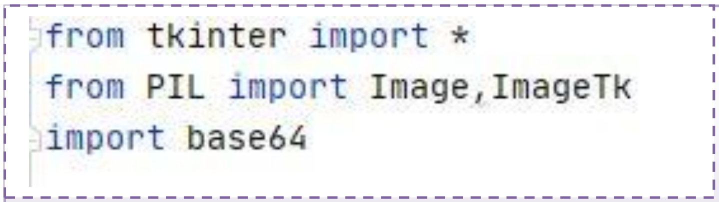
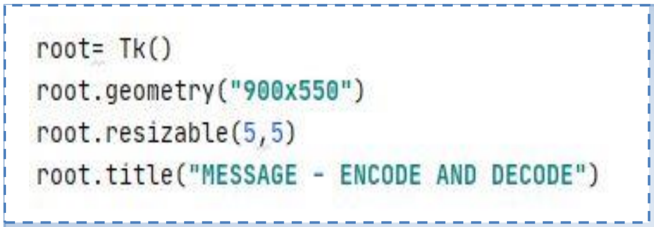
ENCODING/DECODING OF MESSAGES



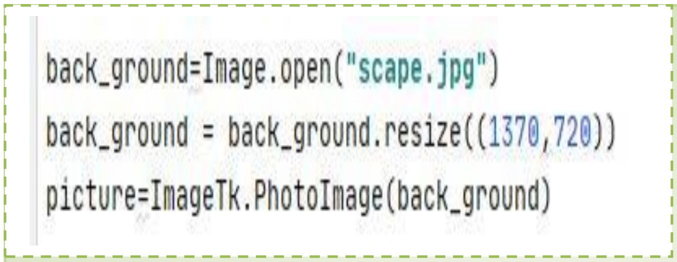
```
from tkinter import *
from PIL import Image,ImageTk
import base64
```

- The line **import*** imports all modules from tkinter
- The **PIL module** helps us to include images .
- The **module base 64** helps us in doing the functionality of encoding and decoding



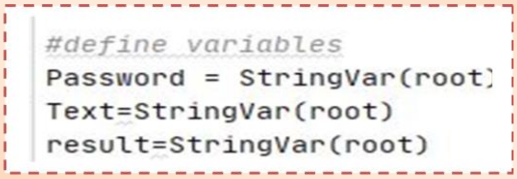
```
root= Tk()
root.geometry("900x550")
root.resizable(5,5)
root.title("MESSAGE - ENCODE AND DECODE")
```

- **Tk()** initializes the tkinter window
- **Geometry()** set width and height of the tkinter window
- **Resizable()** sets the fixed and unfixed size of the window
- **Title()** sets the title of the window



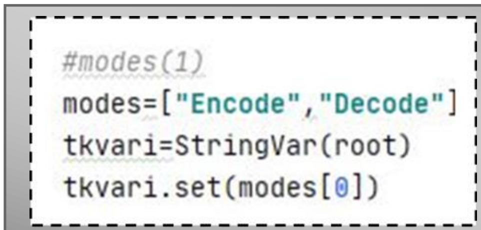
```
back_ground=Image.open("scape.jpg")
back_ground = back_ground.resize((1370,720))
picture=ImageTk.PhotoImage(back_ground)
```

- **Image** module provides a class with same name which is used as a **PIL** image. **Image.open()** loads the image .
- **Background.resize()** resizes the image size
- **ImageTk** module contains support to create and modify **PhotoImage** objects from PIL Images
- **PhotoImage()** function returns the Image object.



