ОЛИМПИАДА ШКОЛЬНИКОВ «ШАГ В БУДУЩЕЕ»

НАУЧНО-ОБРАЗОВАТЕЛЬНОЕ СОРЕВНОВАНИЕ «ШАГ В БУДУЩЕЕ, МОСКВА»

ШМ0292

Информатика и системы управления

ИУ8 – Информационная безопасность

Шифрование изображений

Автор: Чиженков Борис Михайлович
ГБОУ ЛИТ №1537, 11«Л»

Научный руководитель: Глинская Елена Вячеславовна
МГТУ им. Баумана
старший преподаватель

Москва - 2016

ОГЛАВЛЕНИЕ

Научно-исследовательская работа по теме:

ШИФРОВАНИЕ ИЗОБРАЖЕНИЙ

Постановка проблемы

Более 90% информации об окружающем мире человек получает с помощью зрения. Поэтому не удивительно, что на большинстве носителей информации хранятся различные графические файлы, являющиеся, возможно, семейными фотографиями, сохраненными картинками из интернета и прочим. Однако, это может быть и что-то более значимое, то, что ни при каких обстоятельствах не должно попасть к посторонним людям, например, схема военно-промышленного комплекса. Чаще всего утечка подобной информации может происходить при передаче файлов через интернет, а также всегда существует вероятность банально потерять флэшку с данными. В любом случае, даже одно изображение может нести очень много важной информации, и поэтому утечка таких файлов, даже в малых количествах, может быть катастрофична. Следовательно, безопасность хранения и передачи изображений особенно важна.

Цель

Исследовать способы хранения, представления и передачи графической информации. Изучить модели представления цветовой гаммы, форматы и расширения, используемые в личных и профессиональных целях, их особенностей, преимуществ и недостатков. Ознакомиться с криптографией и стеганографией, различными шифрами и их видами. На основе полученных знаний создать ПО для шифрования и дешифровки изображений, поддерживающее большинство графических форматов, цветовых моделей представления изображений, а также поддержку хорошего качества цветопередачи без каких-либо потерь. Создание соответствующих алгоритмов для достижения цели, при которых исключается возможность доступа посторонних лиц к исходному файлу, даже при их максимальной контрастности и прочим характеристикам. Реализация графического интерфейса для пользователей программы, который дает возможность ясного отображения возможных ошибок, прогресса выполнения программы и прочего. Планирование будущего ПО и возможностей его развития, улучшение степени безопасности в целом. Код итогового продукта приведен в разделе «Приложение».

Исследования

Постановка задач программы требовала достаточно глубокого изучения графики и её представления в целом. Наиболее часто используемыми являются такие форматы и расширения, как:

1. JPEG (.jpeg/.jpg)
2. PNG (.png)
3. TIFF (.tiff/.tif)
4. BMP (.bmp)

Но более того, большинство форматов из приведенного списка могут быть представлены в нескольких цветовых моделях, что тоже нельзя не учитывать.

Табл. 1 Цветовые модели

	RGB	**RGBA/RGBX**	**CMYK**
JPEG	+	-	+
PNG	+	+	-
TIFF	+	+	+
BMP	+	-	-

Формату JPEG свойственна цветовая субдискретизация, поэтому при шифровании таких изображений зашифрованная картинка будет находиться в ином формате, зависящем от используемой цветовой модели (RGB - PNG, CMYK - TIFF). Это конвертирование необходимо, потому что при шифровании/дешифровке не должно происходить абсолютно никаких сжатий и прочих произвольных изменений

цветов. Малейшее смещение по гамме вызывет полное искажение исходных данных при расшифровке, так как почти каждое значение каналов всех пикселей несет в себе неоходимую для дешифровки информацию, искажать которую никак нельзя. Однако, если исходник был в JPEG, то расшифрованная картинка тоже будет в этом формате. Использование других форматов необходимо только для промежуточной стадии. Для определения программой изначального формата используется один из каналов в правом нижнем пикселе любого зашифрованного изображения.

Алгоритм

Основной задачей по мере создания программы стало создание такого алгоритма шифрования, при использовании которого ключ для дешифровки хранится в самом изображении, а не в виде отдельного текстового или графического файла. Экспериментальным путем я установил, что для шифрования изображений в больших разрешениях не подходят алгоритмы, которыми не предусмотрена геометрическая деформация, или предусмотрена, но составлена по какой-либо математической модели, поэтому одним из свойств итогового алгоритма является случайная перестановка зашифрованных пикселей исходной картинки или компонентов их значений. Далее необходимо было изменить цвета таким образом, при котором они ни коем образом не содержали информацию об исходной картинке. Использование алгоритмов, которые включают в себя какие-либо математические операции со значениями каналов каждого пикселя исходного изображения не являются успешными, так как всегда найдется такой размер изображения, при котором человеческий глаз будет различать контуры исходного файла. Более того, интеграция случайных чисел в подобные преобразования также безуспешна, в связи с ограничением по глубине цвета и области допустимых значений каналов соответственно. Особенно подвержены этой проблеме картинки, которые представляют из себя контрастные, четкие геометрические фигуры или надписи. Два нижеприведенных изображения представляют из себя исходную картинку и зашифрованную.

