\*RUN, THIS CODE IN ANY PYTHON IDE WITH PROPER LIBRARIES INTALLED.

**import** PIL

**from** PIL **import** ImageEnhance

**from** PIL **import** ImageFilter

**from** PIL **import** Image

**from** os **import** system

**import** os

**def** **saveimg**(**img**):

    system('cls')

    choice**=input**("Do you want to save the image: y/n\n").lower()

**if** choice**==**'y':

        sloc**=input**("Enter File location to save:- ")

        name**=input**('Enter Name with which to save:- \n\*NOTE:- Entering same name as existing image will replace the original image\n')

        img.save(sloc**+**'/'**+** name**+**".png")

**def** **main\_menu**():

    system('cls')

    choice**=input**(("Welcome to the Image Editor.\n\n\*\*\*\*Main Menu\*\*\*\*\nEnter the choice from the following(1-7):\n1. Open an Image.\n2. Copying an Image.\n3. Basic Menu.\n4. Creating Tumbnail.\n5. Create New Custom Image Menu.\n6. Filters Menu.\n7. Image Processing Menu.\n8. Get all the Colours Present in the Image.\n"))

    system('cls')

**if** choice**==**'1':

**try**:

            Image.open(**input**("Enter the location of an image which you wants to open: ")).show()

**except** **IOError**:

**input**("Give Correct Path.")

            main\_menu()

        main\_menu()

**elif** choice**==**'2':

        copyimg()

**elif** choice**==**'3':

        basic\_menu()

**elif** choice**==**'4':

        thumbnailimg()

**elif** choice**==**'5':

        createnew\_menu()

**elif** choice**==**'6':

        filter\_menu()

**elif** choice**==**'7':

        process\_menu()

**elif** choice**==**'8':

**try**:

**input**(**print**(Image.Image.getcolors(Image.open(**input**("Enter the location of an image which you wants to open: "))).convert("L")))

**except** **IOError**:

**input**("Give Correct Path.")

            main\_menu()

        main\_menu()

**else**:

**input**("Please Enter Valid Choice.")

        main\_menu()

**def** **copyimg**():

    system('cls')

**try**:

        copopen**=**Image.open(**input**("Enter image location to copy: "))

**except** **IOError**:

**input**("Give Correct Path.")

        copyimg()

    new\_copied**=**copopen.copy()

    new\_copied.show()

    saveimg(new\_copied)

    main\_menu()

**def** **basic\_menu**():

    system('cls')

    choice1**=input**("\*\*\*\*Basic Menu\*\*\*\*\nEnter the choice from the following(1-5):\n1. Rotate an image by a specific angle.\n2. Resizing the image.\n3. Cropping an image.\n4. Flip Image\n5. Bact to Main Menu.\n")

    system('cls')

**if** choice1**==**'1':

        rotateimg()

**elif** choice1**==**'2':

        resizeimg()

**elif** choice1**==**'3':

        cropimg()

**elif** choice1**==**'4':

        flipimg()

**elif** choice1**==**'5':

        main\_menu()

**else**:

**input**("Please Enter Valid Choice.")

        basic\_menu()

**def** **thumbnailimg**():

    system('cls')

**try**:

        thumbopen**=**Image.open(**input**("Enter image location of which thumbnail is required: "))

**except** **IOError**:

**input**("Give Correct Path.")

        thumbnailimg()

    thumbopen.thumbnail((**int**(**input**("Enter Width of Thumbnail")),**int**(**input**("Enter Height of Thumbnail"))))

    thumbopen.show()

    saveimg(thumbopen)

    main\_menu()

**def** **createnew\_menu**():

    system('cls')

    choice1**=input**("\*\*\*\*Create New Custom Image Menu\*\*\*\*\nEnter the choice from the following(1-5):\n1. Image Containing Noise.\n2. Linear Gradient.\n3. Radial Gradient.\n4. Image of a Colour.\n5. Bact to Main Menu.\n")

    system('cls')

**if** choice1**==**'1':

        effectnoise()

**elif** choice1**==**'2':

        linear\_gradient()

**elif** choice1**==**'3':

        radial\_gradient()

**elif** choice1**==**'4':

        newimg()

**elif** choice1**==**'5':

        main\_menu()

**else**:

**input**("Please Enter Valid Choice.")

        createnew\_menu()