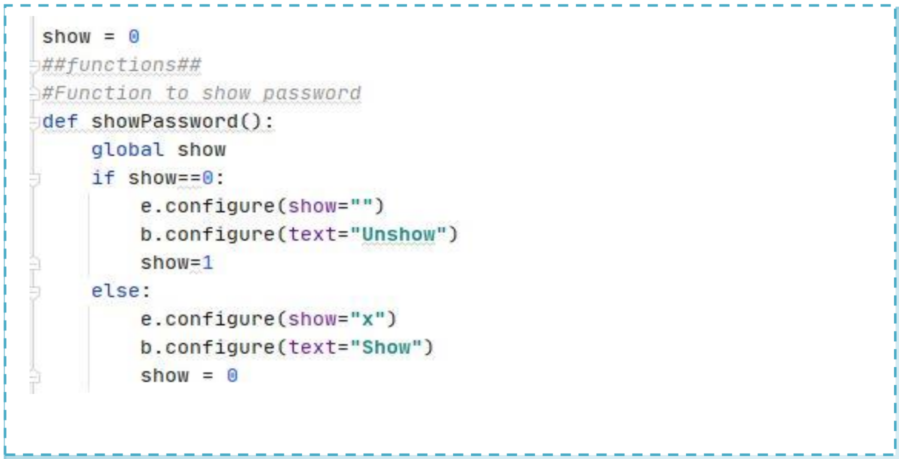
```
#define variables
Password = StringVar(root)
Text=StringVar(root)
result=StringVar(root)
```

- **Password** variable stores the key used in encode and decode.
- **Text** variable stores the message to encode and encode.
- **Result** variable stores the result.



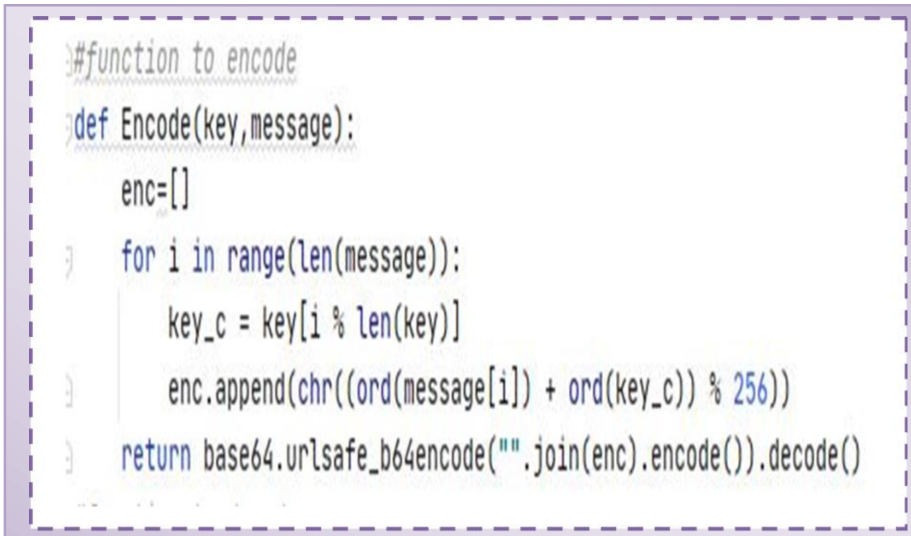
```
#modes(1)
modes=["Encode","Decode"]
tkvari=StringVar(root)
tkvari.set(modes[0])
```

- **Modes** list stores the modes encode and decode.
- **Tkvari** variable is used to select the mode encoding or decoding.
- **Set()** function sets the first element of modes list to the variable **Tkvari.**



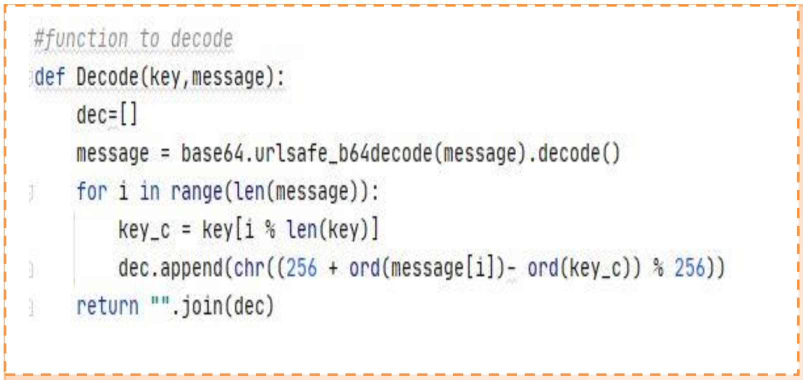
```
show = 0
##functions##
#Function to show password
def showPassword():
    global show
    if show==0:
        e.configure(show="")
        b.configure(text="Unshow")
        show=1
    else:
        e.configure(show="x")
        b.configure(text="Show")
        show = 0
```

