In [2]:
```
import cv2
import glob
```

In [3]:
```
files = glob.glob('*.avif')
print(files)
```

```
[]
```

In [4]:
```
my_list = []
path = "*.*"
for file in glob.glob(path):
    print(file)
    a = cv2.imread(file)
    my_list.append(a)
```

```
DIP_lab.ipynb
download (2).jpg
download.jpg
pic.jpg
```

In [5]:
```
from matplotlib import pyplot as plt
```

In [6]:
```
plt.imshow(my_list[1])
# plt.imshow(my_list[2])
# plt.imshow(my_list[3])
```

Out[6]:
```
<matplotlib.image.AxesImage at 0x2393054ffd0>
```

Load Image

In [10]:
```
import cv2
a = cv2.imread("pic.jpg")
img_number = 1

cv2.imwrite("Color_image"+str(img_number)+".jpg",a)
```

Out[10]: True

In [12]:
```
import os
import cv2

# Path to the directory containing images
directory = r'G:\picture'

# Get a list of all files in the directory
files = os.listdir(directory)

# Counter for naming output files
img_number = 1

# Loop through each file in the directory
for file in files:
    # Check if the file is an image
    if file.endswith(".jpg") or file.endswith(".jpeg") or file.endswith(".png")
        # Read the image
        img = cv2.imread(os.path.join(directory, file))

        # Process the image (replace this with your processing code)
        # For example, you can apply some operations on the image

        # Write the processed image to a new file
        output_file = os.path.join(directory, "Color_image" + str(img_number) +
        cv2.imwrite(output_file, img)

        # Increment the image number counter
        img_number += 1

print("All images processed and saved.")
```

```
All images processed and saved.
```

Next topic

In [ ]:
```
```

In [16]:
```
import os
import cv2

# Path to the directory containing images
directory = r'G:\picture'

# Get a list of all files in the directory
files = os.listdir(directory)

# Counter for naming output files
img_number = 1

# Loop through each file in the directory
for file in files:
    # Check if the file is an image
    if file.endswith(".jpg") or file.endswith(".jpeg") or file.endswith(".png")
        # Read the image
        img = cv2.imread(os.path.join(directory, file))

        # Convert BGR to RGB
        img_rgb = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)

        # Process the image (if needed)

        # Write the processed image to a new file
        output_file = os.path.join(directory, "Color_Img" + str(img_number) + '
        cv2.imwrite(output_file, img_rgb)

        # Increment the image number counter
        img_number += 1

print("All images processed and saved as RGB.")
```

```
All images processed and saved as RGB.
```

next topic

In [18]:
```
#opor er code
for file in glob.glob(path):
        print(file)
        a=cv2.imread(file)
        print(a)
cv2.imshow('Original Image',a)
cv2.waitKey(0)
cv2.destroyAllWindows()

c=cv2.cvtColor(a,cv2.COLOR_BGR2RGB)
cv2.imwrite("color range"+str(img_number)+".jpg",c)
img_number+=1
cv2.imshow('Color Image',c)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

```
  [255 255 255]
  ...
  [255 255 255]
  [255 255 255]
  [255 255 255]]

 [[255 255 255]
  [255 255 255]
  [255 255 255]
  ...
  [255 255 255]
  [255 255 255]
  [255 255 255]]

 [[255 255 255]
  [255 255 255]
  [255 255 255]
  ...
  [255 255 255]
  [255 255 255]
```

In [19]:
```
import os
path = r'G:\picture'
print(os.listdir(path))
```

```
['Color_Img1.jpg', 'Color_Img2.jpg', 'Color_Img3.jpg', 'pic1.jpg', 'pic2.jp
g', 'pic3.jpg']
```

In [21]:
```
for l in os.listdir(path):
    print(l)
```

```
Color_Img1.jpg
Color_Img2.jpg
Color_Img3.jpg
pic1.jpg
pic2.jpg
pic3.jpg
```

os.walk() function

In [22]:
```
import os
print(os.walk(".")) #noting to see here as this is just a generator object
```

```
<generator object _walk at 0x00000239304FCBC0>
```

In [24]:
```
#traverse root directory and list directories as dirs and files as files
for root, dirs, files in os.walk("."):
#     print(root) #print root directory name

    path = root.split(os.sep) #split at separator (\ or /)
    #print(path) # gives names of directores for easy location of files
    #print(files)  #prints all files names in all
```

In [33]:
```
print((len(path)-1) * '---', os.path.basename(root))
for file in files:
    print(len(path) * '---', file)
```