Рис. 1 Исходник

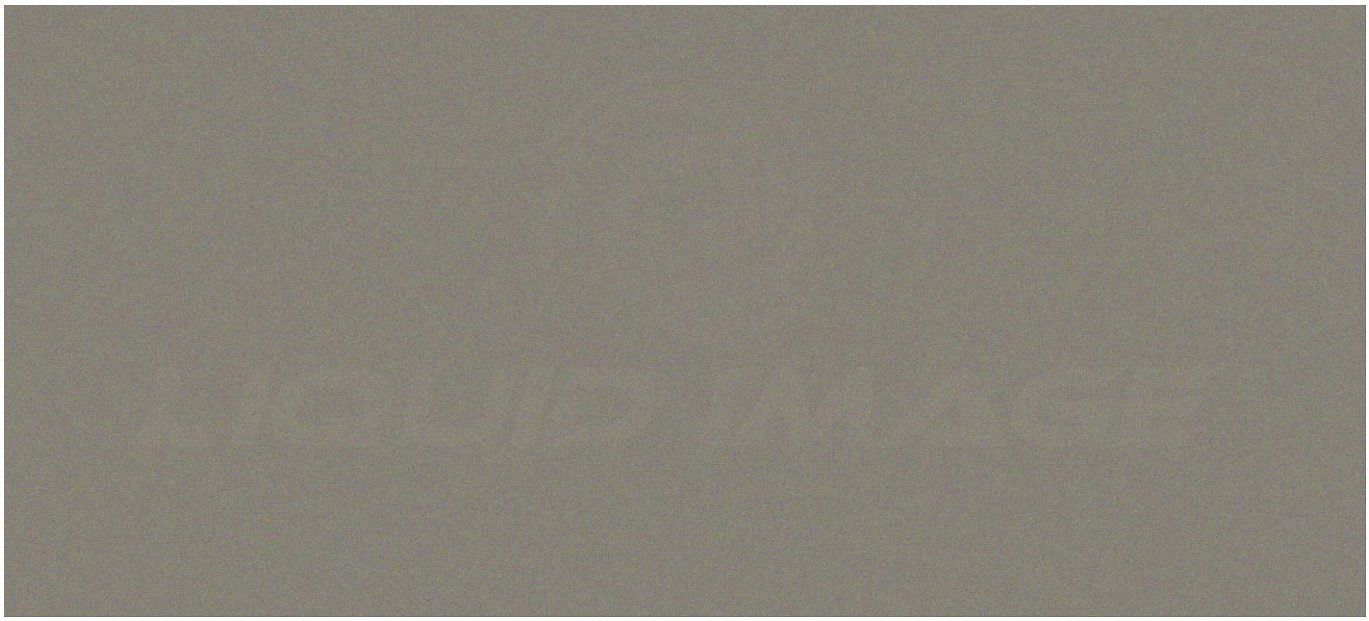

Рис. 2 Зашифрованный вид

Как видно, зашифрованной ее назвать нельзя, так как в глаза сразу бросается светлый контур букв и даже очерки логотипа. Решение этой проблемы стало возможным при использовании поразрядного разделения числовых компонентов каналов и работе с каждым разрядом отдельно. В результате получился алгоритм, который работает по следующему принципу:

1. Получение числовых значений каждого канала пикселя.

2. Разделение каждого значения по разрядам.
3. Добавление к каждому из полученных цифр случайного числа, оканчивающегося на 0 и находящемуся в пределах глубины цвета.
4. Формирование пикселей, каналы которых принимают полученные значения.
5. Создание графического блока, размер которого соответствует количеству новых пикселей.
6. Заполнение пустого изображения этими блоками случайным образом.

В конечном итоге, алгоритм представляет из себя смешение шифров подстановки и перестановки. Перестановочную составляющую шифра можно разделить на 2 части – составление новых графических ячеек при помощи шифра маршрутной перестановки и дальнейшее случайное распределение этих ячеек. Подстановочная составляющая предствляет из себя добавление к изъятым из исходных разрядов значений случайных чисел кратных десяти с учетом ограничений по глубине цвета.

При более подробном рассмотрении необходимо различать виды используемых шифров и их назначение. Перестановочный шифр представляет из себя метод симметричного шифрования, в котором элементы исходного содержимого меняют местами. В классической криптографии их можно разделить на два класса: одинарной и множественной перестановки, различие которых заключается в количестве перестановок соответственно. В моем случае используется простая перестановка, которую можно назвать разновидностью табличной маршрутной перестановки. Подстановочные шрифты - это методы шифрования, при которых элементы исходных данных заменяются другими в соответствии с некоторыми правилами. Элементами могут являться как отдельные символы, так и их группы. Мною используется добавление к существующим цифрам таких чисел, при суммировании с которыми младший разряд получившегося числа является числом изначальным, что и позволяет совершать расшифровку.