- **ShowPassword()** function is used to unable and able the contents of key. It also used to enable and disable the show button.
- **Configure()** is used to access the contents of show button



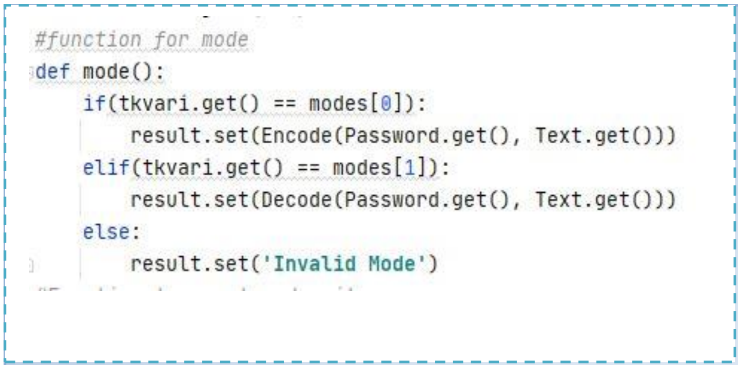
```
#function to encode
def Encode(key,message):
    enc=[]
    for i in range(len(message)):
        key_c = key[i % len(key)]
        enc.append(chr((ord(message[i]) + ord(key_c)) % 256))
    return base64.urlsafe_b64encode("".join(enc).encode()).decode()
```

- **Enc** is an empty list
- **Ord()** function takes Unicode character and returns Unicode value
- **Chr()** function takes integer argument and returns the string
- **Key_c** variable contains the remainder of division between I and len(key)
- **Base64.urlsafe_b64encode()** function encodes the string
- **Join()** method joins each element of the enc list by string separator and returns the concatenated string
- **Encode()** method returns encoded string
- **Decode()** method decodes the string



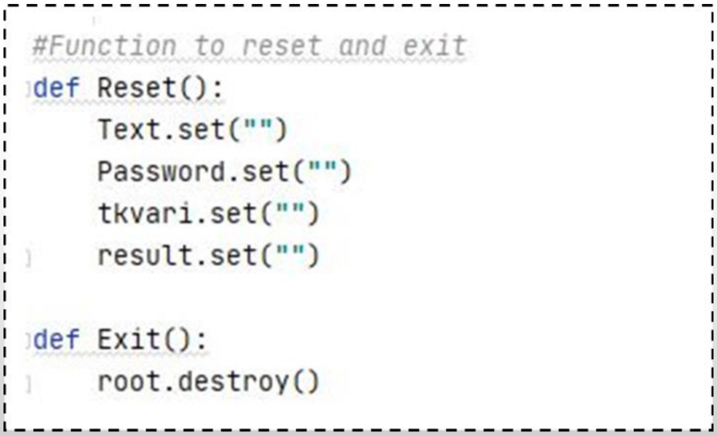
```
#function to decode
def Decode(key,message):
    dec=[]
    message = base64.urlsafe_b64decode(message).decode()
    for i in range(len(message)):
        key_c = key[i % len(key)]
        dec.append(chr((256 + ord(message[i])- ord(key_c)) % 256))
    return "".join(dec)
```

- **Dec** is an empty list
- decode the contents from input and writes the result in binary
- 256+ord(**message**[i]-ord(key_c))%256 gives reminder of addition of 256 with subtraction of ord(message[i]-ord(key_c)) and divides with 256 and passes that reminder to chr() function
- Chr() function convert integer value to string and store to dec
- Return "".join(dec) returns the result