```
--- .ipynb_checkpoints
------ DIP_lab-checkpoint.ipynb
```

In [32]:
```
# shortcut way
import os
for root,dirs, files in os.walk("."):
#for root,subdirs, files in os.walk("."):
    for name in dirs:
        print(os.path.join(root,name)) #will print path of directories
    for name in files:
        print(os.path.join(root,name)) #will print path of files
```

```
.\color range4.jpg
.\color range5.jpg
.\Color_image1.jpg
.\DIP_lab.ipynb
.\download (2).jpg
.\download.jpg
.\pic.jpg
.\.ipynb_checkpoints\DIP_lab-checkpoint.ipynb
```

Image processing using scikit-image library:

In [35]:
```
#interpolation -- zoom korle jno quality nosto na hoy
#3 vabe interpolation kora jay -- nam gulu jante hobe
```

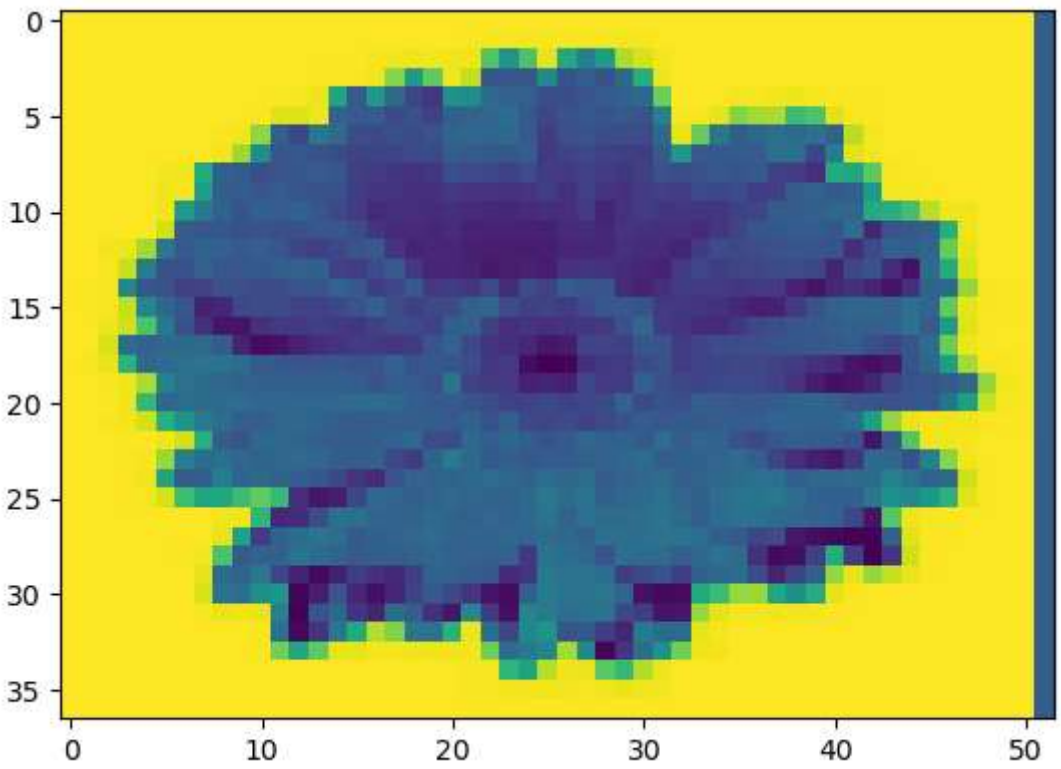
In [37]:
```
import matplotlib.pyplot as plt
from skimage import io, color
from skimage.transform import rescale, resize, downscale_local_mean
```

In [41]:
```
# img = io.imread("images/Osteosarcoma_01.tif", as_gray= True)
img = io.imread("pic.jpg", as_gray= True)
```

In [49]:
```
#Rescale, resize image by a given factor while rescaling image
#gaussian smooting can performed to avoid anti aliasing artifacts.
img_rescale = rescale(img, 1.0 / 4.0, anti_aliasing = False) #anti_aliasing = 

#Resize
img_resized = resize(img,(200,200), anti_aliasing = True)
#resize re
img_downscaled = downscale_local_mean(img,(4,3))
plt.imshow(img_downscaled)
# plt.imshow(img_rescale)
```

Out[49]:
```
<matplotlib.image.AxesImage at 0x23935543c50>
```



In [ ]: