,  19, 120],
        [ 32,  24, 125],
        [ 32,  24, 125]],

       [[204, 211, 220],
        [203, 210, 219],
        [200, 209, 218],
        ...,
        [ 36,  31, 130],
        [ 42,  34, 135],
        [ 42,  34, 135]]], dtype=uint8)
```

In [55]:
```
# The above output is an nd array.
```

In [56]:
```
i1.shape
```

Out[56]:
```
(1308, 736, 3)
```

In [57]:
```
# The above output (Height-1308, Width-736, 3D channel(RGB)- 3)
```

In [58]:
```
train = ImageDataGenerator(rescale = 1/200)
validation = ImageDataGenerator(rescale = 1/200)
```

In [59]:
```
train_dataset = train.flow_from_directory(r'C:\Users\lenovo\Desktop\Training',
                                          target_size=(200, 200),
                                          batch_size = 32,
                                          class_mode = 'binary')
validation_dataset = validation.flow_from_directory(r'C:\Users\lenovo\Desktop\Va
                                          target_size=(200, 200),
                                          batch_size = 32,
                                          class_mode = 'binary')
```

```
Found 14 images belonging to 2 classes.
Found 0 images belonging to 2 classes.
```

In [60]:
```
train_dataset.class_indices
```

Out[60]:
```
{'Happy': 0, 'Sad': 1}
```

In [61]:
```
train_dataset.classes
```

Out[61]:
```
array([0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1])