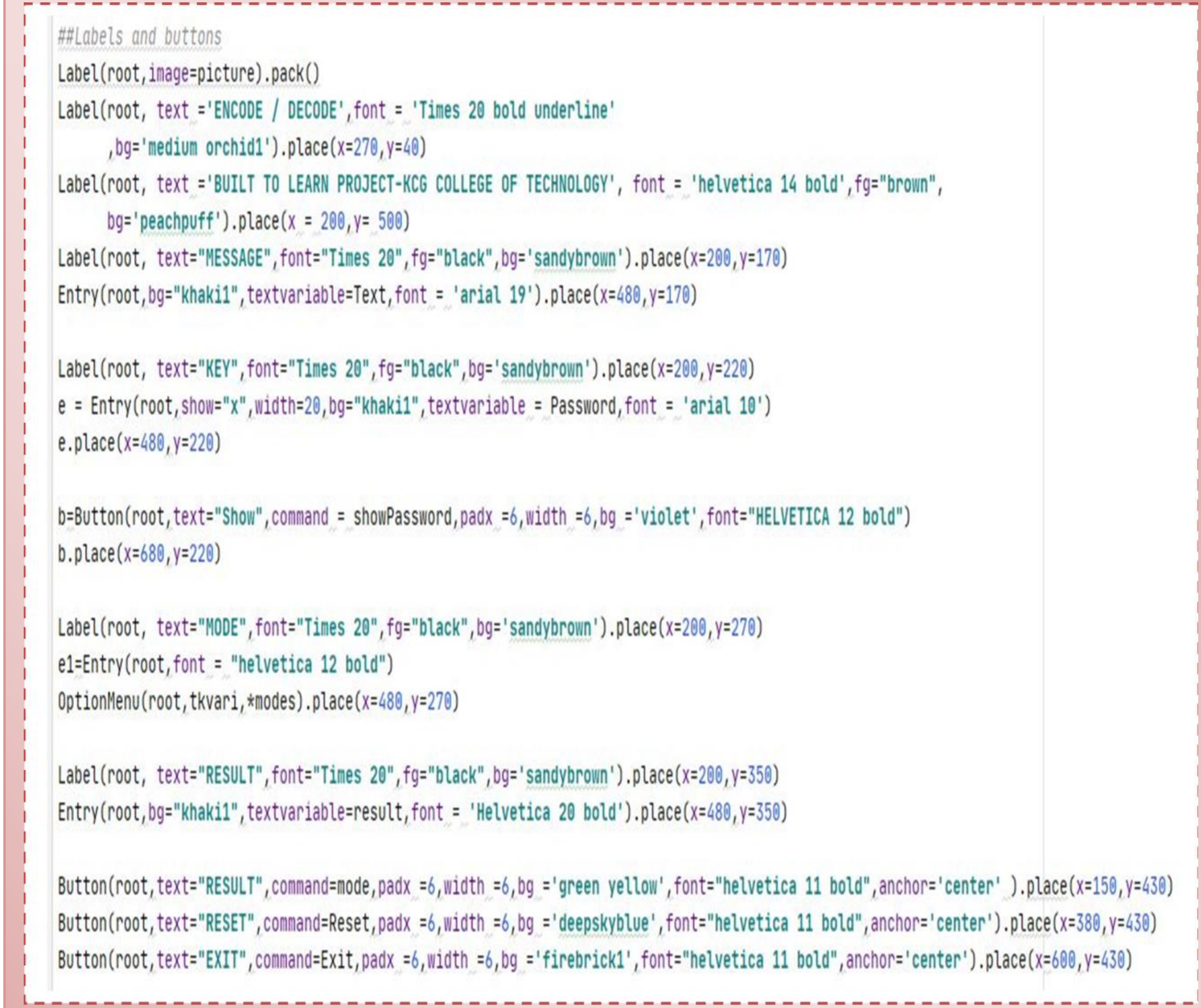
```
#function for mode
def mode():
    if(tkvari.get() == modes[0]):
        result.set(Encode(Password.get(), Text.get()))
    elif(tkvari.get() == modes[1]):
        result.set(Decode(Password.get(), Text.get()))
    else:
        result.set('Invalid Mode')
```

- If the mode set by the user is modes[0] , then **Encode()** function will be called.
- If the mode set by the user is modes[1] , then **Decode()** function will be called.
- Else print “INVALID MODE”
- **Password.get()** and **Text.get()** values are passed as an argument to Encode() and Decode() function.