**def** **filter\_menu**():

    system('cls')

    choice1**=input**("\*\*\*Filter Menu\*\*\*\*\nEnter the choice from the following(1-10):\n1.  Spread blur\n2.  Enhance Colour\n3.  Enhance Contrast\n4.  Enhance Brightness\n5.  Enhance Sharpness\n6.  Gaussian Blur\n7.  Contour\n8.  Detailed\n9.  Enhance Edges\n10. Smoothen\n11. Back to Main Menu.\n")

**if** choice1**==**'11':

        main\_menu()

    system('cls')

**try**:

        fil1**=**Image.open(**input**("Enter the location of an image on which you wants to apply the selected filter: "))

**except** **IOError**:

**input**("Give Correct Path.")

        system('cls')

        filter\_menu()

**if** choice1**==**'1':

        value**=int**(**input**("Enter value for spreading the pixels: "))

        filtered**=**fil1.effect\_spread(value)

        filtered.show()

**elif** choice1**==**'2':

        value**=float**(**input**("Enter value Enhancing Colour (Default is 1): "))

        filtered**=**ImageEnhance.Color(fil1).enhance(value)

        filtered.show()

**elif** choice1**==**'3':

        value**=float**(**input**("Enter value Enhancing Contrast (Default is 1): "))

        filtered**=**ImageEnhance.Contrast(fil1).enhance(value)

        filtered.show()

**elif** choice1**==**'4':

        value**=float**(**input**("Enter value Enhancing Brightness (Default is 1):- "))

        filtered**=**ImageEnhance.Brightness(fil1).enhance(value)

        filtered.show()

**elif** choice1**==**'5':

        value**=float**(**input**("Enter value Enhancing Sharpness (Default is 1):- "))

        filtered**=**ImageEnhance.Sharpness(fil1).enhance(value)

        filtered.show()

**elif** choice1**==**'6':

        value**=float**(**input**("Enter radius for Gaussian blur :- "))

        filtered**=**fil1.filter(ImageFilter.GaussianBlur(value))

        filtered.show()

**elif** choice1**==**'7':

        filtered**=**fil1.filter(ImageFilter.CONTOUR)

        filtered.show()

**elif** choice1**==**'8':

        filtered**=**fil1.filter(ImageFilter.DETAIL)

        filtered.show()

**elif** choice1**==**'9':

        filtered**=**fil1.filter(ImageFilter.EDGE\_ENHANCE)

        filtered.show()

**elif** choice1**==**'10':

        filtered**=**fil1.filter(ImageFilter.SMOOTH)

        filtered.show()

**else**:

**input**("Please Enter Valid Choice.")

        filter\_menu()

    saveimg(filtered)

    filter\_menu()

**def** **process\_menu**():

    system('cls')

    choice1**=input**("\*\*\*Image Processing Menu\*\*\*\*\nEnter the choice from the following(1-4):\n1. Blending Image.\n2. Alphing Image.\n3. Composite Image.\n4. Back to Main Menu.\n")

    system('cls')

**if** choice1**==**'1':

        blendimg()

**elif** choice1**==**'2':

        alphimg()

**elif** choice1**==**'3':

        compimg()

**elif** choice1**==**'4':

        main\_menu()

**else**:

**input**("Please Enter Valid Choice.")

        process\_menu()

**def** **rotateimg**():

    system('cls')

**try**:

        rotopen**=**Image.open(**input**("Rotate image.\nEnter image location to rotate: "))

**except** **IOError**:

**input**("Give Correct Path.")

        rotateimg()

    rotate**=**rotopen.rotate(**float**(**input**("Enter the Rotating Angle(in degrees): ")))

    rotate.show()

    saveimg(rotate)

    basic\_menu()

**def** **resizeimg**():

    system('cls')

    resizeloc**=input**("Resize Image.\nEnter Image Loaction for Resizing Image: ")

    rx,ry**=input**("Enter starting x and y coordinate as x,y.\n\*NOTE:- Resize will resize the image if new size is smaller than original size. ").split(",")

    resizeopen**=**Image.open(resizeloc)

    resized**=**resizeopen.resize((**int**(rx),**int**(ry)))

    resized.show()

    saveimg(resized)

    basic\_menu()

**def** **cropimg**():

    system('cls')

**try**:

        croploc**=input**("Crop Image.\nEnter Image Loaction for Croping Image: ")