```

In [62]:
```
# In train_dataset, we have 7- Happy Images(named as- 0), 7- Sad Images(named as
```

In [63]:
```
# We are applying max pooling now:
model = tf.keras.models.Sequential([
    tf.keras.layers.Input(shape=(200, 200, 3)),  # Define the input shape explic
    tf.keras.layers.Conv2D(16, (3, 3), activation='relu'),
    tf.keras.layers.MaxPool2D(2, 2),
    #
    tf.keras.layers.Conv2D(32, (3, 3), activation='relu'),
    tf.keras.layers.MaxPool2D(2, 2),
    #
    tf.keras.layers.Conv2D(64, (3, 3), activation='relu'),
    tf.keras.layers.MaxPool2D(2, 2),
    ##
    tf.keras.layers.Flatten(),
    ##
    tf.keras.layers.Dense(512, activation='relu'),
    #
    tf.keras.layers.Dense(1, activation='sigmoid'),
])
```

In [64]:
```
model.compile(loss = 'binary_crossentropy',
                     optimizer = tf.keras.optimizers.RMSprop(learning_rate = 0.
                     metrics = ['accuracy'])
```

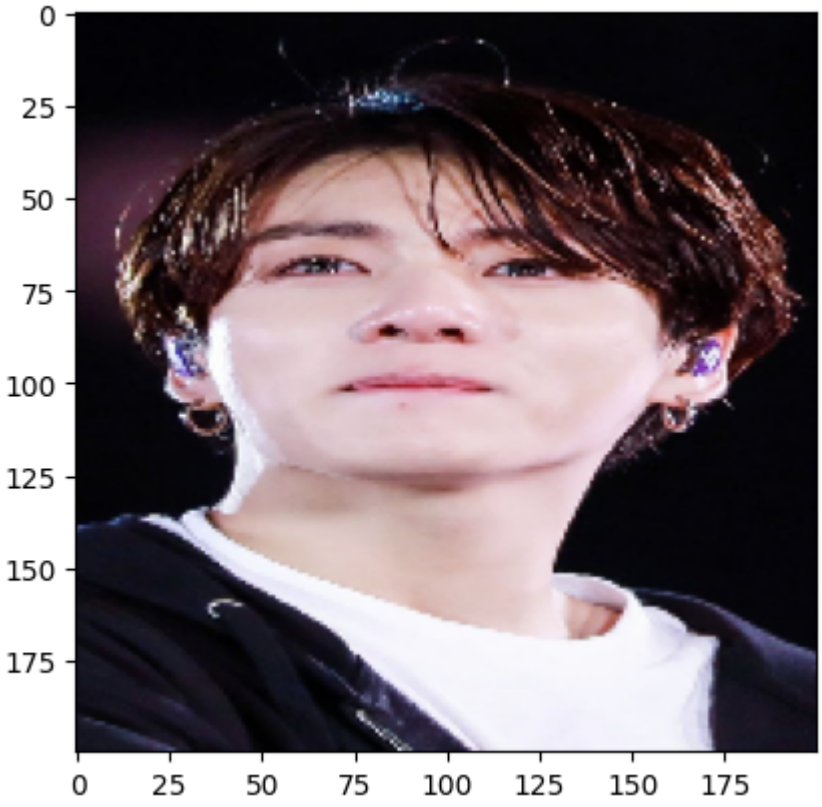
In [66]:
```
history = model.fit(train_dataset, epochs = 9)
```

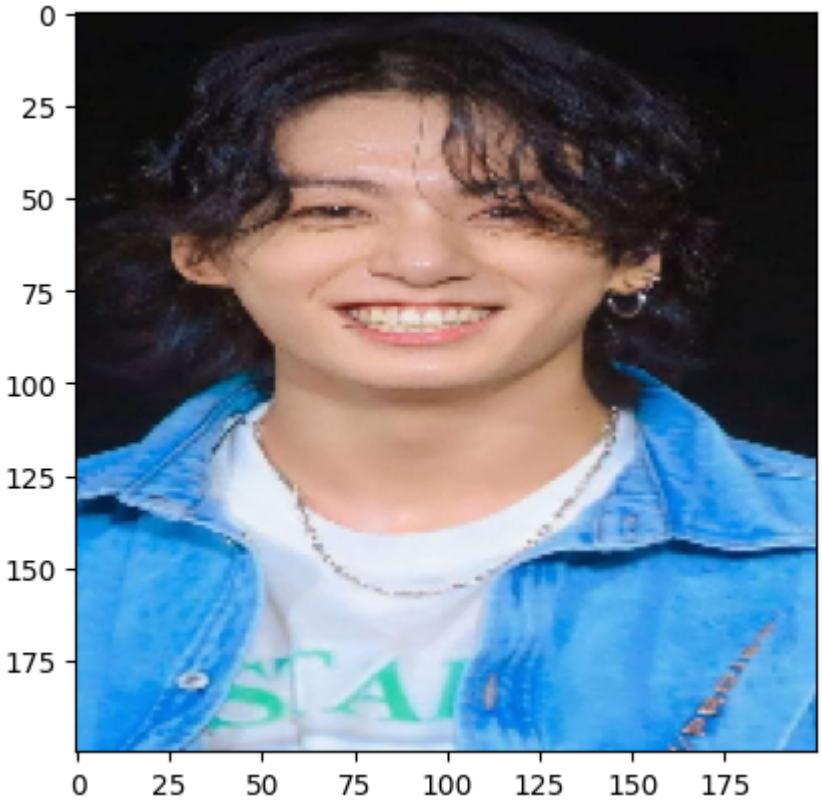
```
Epoch 1/9
1/1 ━━━━━━━━━━━━━━━━━━━━ 1s 780ms/step - accuracy: 0.5000 - loss: 0.6334
Epoch 2/9
1/1 ━━━━━━━━━━━━━━━━━━━━ 1s 790ms/step - accuracy: 1.0000 - loss: 0.2741
Epoch 3/9
1/1 ━━━━━━━━━━━━━━━━━━━━ 1s 569ms/step - accuracy: 1.0000 - loss: 0.1793
Epoch 4/9
1/1 ━━━━━━━━━━━━━━━━━━━━ 1s 980ms/step - accuracy: 1.0000 - loss: 0.1323