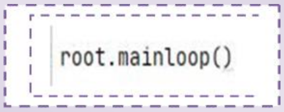
```
#Function to reset and exit
def Reset():
    Text.set("")
    Password.set("")
    tkvari.set("")
    result.set("")

def Exit():
    root.destroy()
```

- **Reset ()** function is used to set all the string variables to empty strings.
- **Exit()** function is used to exit the tkinter window
- **Root.destroy()** will quit the program by stopping the mainloop()



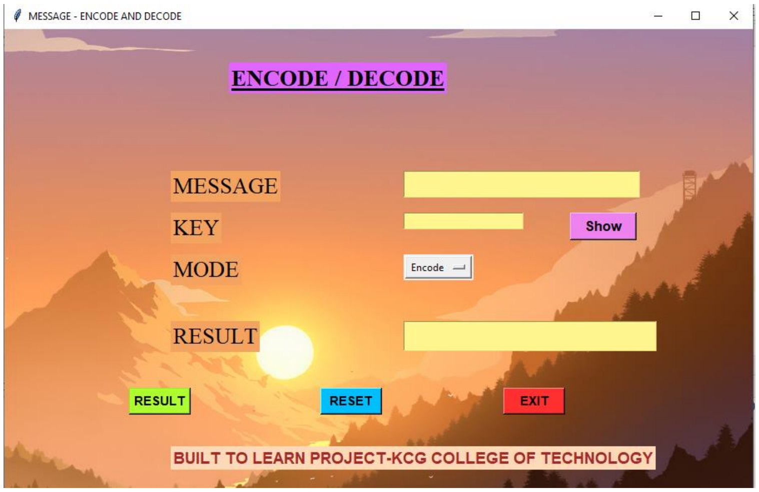
```
##Labels and buttons
Label(root,image=picture).pack()
Label(root, text ='ENCODE / DECODE',font = 'Times 20 bold underline'
      ,bg='medium orchid1').place(x=270,y=40)
Label(root, text ='BUILT TO LEARN PROJECT-KCG COLLEGE OF TECHNOLOGY', font = 'helvetica 14 bold',fg="brown",
      bg='peachpuff').place(x = 200,y= 500)
Label(root, text="MESSAGE",font="Times 20",fg="black",bg='sandybrown').place(x=200,y=170)
Entry(root,bg="khaki1",textvariable=Text,font = 'arial 19').place(x=480,y=170)

Label(root, text="KEY",font="Times 20",fg="black",bg='sandybrown').place(x=200,y=220)
e = Entry(root,show="x",width=20,bg="khaki1",textvariable = Password,font = 'arial 10')
e.place(x=480,y=220)

b=Button(root,text="Show",command = showPassword,padx =6,width =6,bg ='violet',font="HELVETICA 12 bold")
b.place(x=680,y=220)

Label(root, text="MODE",font="Times 20",fg="black",bg='sandybrown').place(x=200,y=270)
e1=Entry(root,font = "helvetica 12 bold")
OptionMenu(root,tkvari,*modes).place(x=480,y=270)

Label(root, text="RESULT",font="Times 20",fg="black",bg='sandybrown').place(x=200,y=350)
Entry(root,bg="khaki1",textvariable=result,font = 'Helvetica 20 bold').place(x=480,y=350)

Button(root,text="RESULT",command=mode,padx =6,width =6,bg ='green yellow',font="helvetica 11 bold",anchor='center' ).place(x=150,y=430)
Button(root,text="RESET",command=Reset,padx =6,width =6,bg ='deepskyblue',font="helvetica 11 bold",anchor='center').place(x=380,y=430)
Button(root,text="EXIT",command=Exit,padx =6,width =6,bg ='firebrick1',font="helvetica 11 bold",anchor='center').place(x=600,y=430)
```

- **Label()** widget used to display one or more than one line of text that users aren’t able to modify.
- **Entry()** widget used to create an input text field.
- **Button()** widget used to display button on our window.
- **Root** is the name which we refer to our window
- **Text** which we display on the label
- **Bg** sets the background colour
- **Command** is call when the button is clicked
- **Textvariable** is used to retrieve the current text to the entry widget.
- **Place()** function is used to place the object in a fixed position
- **Anchor** is used to mention the position
- **Width** is used to set the dimension along x-axis
- **Padx** puts some space between button widget and the right border of the root window.



```
root.mainloop()
```

- **Root.mainloop()** is a method that executes when we want to run our program.