Практическое применение

С точки зрения практического применения моя программа может быть использована для безопасного хранения графических файлов на физических и виртуальных носителях различной степени доступности. Также ее можно использовать для достижения безопасной передачи изображений по открытым каналам, например, незащищенным беспроводным wi-fi сетям. Более того, гораздо более надежно перемещаться по городу с флешкой, на которой данные зашифрованны, нежели легко читаемы посторонними лицами в случае утери носителя.

Графический интерфейс пользователя

Для удобства пользования программой был создан GUI на основе библиотеки tkinter. Окно программы выглядит следующем образом:

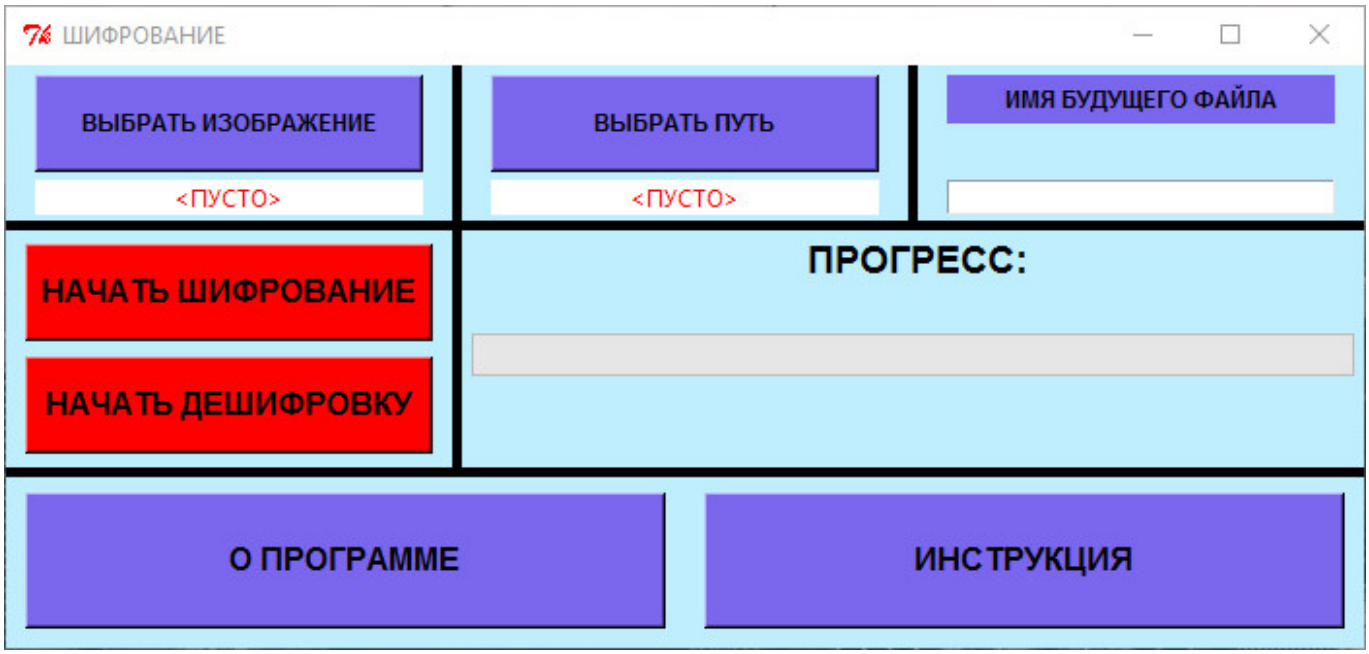

Рис. 3 Меню

При нажатии на кнопку “Выбрать изображение” появляется диалоговое окно выбора файла. Пользователь выбирает изображение, которое необходимо подвергнуть шифрованию/дешифровке. Если выбран файл, который не является изображением, или не содержит графического содержимого, или имеет неподдерживаемый программой формат, то появится сообщение об ошибке. Ниже отображается название выбранного файла. Далее при нажатии кнопки “Выбрать путь” появляется диалоговое окно выбора папки. Здесь пользователь должен выбрать место, куда будет сохранено полученное после шифрования/дешифровки изображение. Ниже отображается путь к выбранному каталогу. Если путь слишком длинный и не вмещается в отведенное для его отображения место, отображается только конец пути к этой папке. Правее расположено окно ввода текста, где пользователю предлагается ввести имя для будущего файла. При использовании спецсимволов или отсутствии имени программа оповещает об ошибке. Все 3 необходимых компонента для начала шифрования/дешифровки, о которых

говорилось ранее, можно менять сколько угодно раз в любой последовательности. При их корректном вводе кнопки “Начать шифрование” и “Начать дешифровку” загораются зеленым. При нажатии любой из них последует анимация прогресса действия, отображающая реальное соотношение совершенной программой работы, а также все кнопки, кроме как “О программе” и “Инструкция” станут неактивными и будут подсвечены желтым цветом. Под строкой прогресса также появится кнопка “Отмена”, при нажатии которой шифрование/дешифровка файла корректно прервется.