Epoch 5/9
1/1 ━━━━━━━━━━━━━━━━━━━━ 1s 1s/step - accuracy: 1.0000 - loss: 0.0990
Epoch 6/9
1/1 ━━━━━━━━━━━━━━━━━━━━ 1s 901ms/step - accuracy: 1.0000 - loss: 0.0754
Epoch 7/9
1/1 ━━━━━━━━━━━━━━━━━━━━ 3s 3s/step - accuracy: 1.0000 - loss: 0.0584
Epoch 8/9
1/1 ━━━━━━━━━━━━━━━━━━━━ 1s 1s/step - accuracy: 1.0000 - loss: 0.0463
Epoch 9/9
1/1 ━━━━━━━━━━━━━━━━━━━━ 1s 688ms/step - accuracy: 1.0000 - loss: 0.0373
```

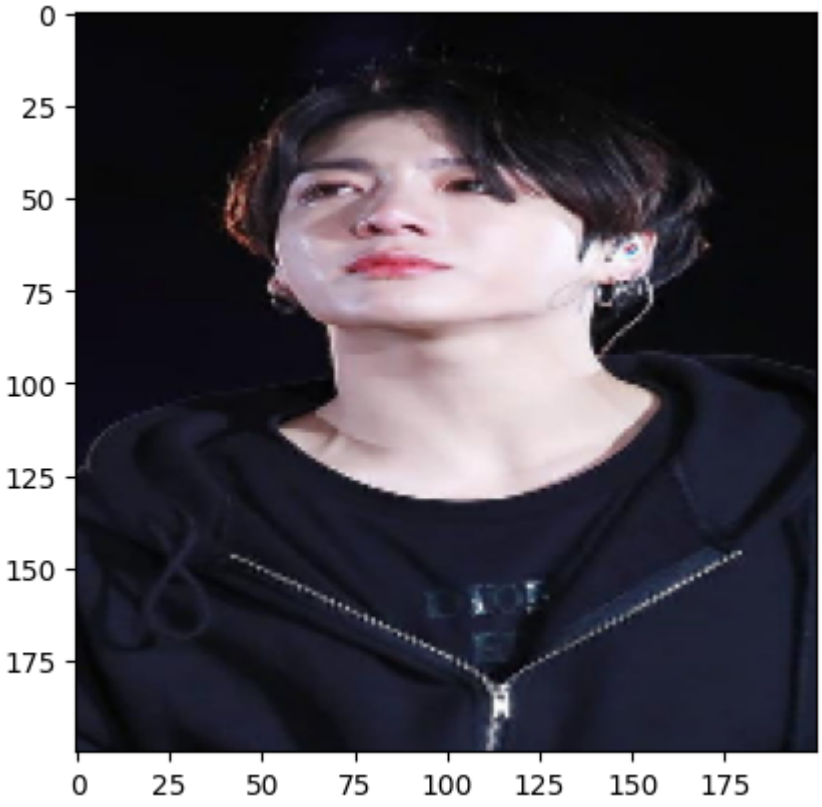
In [68]:
```
dir_path = r'C:\Users\lenovo\Desktop\Testing'
for i in os.listdir(dir_path):
    print(i)
```

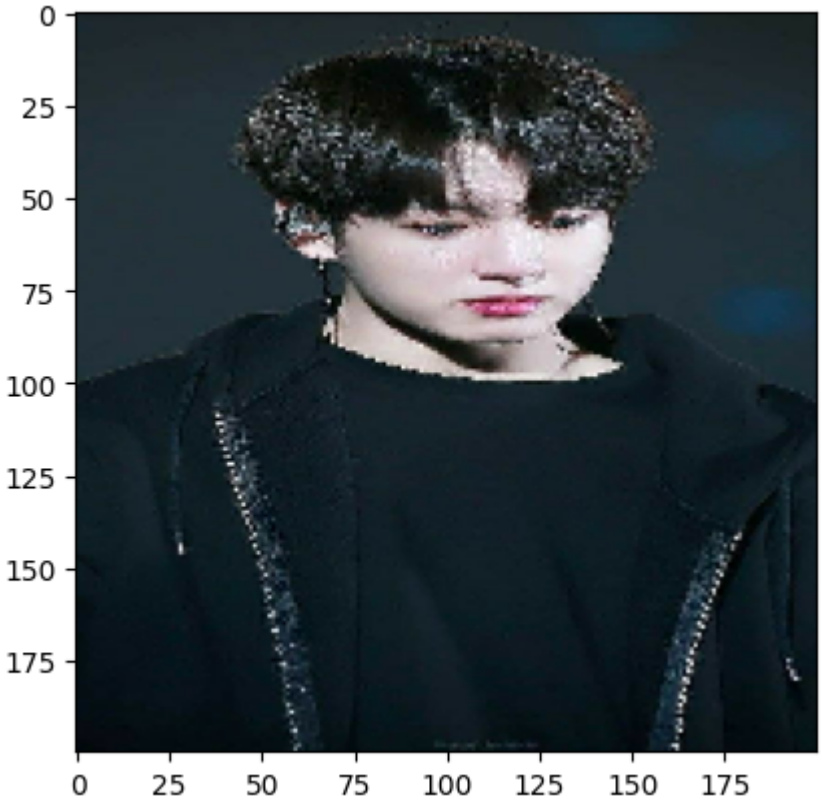
```
1.jpeg
2.jpeg
3.jpeg
4.jpeg
5.jpeg
6.jpeg
7.jpeg
8.jpeg
