In [1]:

```
import cv2 #for image processing
import numpy as np #to store image
import easygui #to open the filebox
import imageio #to read image stored at particular path
import sys
import os
import tkinter as tk
from tkinter import filedialog
from tkinter import *
from PIL import ImageTk, Image
import scipy
from scipy import stats
from collections import defaultdict
```

In [2]:

```
win=tk.Tk()
win.geometry('400x400')
win.title('Cartoonify Your Image !')
win.configure(background='white')
label=Label(win,background='#CDCDCD', font=('calibri',20,'bold'))
def upload():
    ImagePath=easygui.fileopenbox()
    cartoonify(ImagePath)
def close():
    win.destroy()
def update_c(C,hist):
    while True:
        groups=defaultdict(list)
        for i in range(len(hist)):
            if(hist[i] == 0):
                continue
            d=np.abs(C-i)
            index=np.argmin(d)
            groups[index].append(i)

        new_C=np.array(C)
        for i,indice in groups.items():
            if(np.sum(hist[indice])==0):
                continue
            new_C[i]=int(np.sum(indice*hist[indice])/np.sum(hist[indice]))
            if(np.sum(new_C-C)==0):
                break
            C=new_C
    return C,groups
```

In [*]:

```
# Calculates K Means clustering
def K_histogram(hist):
    alpha=0.001
    N=80
    C=np.array([128])
    while True:
        C,groups=update_c(C,hist)
        new_C=set()
        for i,indice in groups.items():
            if(len(indice)<N):
                new_C.add(C[i])
                continue
            z, pval=stats.normaltest(hist[indice])
            if(pval<alpha):
                left=0 if i==0 else C[i-1]
                right=len(hist)-1 if i ==len(C)-1 else C[i+1]
                delta=right-left
                if(delta >=3):
                    c1=(C[i]+left)/2
                    c2=(C[i]+right)/2
                    new_C.add(c1)
                    new_C.add(c2)
                else:
                    new_C.add(C[i])
            else:
                new_C.add(C[i])
        if(len(new_C)==len(C)):
            break
        else:
            C=np.array(sorted(new_C))
    return C
def cartoonify(ImagePath):
    # read the image
    img = cv2.imread(ImagePath)
    kernel=np.ones((2,2), np.uint8)
    output=np.array(img)
    x,y,c=output.shape
    for i in range(c):
        output[:,:,i]=cv2.bilateralFilter(output[:,:,i],5,150,150)
    edge=cv2.Canny(output, 100, 200)
    output=cv2.cvtColor(output,cv2.COLOR_RGB2HSV)

    hists=[]

    hist,_=np.histogram(output[:,:,0],bins =np.arange(180+1))
    hists.append(hist)
    hist,_=np.histogram(output[:,:,1],bins =np.arange(256+1))
    hists.append(hist)
    hist,_=np.histogram(output[:,:,2],bins =np.arange(256+1))
    hists.append(hist)
    C=[]
    for h in hists:
        C.append(K_histogram(h))
    #print("centroids: {0}".format(C))
    output=output.reshape((-1,c))
    for i in range(c):
        channel=output[:,i]
        index=np.argmin(np.abs(channel[:, np.newaxis] - C[i]), axis=1)
        output[:,i]=C[i][index]
```

```
    output=output.reshape((x,y,c))
    output=cv2.cvtColor(output, cv2.COLOR_HSV2RGB)
    contours,_=cv2.findContours(edge,cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_NONE)
    cv2.drawContours(output,contours,-1,0,thickness=1)
    #cartoon = cv2.bitwise_and(output, output, mask=contours)
    for i in range(3):
        output[:,:,i]=cv2.erode(output[:,:,i], kernel, iterations=1)

    save1=Button(win,text="Save cartoon image",command=lambda: save(output, ImagePath),padx
    save1.configure(background='#364156', foreground='white',font=('calibri',10,'bold'))
    save1.pack(side=TOP,pady=50)

def refresh():
    win.destroy()
    os.system('python "C:/Users/Hashi Ktk/Desktop/Cartoonifier/Cartoonify_the_reality(By K-

def nextnonexistent(f):
    fnew = f
    root, ext = os.path.splitext(f)
    i = 0
    while os.path.exists(fnew):
        i += 1
        fnew = '%s_%i%s' % (root, i, ext)
    return fnew

def save(output, ImagePath):
    #saving an image using imwrite()
    newName="cartoonified_Image"
    path1 = os.path.dirname(ImagePath)
    extension=os.path.splitext(ImagePath)[1]
    path = os.path.join(path1, newName+extension)
    newName = nextnonexistent(path)
    path = nextnonexistent(path)
    cv2.imwrite(path, cv2.cvtColor(output, cv2.COLOR_RGB2BGR))
    I= "Image saved by name " + newName +" at "+ path
    tk.messagebox.showinfo(title=None, message=I)
    refresh()

close=Button(win,text="Exit", compound="left", command=close,padx=10,pady=5)
close.configure(background='#364156', foreground='white',font=('calibri',10,'bold'))
close.place(x=300 ,y=320)
upload=Button(win,text="Cartoonify an Image",command=upload,padx=10,pady=5)
upload.configure(background='#364156', foreground='white',font=('calibri',10,'bold'))
upload.pack(side=TOP,pady=50)
win.mainloop()
```

In [ ]:

```
```