**except** **IOError**:

**input**("Give Correct Path.")

        cropimg()

    x1,y1**=input**("Enter Starting x and y Co-ordinate as x,y: ").split(",")

    x2,y2**=input**("Enter Ending x and y Co-ordinate as x,y: ").split(",")

    cropopen**=**Image.open(croploc)

    cropped**=**cropopen.crop((**int**(x1),**int**(y1),**int**(x2),**int**(y2)))

    cropped.show()

    saveimg(cropped)

    basic\_menu

**def** **flipimg**():

    system('cls')

**print**("Flipping of Image.")

**try**:

        oimage1**=**Image.open(**input**("Enter Image Location for Fliping Image: "))

**except** **IOError**:

**input**("Give Correct Path.")

        flipimg()

    flip**=input**("Enter Image orientation \n1. Flip Left to Right\n2. Flip Top to Bottom\n3. Transpose\n4. Transverse\n")

**if** flip**==**"1":

        dis**=**oimage1.transpose(PIL.Image.FLIP\_LEFT\_RIGHT)

**elif** flip**==**"2":

        dis**=**oimage1.transpose(PIL.Image.FLIP\_TOP\_BOTTOM)

**elif** flip**==**"3":

        dis**=**oimage1.transpose(PIL.Image.TRANSPOSE)

**elif** flip**==**"4":

        dis**=**oimage1.transpose(PIL.Image.TRANSVERSE)

    dis.show()

    saveimg(dis)

    basic\_menu()

**def** **effectnoise**():

**print**("Make an Image Containing Noise.")

    noiseinp**=input**("1. Size is 256x256\n2. Size is 720x720\n3. Size is 1080x1080\n4. Size is 2000x2000\n5. Custom\n")

**if** noiseinp**==**"1":

        size**=**(256,256)

        sigma**=**500

**elif** noiseinp**==**"2":

        size**=**(720,720)

        sigma**=**500

**elif** noiseinp**==**"3":

        size**=**(1080,1080)

        sigma**=**500

**elif** noiseinp**==**"4":

        size**=**(2000,2000)

        sigma**=**500

**elif** noiseinp**==**"Custom":

        width**=int**(**input**("Enter Width: "))

        height**=int**(**input**("Enter Height: "))

        size**=**(width,height)

        sigma**=int**(**input**("Enter Sigma for noise: "))

    noiseeffect**=**Image.effect\_noise(size, sigma)

    noiseeffect.show()

    saveimg(noiseeffect)

    createnew\_menu()

**def** **linear\_gradient**():

    system('cls')

    lig**=**Image.linear\_gradient("L")

    lig.show()

    saveimg(lig)

    createnew\_menu()

**def** **radial\_gradient**():

    system('cls')

    rad**=**Image.radial\_gradient("L")

    rad.show()

    saveimg(rad)

    createnew\_menu()

**def** **newimg**():

    system('cls')

    mode1**=input**("Enter Mode: RGB, RGBA or L: ")

    size1**=int**(**input**("Enter Width: "))

    size2**=int**(**input**("Enter Height: "))

    colour1**=input**("Enter colour:\nBlack\nWhite\nRed\nMaroon\nMagenta\nLime\nGreen\nTeal\nBlue\nCyan\nNavy\nPurple\nYellow\nOlive\nSilver\nGrey\n")

**if** colour1**==**"Black" **and** mode1**!=**'L':

        makenew**=**Image.new(mode1,(size1,size2),(0,0,0))

**elif** colour1**==**"White" **and** mode1**!=**'L':

        makenew**=**Image.new(mode1,(size1,size2),(255,255,255))

**elif** colour1**==**"Red" **and** mode1**!=**'L':

        makenew**=**Image.new(mode1,(size1,size2),(255,0,0))

**elif** colour1**==**"Maroon" **and** mode1**!=**'L':

        makenew**=**Image.new(mode1,(size1,size2),(128,0,0))

**elif** colour1**==**"Magenta" **and** mode1**!=**'L':

        makenew**=**Image.new(mode1,(size1,size2),(255,0,255))

**elif** colour1**==**"Lime" **and** mode1**!=**'L':

        makenew**=**Image.new(mode1,(size1,size2),(0,255,0))

**elif** colour1**==**"Green" **and** mode1**!=**'L':

        makenew**=**Image.new(mode1,(size1,size2),(0,128,0))

**elif** colour1**==**"Teal" **and** mode1**!=**'L':

        makenew**=**Image.new(mode1,(size1,size2),(0,128,128))