```

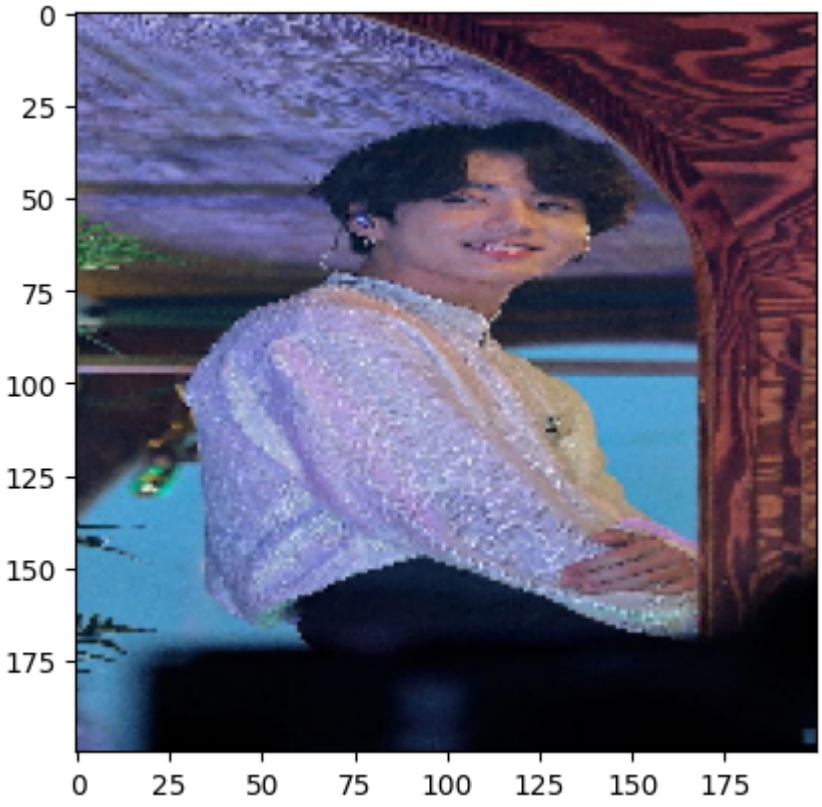
In [69]:
```
dir_path = r'C:\Users\lenovo\Desktop\Testing'
for i in os.listdir(dir_path):
    img = load_img(dir_path+ '//' +i, target_size = (200,200))
    plt.imshow(img)
    plt.show()
```

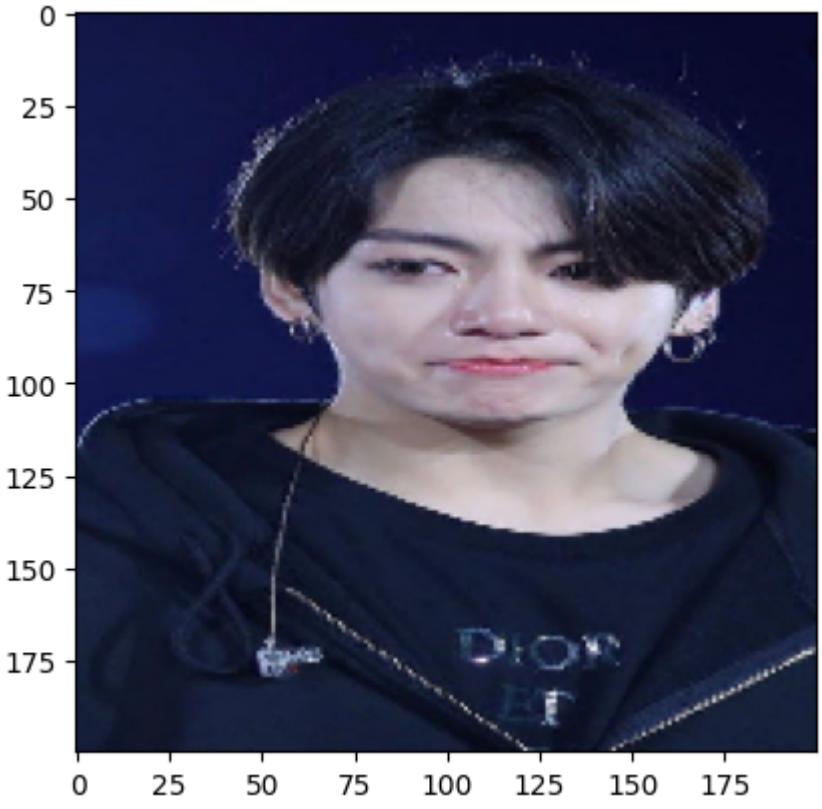

0
25
50
75
100
125
150
175
0
25
50
75
100
125
150
175

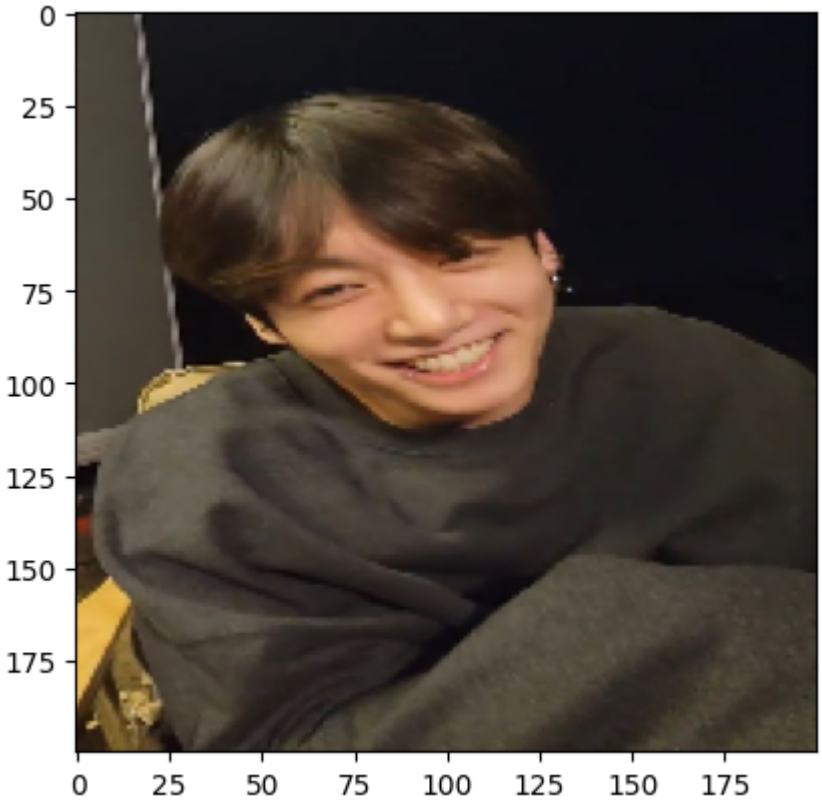

0
25
50
75
100
125
150
175
0
25
50
75
100
125
150
175

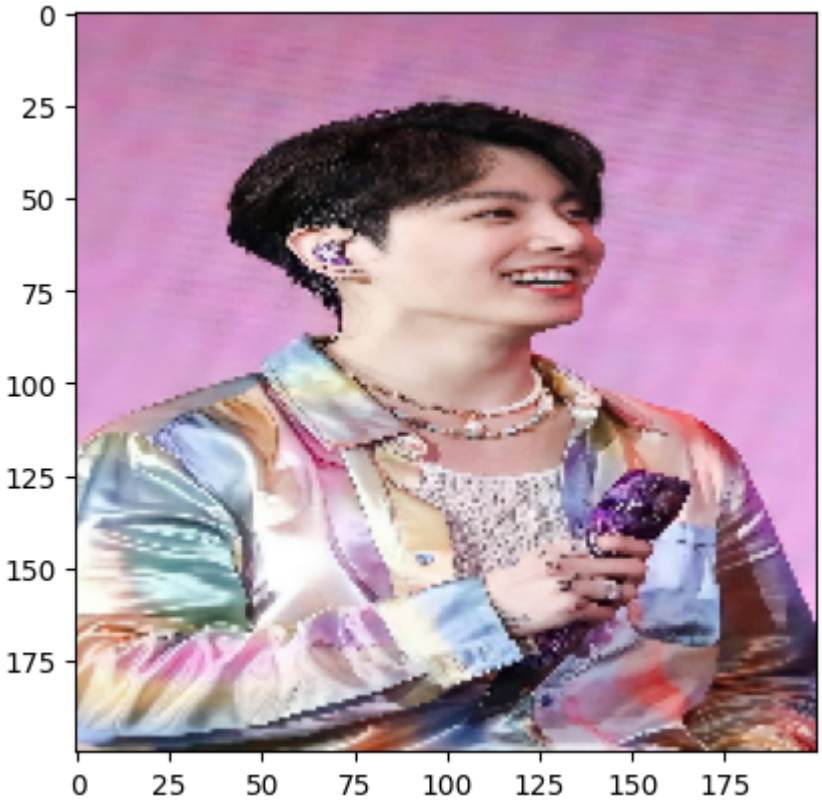

0
25
50
75
100
125
150
175
0
25
50
75
100
125
150
175


0
25
50
75
100
125
150
175
0
25
50
75
100
125
150
175


0
25
50
75
100
125
150
175
0 25 50 75 100 125 150 175


0
25
50
75
100
125
150
175
0 25 50 75 100 125 150 175