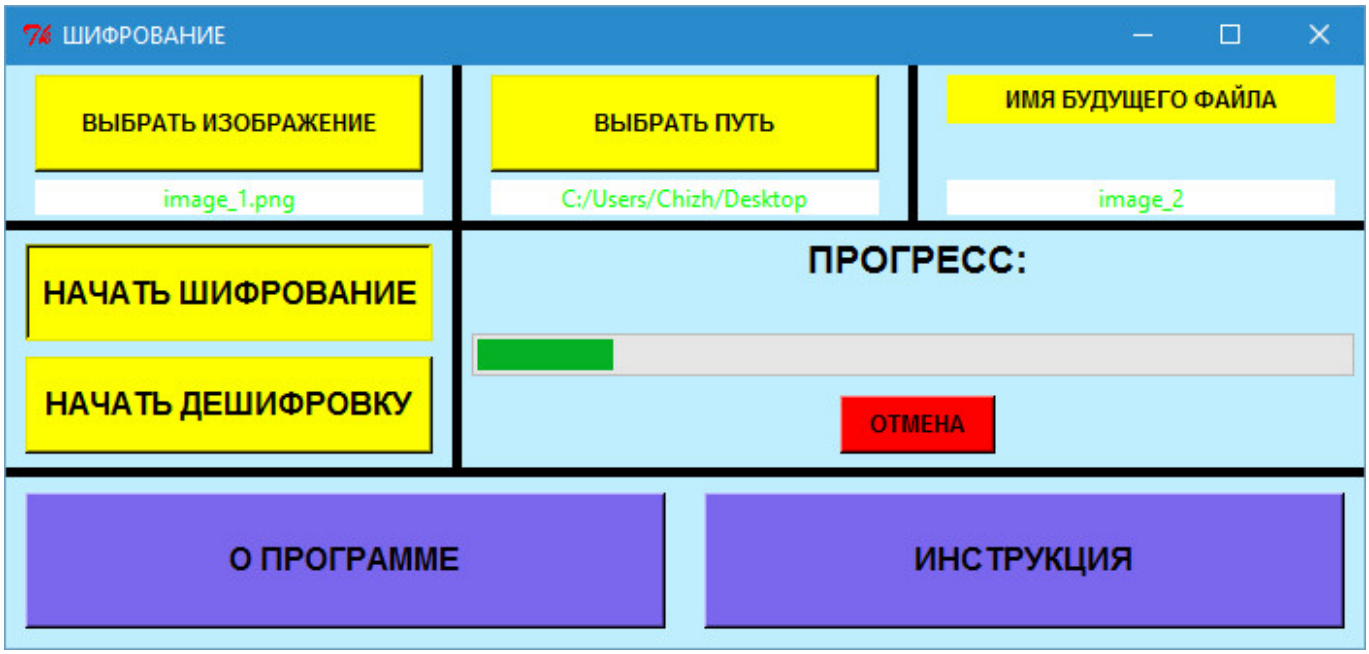

Рис. 4 Процесс

Если на ее месте появляется кнопка “Ошибка”, скорее всего, пользователь хотел расшифровать изображение, которое не подвергалось шифрованию. Для продолжения работы с ПО пользователь должен нажать на эту кнопку соответственно. В случае корректного завершения действия программы под строкой прогресса появляется кнопка “Готово”, после нажатия которой все значения и цвета интерфейса откатываются до исходного. При нажатии кнопки “О программе” можно прочитать о том, какие именно изображения она поддерживает и прочую информацию. А при нажатии кнопки “Инструкция” пользователь сможет ознакомится с кратким руководством по использованию программой. При выходе из программы появляется окно подтверждения действия.

Преимущества и недостатки

Преимущества:

1. Скорость работы. Шифрования изображения размером 1920x1080 в RGB занимает 15 минут, расшифровка – 7.5 минут.
2. Поддержка большинства часто используемых форматов.
3. Поддержка большинства цветовых моделей, в которых может быть представлена графика этих форматов.
4. Удобный и интуитивно понятный графический интерфейс.
5. Полное визуальное искажение исходных файлов, даже с большой контрастностью.
6. Достаточно устойчивый к взлому алгоритм.
7. Возможность шифрования уже зашифрованных изображений, что увеличивает надежность (оптимально для небольших изображений).

Недостатки:

1. Отсутствие поддержки некоторых форматов и цветовых моделей.
2. Отсутствие “резинового” интерфейса.
3. Только один алгоритм, при подборе которого программа перестает быть эффективной.
4. Приметность зашифрованных файлов, отсутствие стеганографических методов.
5. Увеличение размера и веса зашифрованных изображений.

