```
from tkinter import *
from tkinter import ttk
from tkinter.ttk import *
from tkinter.messagebox import *
from tkinter.filedialog import *
import os, sys, sqlite3, hashlib

def hashKey(pswd):
    h = hashlib.sha256()
    h.update(pswd.encode())
    pk = h.hexdigest()
    k = ''
    for i in list(pk):
        h = hashlib.sha256()
        h.update(i.encode())
        k += h.hexdigest()

    return k

class ExOr:
    def __init__(self):
        self.data = None
        self.key = None
        self.rslt = None
        self.file = None

    def send(self, file, key):
        try:
            self.file = file
            f = open(self.file, 'rb')
            self.data = f.read()
            f.close()
            self.key = key

        except FileNotFoundError:
            f = open('exor.log', 'a')
            f.write('Error : file not found !\n')
            f.close()
            self.data = b''

    def crypt(self):
        self.rslt = []
        for i in range(len(self.data)):
            self.rslt.append(self.data[i] ^ ord(self.key[i%le
n(self.key)]))

        f = open(self.file, 'wb')
        for i in self.rslt:
            f.write(bytes([i]))
        f.close()

class Graphic:
    def __init__(self, file, project = None):
        self.title = file
        self.project = project
        self.data = None

    def ask(self):
        self.window()
        self.content()
        self.Geom()
        self.Generate()

        return self.data

    def window(self, is_file = True):
        self.master = Tk()
        self.master.overrideredirect(1)
        self.master.resizable(width=False, height=False)

        self.top = Frame(self.master)
        self.top.grid(row=1, column=0)

        self.menu = Frame(self.master)
        self.menu.grid(row=0, column=0)

        name = 'Close '
        if is_file:
            name += '"' + self.title + '"'
        Button(self.menu, text=name, command=self.Quitter).gr
id(row=0, column=0)

    def Quitter(self):
        self.data = None
        self.master.destroy()

    def Geom(self):
        self.master.update()

        sw = self.master.winfo_screenwidth()
        sh = self.master.winfo_screenheight()

        ww = self.master.winfo_reqwidth()
        wh = self.master.winfo_reqheight()

        x = '+' + str(int((sw/2) - (ww/2)))
        y = '+' + str(int((sh/2) - wh))

        self.master.geometry(x + y)

    def content(self, disabled = False):
        Label(self.top, text='ExOr Encrypton System', font=('
Consolas', 20)).grid(row=0, column=0)
        if self.project:
            Label(self.top, text=self.project.upper(), bg='bl
ue', fg='red', font=('Courier New', 26, 'bold')).grid(row=1,
column=0)
        Label(self.top, text='Please, Enter password :' if no
t disabled else 'Please, close window...').grid(row=2, column
=0)
        self.pwd = StringVar()
        pwd = ttk.Entry(self.top, textvariable=self.pwd, show
='$', width=20, font=('Calibri', 16), stat = 'disabled' if di
sabled else 'normal')
        pwd.grid(row=3, column=0)
        b = ttk.Button(self.top, text='Encrypt/Decrypt', comm
and=self.BtCommand, stat = 'disabled' if disabled else 'norma
l')
        b.grid(row=4, column=0)
        if not disabled:
            b.bind_all('<Return>', self.BtCommand)

    def Generate(self):
        self.top.mainloop()

    def BtCommand(self, evt=None):
        self.data = self.pwd.get()
        if self.data == '':
            self.data = None
        self.master.destroy()


class Finished(Graphic):
    def __init__(self):
        Graphic.__init__(self, '')
        self.window(False)
        self.content(True)
        self.Geom()
        showinfo('ExOr Encrypton System', 'File Encrypted / D
ecrypted\nPlease, use the same password to Encrypt/Decrypt it
 !')
        self.Generate()

class Main:
    def __init__(self, filename):
        self.filename = filename
        self.pwd = None
        self.key = None

    def lookProject(self):
        conn = sqlite3.connect(sys.argv[0].replace('main.exe'
, 'files.db'))
        cur = conn.cursor()

        cur.execute('CREATE TABLE IF NOT EXISTS list(id INTEG
ER PRIMARY KEY AUTOINCREMENT, file TEXT, project TEXT)')
        conn.commit()

        for row in cur.execute('SELECT * FROM list'):
            if row[1] == self.filename:
                return row

        return [None for i in range(3)]

    def askKey(self):
        pro = self.lookProject()
        g = Graphic(self.filename, pro[2])
        self.pwd = g.ask()

    def hashKey(self):
        self.key = hashKey(self.pwd)

    def crypt(self):
        e = ExOr()
        e.send(self.filename, self.key)
        e.crypt()

    def end(self):
        Finished()

if __name__ == '__main__':
    try:
        m = Main(sys.argv[1])
        m.askKey()
        m.hashKey()
        m.crypt()
        m.end()

    except IndexError:
        f = open('__init__.log', 'a')
        f.write('Error : no command supplied !\n')
        f.close()
```