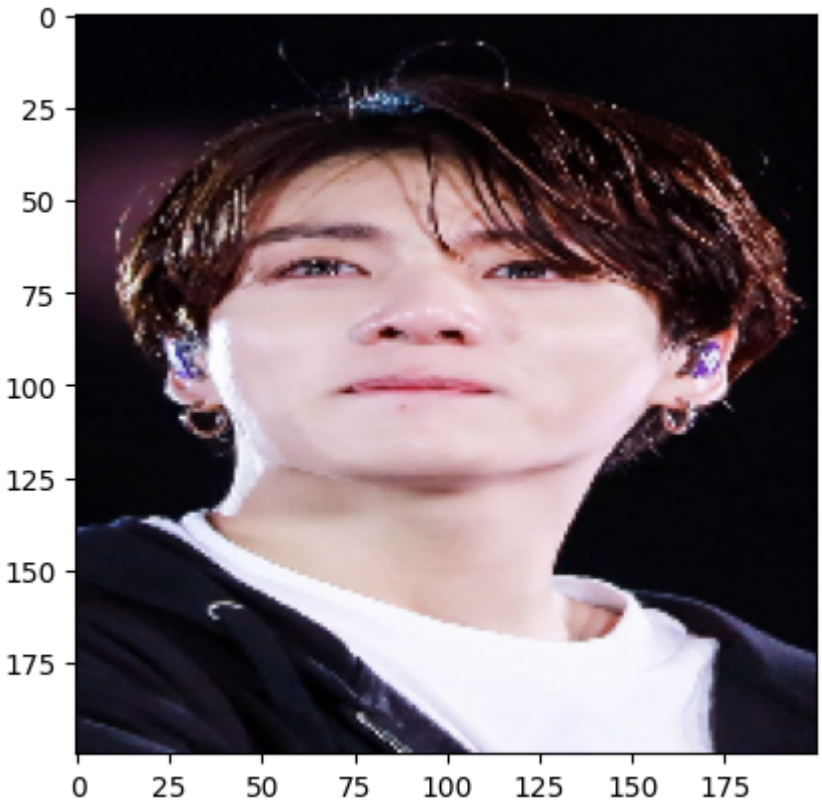




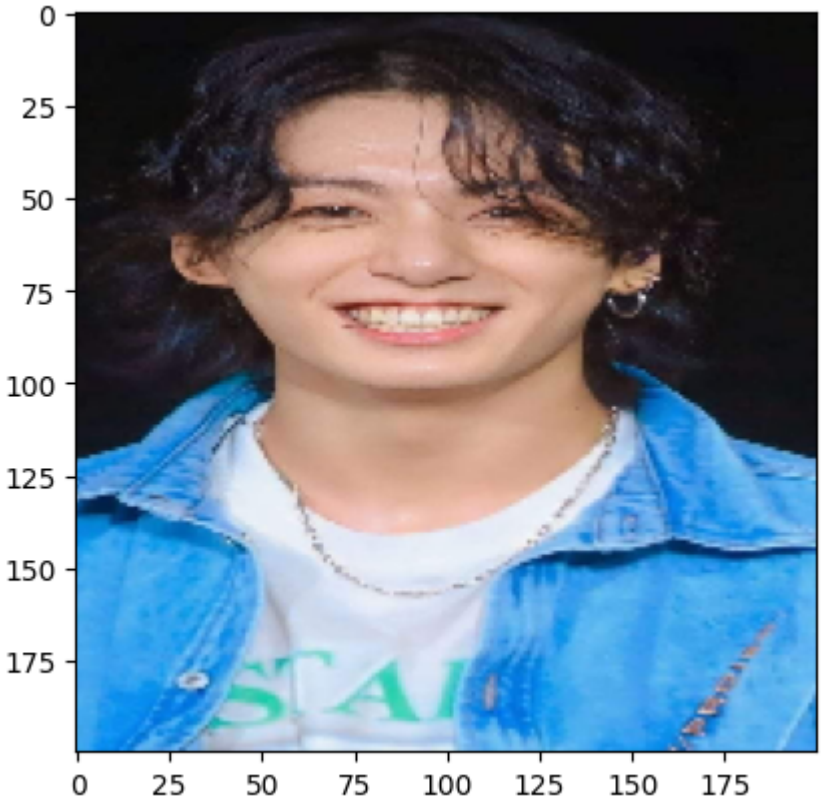
In [70]:
```
dir_path = r'C:\Users\lenovo\Desktop\Testing'
for i in os.listdir(dir_path):
    img = load_img(dir_path+ '//' +i, target_size = (200,200))
    plt.imshow(img)
    plt.show()

    x = img_to_array(img)
    x = np.expand_dims(x, axis = 0)
    images = np.vstack([x])
```

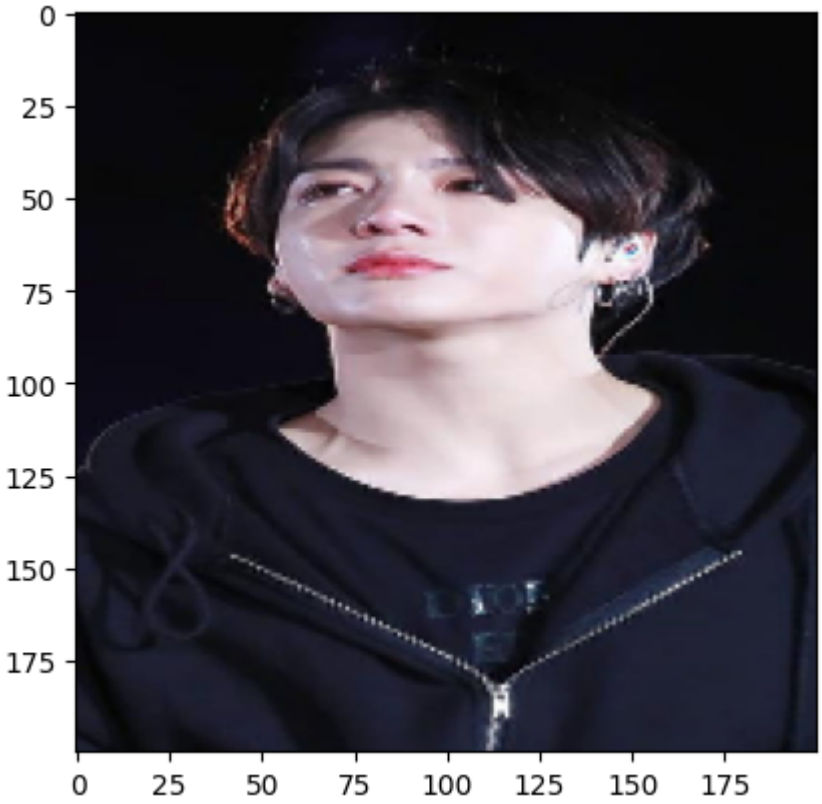
```
val = model.predict(images)
if val == 0:
    print('i am happy')
else:
    print('i am sad')
```



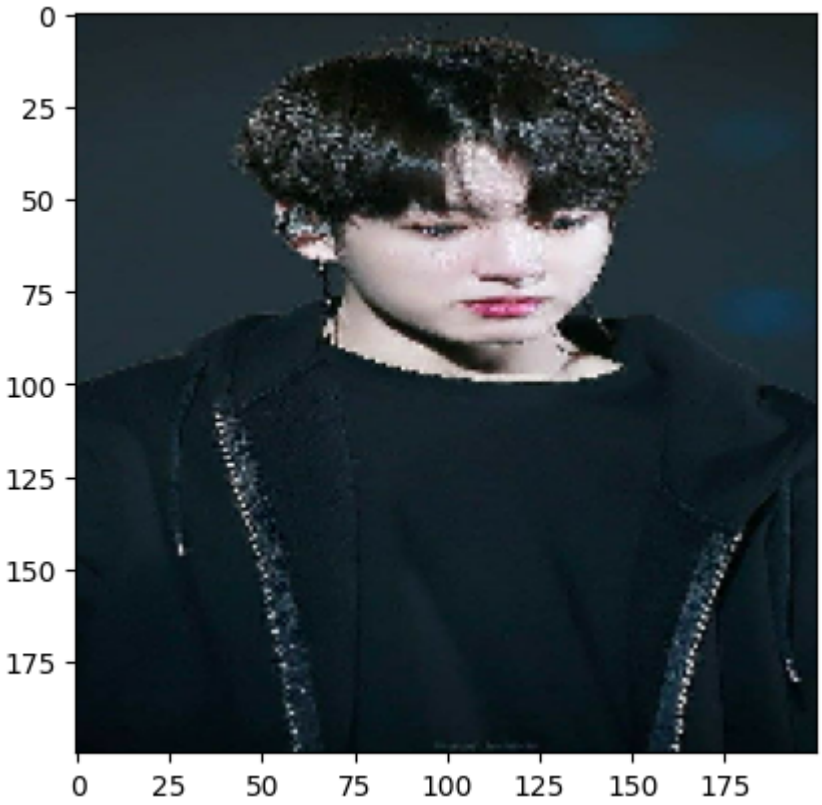

```
1/1 ━━━━━━━━━━━━━━━━━━━━ 6s 6s/step
i am sad
```



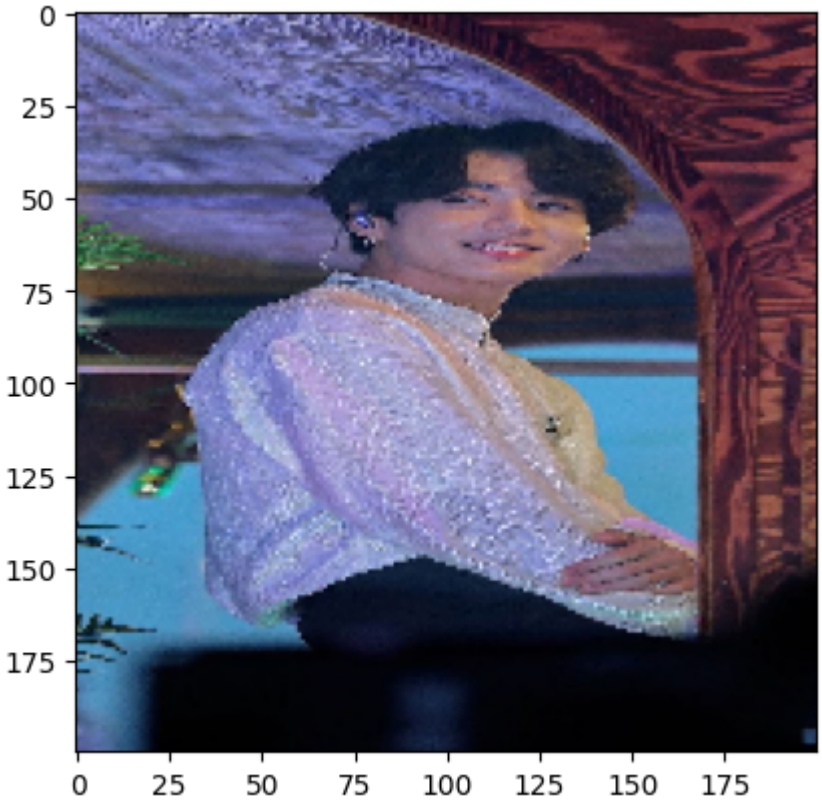

```