Выводы

У программы есть большой потенциал для роста. На данный момент она может быть успешно использована в небольшом кругу лиц как дополнительная мера защиты. Для вывода программы на более высокий уровень мною планируется улучшить ее в следующих областях:

1. Создать “резиновый” интерфейс.
2. Добавить поддержку других форматов и цветовых моделей соответственно.
3. Добавление других алгоритмов генерации зашифрованного изображения, возможности их совмещения.
4. Возможность выбора пользователем соотношения времени на зашифровку и степенью защиты от взлома результата.
5. Улучшение дизайна графического интерфейса.
6. Внедрение стеганографических методов при условии оптимальной работы ПО.

Литература

1. Введение в криптографию / Под общ. ред. В. В. Ященко. – 3-е изд., доп. – М.: МЦНМО: «ЧеРо », 2000. – 288 с.
2. Python 3. Самое необходимое / Н. А. Прохоренок, В. А. Дронов. – СПб.: БХВ-Петербург, 2016. – 464 с.
3. habrahabr.ru
4. pythonworld.ru
5. ru.wikiversity.org
6. cyberforum.ru
7. effbot.org
8. wpython.org
9. stackoverflow.com

Приложение

```
from tkinter import *
from tkinter.filedialog import *
from tkinter.ttk import Progressbar
from PIL import Image
import os
import random
root=Tk()
in_progress=0
status_1=0
status_2=0
status_3=0
def instruction(event):
 top_instruction=Toplevel()
 top_instruction.title('ИНСТРУКЦИЯ')
 top_instruction.geometry('+400+600')
 top_instruction.maxsize(700, 400)
 top_instruction.minsize(700, 400)
 lab_instruction_1=Label(top_instruction,text="1. Чтобы зашифровать/дешифровать изображение, необходимо выбрать файл, с которым будет работать программа, место,",bg="LightBlue1",fg="black",anchor=W, justify=LEFT)
 lab_instruction_1.place(x=0,y=0,width=700,height=20)
 lab_instruction_2=Label(top_instruction,text="куда будет сохранен получившийся результат, а так же имя для будущего файла.",bg="LightBlue1",fg="black",anchor=W, justify=LEFT)
 lab_instruction_2.place(x=0,y=20,width=700,height=20)
 lab_instruction_3=Label(top_instruction,text='2. Чтобы выбрать файл, нажмите кнопку "ВЫБРАТЬ ИЗОБРАЖЕНИЕ", находящуюся в верхнем левом углу.',bg="LightBlue1",fg="black",anchor=W, justify=LEFT)
 lab_instruction_3.place(x=0,y=50,width=700,height=20)
 lab_instruction_4=Label(top_instruction,text='3. Чтобы выбрать место для сохранения, нажмите на кнопку "ВЫБРАТЬ ПАПКУ", которая расположена сверху посередине.',bg="LightBlue1",fg="black",anchor=W, justify=LEFT)
 lab_instruction_4.place(x=0,y=80,width=700,height=20)
 lab_instruction_5=Label(top_instruction,text='4. Чтобы ввести имя для будущего файла, воспользуйтесь строкой ввода, расположеенной в верхнем правом углу.',bg="LightBlue1",fg="black",anchor=W, justify=LEFT)
 lab_instruction_5.place(x=0,y=110,width=700,height=20)
 lab_instruction_6=Label(top_instruction,text='5. Если все выбрано корректно, кнопки "Начать шифрование/дешифровку" загорятся зеленым цветом. Нажмите на нужную.',bg="LightBlue1",fg="black",anchor=W, justify=LEFT)
 lab_instruction_6.place(x=0,y=140,width=700,height=20)
 lab_instruction_7=Label(top_instruction,text='6. В случае необходимости вы можете отменить процесс, нажав кнопку "Отмена", находящейся под полосой прогресса.',bg="LightBlue1",fg="black",anchor=W, justify=LEFT)
 lab_instruction_7.place(x=0,y=170,width=700,height=20)
 lab_instruction_8=Label(top_instruction,text='7. В случае появления ошибки нажмите одноименную кнопку для продолжения работы.',bg="LightBlue1",fg="black",anchor=W, justify=LEFT)
 lab_instruction_8.place(x=0,y=200,width=700,height=20)
 lab_instruction_9=Label(top_instruction,text='8. По завершении прогресса под полосой появится кнопка "Готово". Нажмите ее для продолжения работы с программой.',bg="LightBlue1",fg="black",anchor=W, justify=LEFT)