MESSAGE - ENCODE AND DECODE

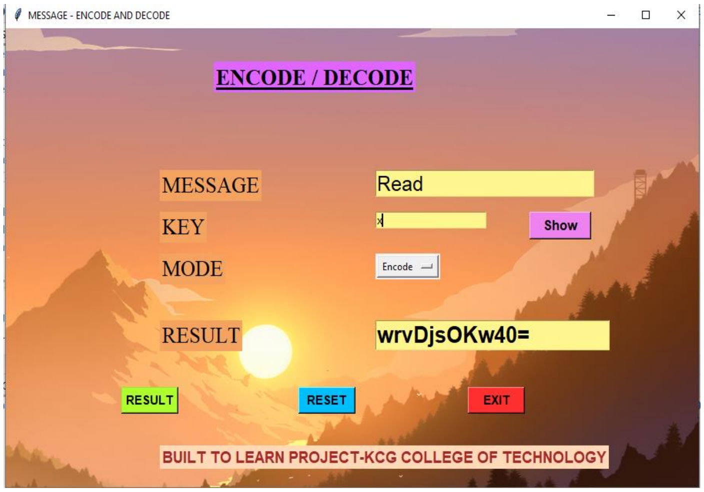
ENCODE / DECODE

MESSAGE

KEY

MODE Encode

RESULT



MESSAGE - ENCODE AND DECODE

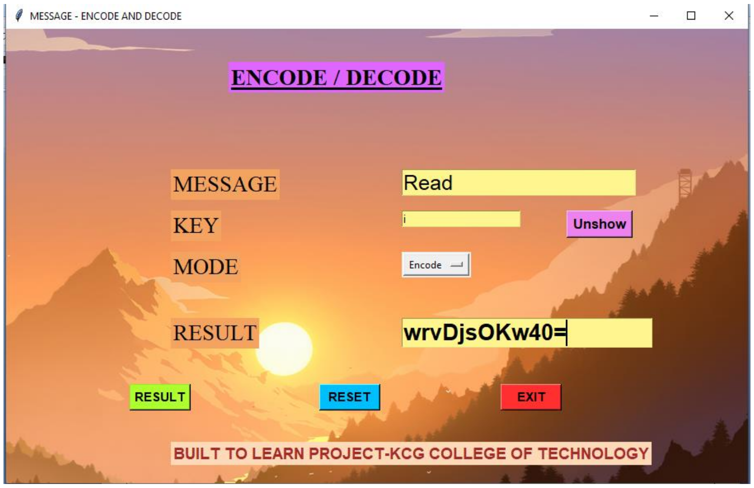
ENCODE / DECODE

MESSAGE Read

KEY x

MODE Encode

RESULT wrvDjsOKw40=



MESSAGE - ENCODE AND DECODE

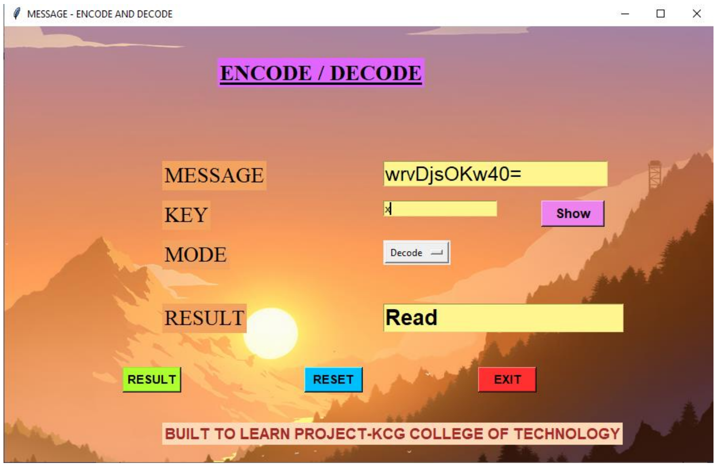
ENCODE / DECODE

MESSAGE Read

KEY i

MODE Encode

RESULT wrvDjsOKw40=



MESSAGE - ENCODE AND DECODE

ENCODE / DECODE

MESSAGE wrvDjsOKw40=

KEY x

MODE Decode

RESULT Read