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 358ms/step
i am happy
```



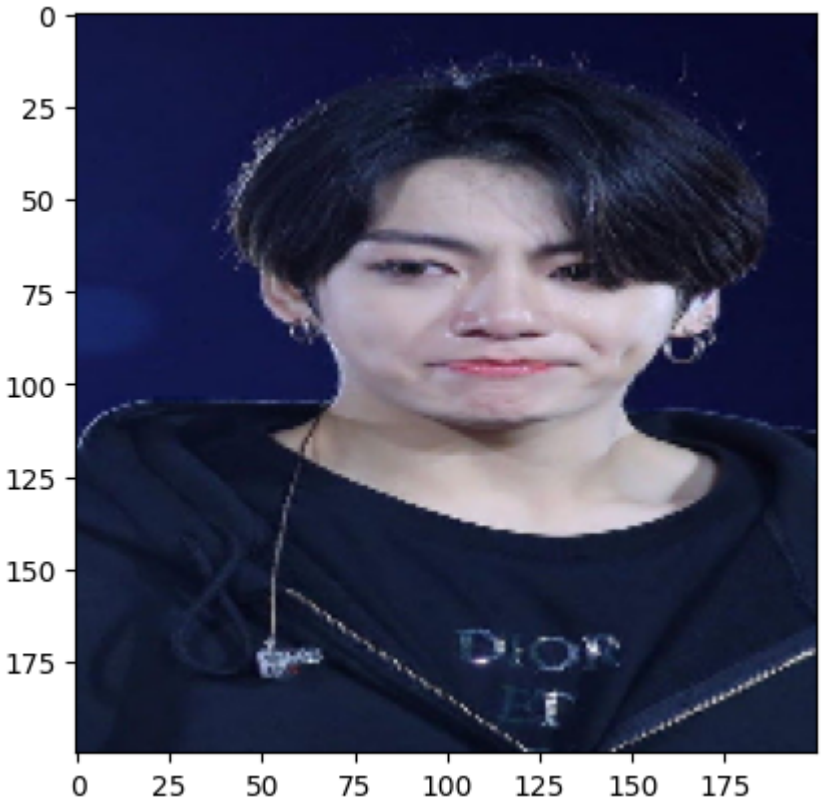

```
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 404ms/step
i am sad
```



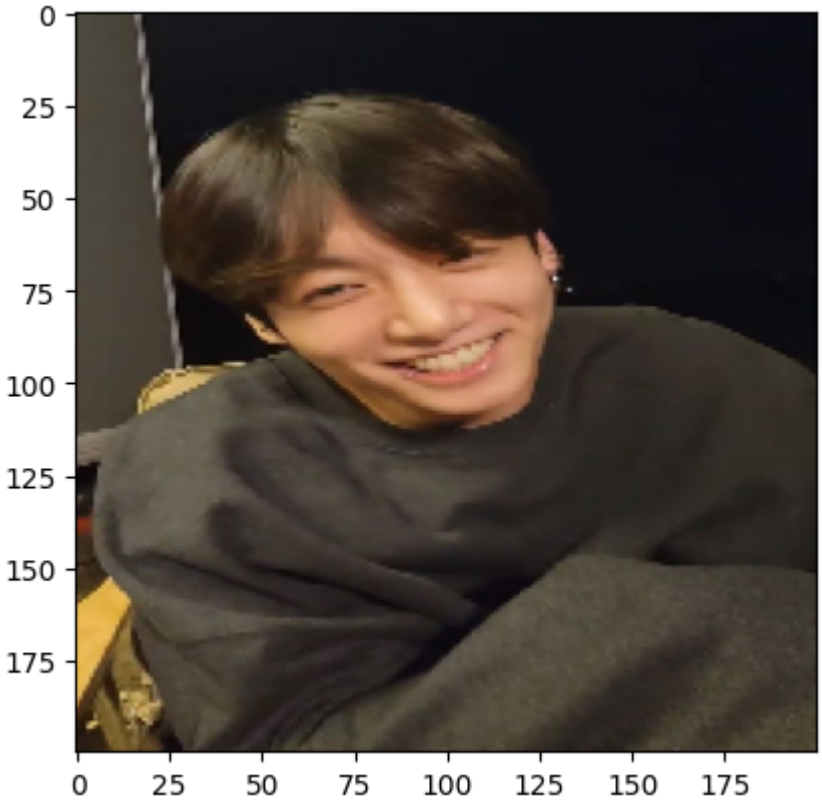

```
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 158ms/step
i am sad
```



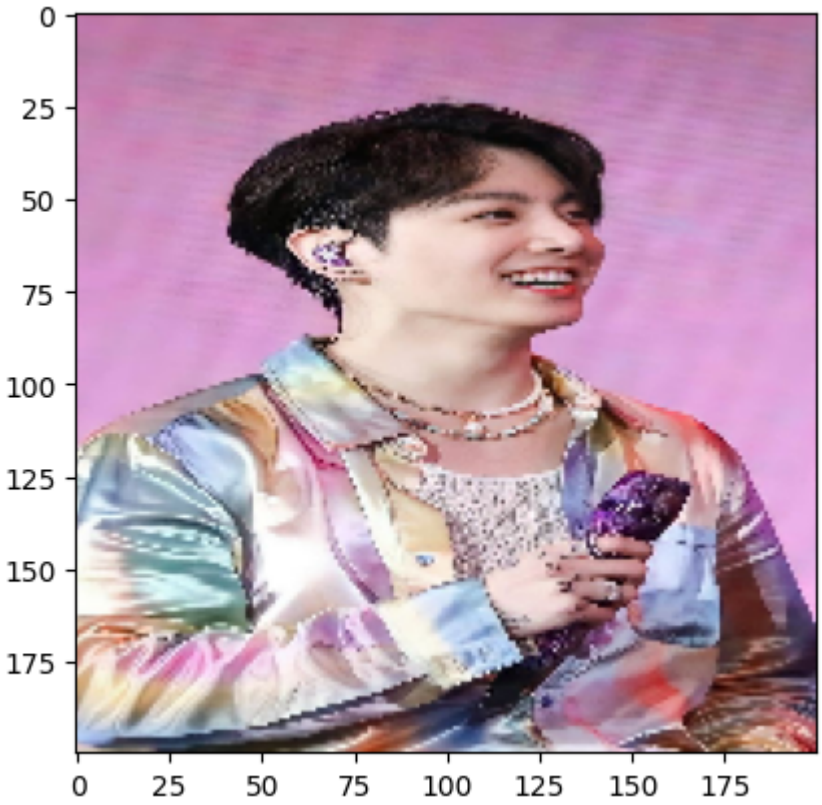

```
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 170ms/step
i am happy
```




```
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 140ms/step
i am sad
```




```
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 56ms/step
i am happy
```




```
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 195ms/step
i am happy
```