 lab_instruction_9.place(x=0,y=230,width=700,height=20)
```

```
def stoper(event):
 global stop
 stop=1
def away_no(event):
 top_away.destroy()
def away_yes(event):
 top_away.destroy()
 root.destroy()
def away():
 global but_away_yes
 global but_away_no
 global top_away
 global lab_away
 top_away=Toplevel()
 top_away.title('ВЫХОД ИЗ ПРОГРАММЫ')
 top_away.geometry('+500+400')
 top_away.maxsize(400, 200)
 top_away.minsize(400, 200)
 lab_away=Label(top_away,text="ВЫ УВЕРЕНЫ ЧТО ХОТИТЕ ВЫЙТИ?",fg="black",font='arial 12 bold')
 lab_away.place(x=30,y=20,width=350,height=20)
 but_away_no=Button(top_away,text="НЕТ",bg="green",fg="black",font='arial 12 bold')
 but_away_no.place(x=50,y=90,width=100,height=50)
 but_away_no.bind("<Button-1>",away_no)
 but_away_yes=Button(top_away,text="ДА",bg="red",fg="black",font='arial 12 bold')
 but_away_yes.place(x=250,y=90,width=100,height=50)
 but_away_yes.bind("<Button-1>",away_yes)
def about(event):
 top_about=Toplevel()
 top_about.title('О ПРОГРАММЕ')
 top_about.geometry('+400+600')
 top_about.maxsize(700, 200)
 top_about.minsize(700, 200)
 lab_formats=Label(top_about,text="Поддерживаемые форматы: PNG (.png), JPEG (.jpeg/.jpg), BMP (.bmp), TIFF (.tiff/.tif)",bg="LightBlue1",fg="black",anchor=W, justify=LEFT)
 lab_formats.place(x=0,y=0,width=700,height=30)
 lab_models=Label(top_about,text="Поддерживаемые цветовые модели: RGB, RGBA, RGBX, CMYK",bg="LightBlue1",fg="black",anchor=W, justify=LEFT)
 lab_models.place(x=0,y=40,width=700,height=30)
 lab_resolutions=Label(top_about,text='Поддерживаемые разрешения: до "9999 x 9999"',bg="LightBlue1",fg="black",anchor=W, justify=LEFT)
 lab_resolutions.place(x=0,y=80,width=700,height=30)
def file(event):
 global op
 global status_1
 global form
 global but_file
 global in_progress
 global image0
 global mode
 if in_progress==0:
  but_file["relief"]='sunken'
  status_1=0
```

```
  op=askopenfilename()
  but_file=Button(root,text="ВЫБРАТЬ
ИЗОБРАЖЕНИЕ",bg="SlateBlue2",fg="black",relief="raised",font='arial 9 bold')
  but_file.place(x=15,y=5,width=200,height=50)
  but_file.bind("<Button-1>",file)
  but_file["relief"]='raised'
  count_1=0
  count_2=0
  count_3=0
  if len(op)==0:
   lab_file["text"]="<ПУСТО>"
   lab_file["fg"]='red'
   status_1=0
  else:
   while op[count_1]!='/':
    count_1=count_1-1
   name=op[count_1+1:len(op)]
   while op[count_3]!='.':
    count_3=count_3+1
   form=op[count_3+1:len(op)]
   if form!='png' and form!='tif' and form!='tiff' and form!='jpg' and form!='jpeg' and form!='bmp' and
form!='BMP':
    lab_file["text"]="ВЫБРАНО НЕ ИЗОБРАЖЕНИЕ"
    lab_file["fg"]='red'
    status_1=0
   else:
    lab_file["text"]=name
    lab_file["fg"]='green'
    status_1=1
    try:
     image0=Image.open(op)
    except Exception:
     lab_file["text"]="ФАЙЛ НЕ СОДЕРЖИТ ГРАФИКУ"
     lab_file["fg"]='red'
     status_1=0
     count_2=1
    else:
     count_2=2
   if count_2==2:
    mode=image0.mode
    if mode!='RGB' and mode!='RGBA' and mode!='RGBX' and mode!='CMYK':
     lab_file["text"]="ПАЛИТРА НЕ ПОДДЕРЖИВАЕТСЯ"
     lab_file["fg"]='red'
     status_1=0
  but_file["relief"]='raised'
  if status_1==1 and status_2==1 and status_3==1:
   but_shifr["bg"]='green'
   but_deshifr["bg"]='green'
  else:
   but_shifr["bg"]='red'
   but_deshifr["bg"]='red'
def catalog(event):
 global in_progress
```

```
 global status_2
 global sa
 global but_catalog
 if in_progress==0:
  but_catalog["relief"]='sunken'
  status_2=0
  sa=askdirectory()
  but_catalog=Button(root,text="ВЫБРАТЬ ПУТЬ",bg="SlateBlue2",fg="black",font='arial 9 bold')
  but_catalog.place(x=250,y=5,width=200,height=50)
  but_catalog.bind("<Button-1>",catalog)
  but_catalog["relief"]='raised'
  if len(sa)<30:
