1. Resize the image to half of its original dimensions using image interpolation.

```
[25]: img2 = cv.resize(img,
                       None,
                       fx=0.5,
                       fy=0.5,
                       interpolation=cv.INTER_LINEAR
                      )
```

1

```
[26]: plt.imshow(cv.cvtColor(img2, cv.COLOR_BGR2RGB))
```

```
[26]: <matplotlib.figure.AxesImage at 0x125f79300>
```

2. Convert the image to grayscale.

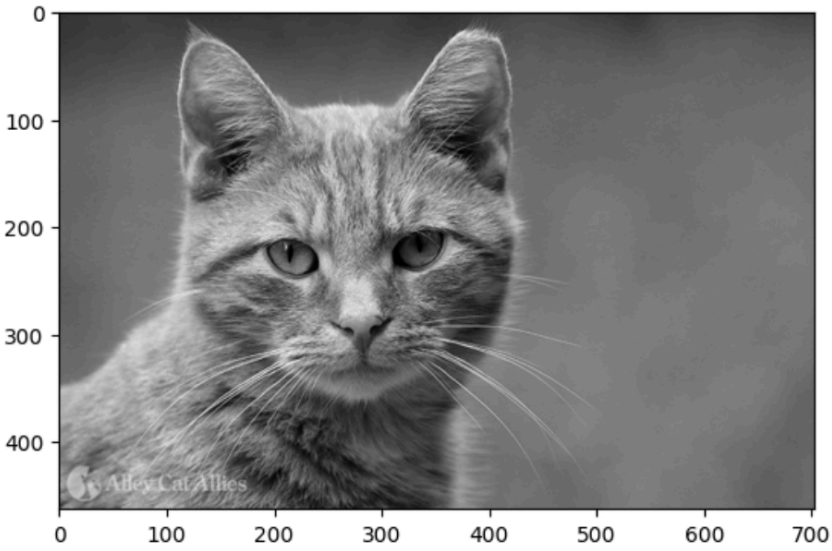
```
[31]: img3 = cv.cvtColor(img, cv.COLOR_BGR2GRAY)
```

```
[35]: cv.imshow('image', img3)

      # waits for user to press any key
      # (this is necessary to avoid Python kernel form crashing)
      cv.waitKey(0)

      # closing all open windows

      cv.destroyAllWindows()

      cv.waitKey(1)
```

```
[35]: -1
```

```
[91]: plt.imshow(cv.cvtColor(img3, cv.COLOR_BGR2RGB))
```

```
[91]: <matplotlib.image.AxesImage at 0x12ce2eb30>
```

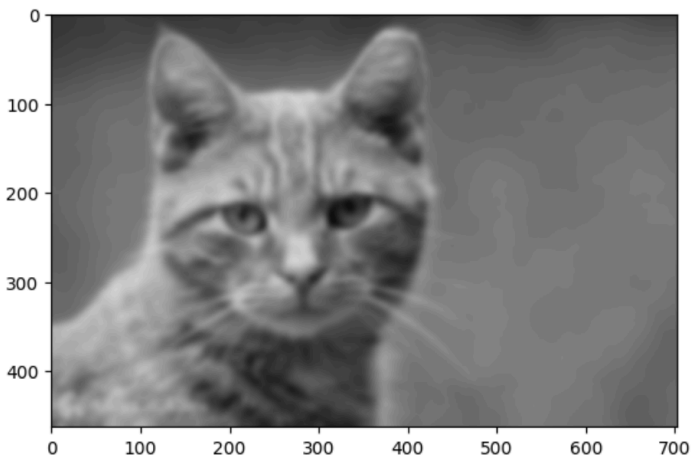


3. Apply Gaussian blur to the grayscale image to reduce noise.

```
[73]: img4 = cv.GaussianBlur(img3, (21, 21), 0)
```

```
[74]: plt.imshow(cv.cvtColor(img4, cv.COLOR_BGR2RGB))
```

```
[74]: <matplotlib.image.AxesImage at 0x126bae4a0>
```



4. Perform edge detection using the Canny edge detection algorithm.

```
[87]: img5 = cv.Canny(img4, 40, 40)
```

```
[88]: plt.imshow(cv.cvtColor(img5, cv.COLOR_BGR2RGB))
```

```
[88]: <matplotlib.image.AxesImage at 0x12cd9f7f0>
```