**elif** colour1**==**"Blue" **and** mode1**!=**'L':

        makenew**=**Image.new(mode1,(size1,size2),(0,0,255))

**elif** colour1**==**"Cyan" **and** mode1**!=**'L':

        makenew**=**Image.new(mode1,(size1,size2),(0,255,255))

**elif** colour1**==**"Navy" **and** mode1**!=**'L':

        makenew**=**Image.new(mode1,(size1,size2),(0,0,128))

**elif** colour1**==**"Purple" **and** mode1**!=**'L':

        makenew**=**Image.new(mode1,(size1,size2),(128,0,128))

**elif** colour1**==**"Yellow" **and** mode1**!=**'L':

        makenew**=**Image.new(mode1,(size1,size2),(255,255,0))

**elif** colour1**==**"Olive" **and** mode1**!=**'L':

        makenew**=**Image.new(mode1,(size1,size2),(128,128,0))

**elif** colour1**==**"Silver" **and** mode1**!=**'L':

        makenew**=**Image.new(mode1,(size1,size2),(192,192,192))

**elif** colour1**==**"Green" **and** mode1**!=**'L':

        makenew**=**Image.new(mode1,(size1,size2),(128,128,128))

**elif** colour1**==**"Black" **and** mode1**==**'L':

        makenew**=**Image.new(mode1,(size1,size2),0)

**elif** colour1**==**"White" **and** mode1**==**'L':

        makenew**=**Image.new(mode1,(size1,size2),255)

**else**:

**print**("Mode and colour not compatible ")

    makenew.show()

    saveimg(makenew)

    createnew\_menu()

**def** **blendimg**():

    system('cls')

**print**("Blend Image.")

**try**:

        image1**=**Image.open(**input**("Enter Background Image Location: ")).convert("RGBA")

        image2**=**Image.open(**input**("Enter Foreground Image Location: ")).convert("RGBA")

**except** **IOError**:

**input**("Give Correct Path.")

        system('cls')

        blendimg()

    choice**=input**("Choose Transparency:-\n1. More background\n2. 50-50\n3. More foreground\nPress any key for custom alpha.")

**if** choice**==**'1':

        alpha**=**0.3

**elif** choice**==**'2':

        alpha**=**0.5

**elif** choice**==**'3':

        alpha**=**0.7

**else**:

        alpha**=float**(**input**("Enter alpha(Background fraction(from 0-1)): "))

    image1.thumbnail(image2.size)

    img1\_edit**=**image1.resize(image2.size)

    blend**=**Image.blend(img1\_edit,image2,alpha)

    blend.show()

    saveimg(blend)

    process\_menu()

**def** **alphimg**():

    system('cls')

**print**("Alpha Image.")

**try**:

**print**("\*NOTE-If foreground is not RGBA and has tranceparency, only Foreground is shown.")

        aimage1**=**Image.open(**input**("Enter Background Image Location: ")).convert("RGBA")

        aimage2**=**Image.open(**input**("Enter Foreground Image Location: ")).convert("RGBA")

**except** **IOError**:

**input**("Give Correct Path.")

        system('cls')

        alphimg()

    aimage1.thumbnail(aimage2.size)

    aimg1\_edit**=**aimage1.resize(aimage2.size)

    amix**=**Image.alpha\_composite(aimg1\_edit,aimage2)

    amix.show()

    saveimg(amix)

    process\_menu()

**def** **compimg**():

    system('cls')

**print**("Composite Image.")

**try**:

        cimage1**=**Image.open(**input**("Enter Background Image Location: ")).convert("RGBA")

        cimage2**=**Image.open(**input**("Enter Foreground Image Location: ")).convert("RGBA")

**print**("\*NOTE-Third image is cut from foreground to show background, so Third image should be of transperent type.")

        cimage3**=**Image.open(**input**("Enter Third Image Location: ")).convert("RGBA")

**except** **IOError**:

**input**("Give Correct Path.")

        system('cls')

        compimg()

    cimage1.thumbnail(cimage2.size)

    cimg1\_edit**=**cimage1.resize(cimage2.size)

    cimage3.thumbnail(cimage2.size)

    cimg3\_edit**=**cimage3.resize(cimage2.size)

    compos**=**Image.composite(cimg1\_edit,cimage2,cimg3\_edit)

    compos.show()

    saveimg(compos)

    process\_menu()

main\_menu()