   name=sa
   status_2=1
  else:
   name='...'+sa[-27:len(sa)]
  lab_catalog["text"]=name
  lab_catalog["fg"]='green'
  status_2=1
  if sa=='':
   lab_catalog["text"]="<ПУСТО>"
   lab_catalog["fg"]='red'
   status_2=0
  if status_1==1 and status_2==1 and status_3==1:
   but_shifr["bg"]='green'
   but_deshifr["bg"]='green'
  else:
   but_shifr["bg"]='red'
   but_deshifr["bg"]='red'
def name(event):
 global name
 global status_3
 global in_progress
 if in_progress==0:
  status_3=0
  name = ent_name.get()
  lab_name_2["text"]=''
  ent_name["fg"]='green'
  status_3=1
  for i in range(len(name)):
   if name[i]=='<' or name[i]=='>' or name[i]==':' or name[i]=='«' or name[i]=='/' or name[i]=='?' or
name[i]=='*' or name[i]=='?' or name[i]=='.':
    lab_name_2["text"]='НЕПРАВИЛЬНОЕ ИМЯ ФАЙЛА'
    lab_name_2["fg"]='red'
    ent_name["fg"]='red'
    status_3=0
  if name=='':
   lab_name_2["text"]='ИМЯ НЕ МОЖЕТ БЫТЬ ПУСТЫМ'
   lab_name_2["fg"]='red'
   status_3=0
  if status_1==1 and status_2==1 and status_3==1:
   but_shifr["bg"]='green'
   but_deshifr["bg"]='green'
```

```
  else:
    but_shifr["bg"]='red'
    but_deshifr["bg"]='red'
def ready(event):
 global status_1
 global status_2
 global status_3
 global but_shifr
 global but_deshifr
 global in_progress
 lab_name_3.destroy()
 but_ready.destroy()
 but_shifr=Button(root,text="НАЧАТЬ ШИФРОВАНИЕ",bg="red",fg="black",font='arial 12 bold')
 but_shifr.place(x=10,y=92,width=210,height=50)
 but_shifr["relief"]='raised'
 but_shifr.bind("<Button-1>",shifr)
 but_deshifr=Button(root,text="НАЧАТЬ ДЕШИФРОВКУ",bg="red",fg="black",font='arial 12 bold')
 but_deshifr.place(x=10,y=150,width=210,height=50)
 but_deshifr["relief"]='raised'
 but_deshifr.bind("<Button-1>",deshifr)
 but_file["bg"]='SlateBlue2'
 but_catalog["bg"]='SlateBlue2'
 but_shifr["bg"]='red'
 but_deshifr["bg"]='red'
 lab_name_1["bg"]='SlateBlue2'
 lab_file["text"]='<ПУСТО>'
 lab_catalog["text"]='<ПУСТО>'
 lab_file["fg"]='red'
 lab_catalog["fg"]='red'
 lab_name_1["fg"]='black'
 ent_name.delete(0, END)
 bar["value"]=0
 status_1=0
 status_2=0
 status_3=0
 in_progress=0
 root.update()
def fail(event):
 global status_1
 global status_2
 global status_3
 global but_shifr
 global but_deshifr
 global in_progress
 lab_name_3.destroy()
 but_fail.destroy()
 but_stop.destroy()
 but_shifr=Button(root,text="НАЧАТЬ ШИФРОВАНИЕ",bg="red",fg="black",font='arial 12 bold')
 but_shifr.place(x=10,y=92,width=210,height=50)
 but_shifr["relief"]='raised'
 but_shifr.bind("<Button-1>",shifr)
 but_deshifr=Button(root,text="НАЧАТЬ ДЕШИФРОВКУ",bg="red",fg="black",font='arial 12 bold')
 but_deshifr.place(x=10,y=150,width=210,height=50)
```

```
 but_deshifr["relief"]='raised'
 but_deshifr.bind("<Button-1>",deshifr)
 but_file["bg"]='SlateBlue2'
 but_catalog["bg"]='SlateBlue2'
 but_shifr["bg"]='red'
 but_deshifr["bg"]='red'
 lab_name_1["bg"]='SlateBlue2'
 lab_file["text"]='<ПУСТО>'
 lab_catalog["text"]='<ПУСТО>'
 lab_file["fg"]='red'
 lab_catalog["fg"]='red'
 lab_name_1["fg"]='black'
 ent_name.delete(0, END)
 bar["value"]=0
 status_1=0
 status_2=0
 status_3=0
 in_progress=0
 root.update()
def shifr(event):
 global lab_name_3
 global stop
 global but_ready
 global but_shifr
 global in_progress
 global but_stop
 if status_1==1 and status_2==1 and status_3==1 and in_progress==0:
  stop=0
  in_progress=1
  but_shifr["relief"]='sunken'
  but_file["bg"]='yellow'
  but_catalog["bg"]='yellow'
  but_shifr["bg"]='yellow'
  but_deshifr["bg"]='yellow'
  lab_name_1["bg"]='yellow'
  lab_name_3=Label(root,text=name,bg="white",fg="green")
  lab_name_3.place(x=485,y=59,width=200,height=18)
  key=0
  if (form=='jpg') and (mode=='RGB' or mode=='RGBA' or mode=='RGBX'):
   image0.save('image_delete.png', 'PNG')
   image1=Image.open('image_delete.png')
   key=0
  elif (form=='jpeg') and (mode=='RGB' or mode=='RGBA' or mode=='RGBX'):
   image0.save('image_delete.png', 'PNG')
   image1=Image.open('image_delete.png')
   key=1
  elif (form=='jpg') and (mode=='CMYK'):
   image0.save('image_delete.tiff', 'TIFF')
   image1=Image.open('image_delete.tiff')
   key=0
  elif (form=='jpeg') and (mode=='CMYK'):
   image0.save('image_delete.tiff', 'TIFF')
   image1=Image.open('image_delete.tiff')
```

```
 key=1
elif (form=='png'):
 image1=image0
elif (form=='tiff'):
 image1=image0
elif (form=='tif'):
 image1=image0
elif (form=='bmp'):
 image1=image0
elif (form=='BMP'):
 image1=image0
width1=image1.size[0]
height1=image1.size[1]
size=width1*height1
pix1=image1.load()
status=0
bar["maximum"]=size
but_stop=Button(root,text="OTMEHA",bg="red",fg="black",font='arial 9 bold')
but_stop.place(x=430,y=170,width=80,height=30)
but_stop.bind("<Button-1>",stoper)
if mode=='RGBA':
 image2=Image.new('RGBA', (width1*2, height1*3), (255, 255, 255, 255))
 for j in range(height1):
  if stop==1:
   break
  for i in range(width1):
   if stop==1:
    break
   r0=pix1[i, j][0]
   g0=pix1[i, j][1]
   b0=pix1[i, j][2]
   a0=pix1[i, j][3]
   r1=r0//100
   r2=(r0-r1*100)//10
   r3=r0-r1*100-r2*10
   g1=g0//100
   g2=(g0-g1*100)//10
   g3=g0-g1*100-g2*10
   b1=b0//100
   b2=(b0-b1*100)//10
   b3=b0-b1*100-b2*10
   a1=a0//100
   a2=(a0-a1*100)//10
   a3=a0-a1*100-a2*10
   i1=i//1000
   i2=(i-i1*1000)//100
   i3=(i-i1*1000-i2*100)//10
   i4=i-i1*1000-i2*100-i3*10
   j1=j//1000
   j2=(j-j1*1000)//100
   j3=(j-j1*1000-j2*100)//10
   j4=j-j1*1000-j2*100-j3*10
   random1=random.randrange(0, 241, 10)
```

```
   random2=random.randrange(0, 241, 10)
   random3=random.randrange(0, 241, 10)
   random4=random.randrange(0, 241, 10)
   random5=random.randrange(0, 241, 10)
   random6=random.randrange(0, 241, 10)
   random7=random.randrange(0, 241, 10)
   random8=random.randrange(0, 241, 10)
   random9=random.randrange(0, 241, 10)
   random10=random.randrange(0, 241, 10)
   random11=random.randrange(0, 241, 10)
   random12=random.randrange(0, 241, 10)
   random13=random.randrange(0, 241, 10)
   random14=random.randrange(0, 241, 10)
   random15=random.randrange(0, 241, 10)
   random16=random.randrange(0, 241, 10)
   random17=random.randrange(0, 241, 10)
   random18=random.randrange(0, 241, 10)
   random19=random.randrange(0, 241, 10)
   random20=random.randrange(0, 241, 10)
   random21=random.randint(0, 255)
   random22=random.randint(0, 255)
   random23=random.randint(0, 255)
   random24=random.randint(2, 255)
   color=(0, 0, 0, 0)
   while color!=(255, 255, 255, 255):
    i5=random.randrange(0, width1*2, 2)
    j5=random.randrange(0, height1*3, 3)
    color=image2.getpixel((i5, j5))
   image2.putpixel((i5, j5), (r1+random1, r2+random2, r3+random3, g1+random4))
   image2.putpixel((i5+1, j5), (g2+random5, g3+random6, b1+random7, b2+random8))
   image2.putpixel((i5, j5+1), (b3+random9, a1+random10, a2+random11, a3+random12))
   image2.putpixel((i5+1, j5+1), (i1+random13, i2+random14, i3+random15, i4+random16))
   image2.putpixel((i5, j5+2), (j1+random17, j2+random18, j3+random19, j4+random20))
   if (i5+1==width1*2-1) and (j5+2==height1*3-1) and (form=='jpg' or form=='jpeg'):
    image2.putpixel((i5+1, j5+2), (random21, random22, random23, h))
   else:
    image2.putpixel((i5+1, j5+2), (random21, random22, random23, random24))
   status=status+1
   bar["value"]=status
   root.update()
elif mode=='RGB':
 image2 = Image.new('RGB', (width1*2, height1*3), (255, 255, 255))
 for j in range(height1):
  if stop==1:
   break
  for i in range(width1):
   if stop==1:
    break
   r0=pix1[i, j][0]
   g0=pix1[i, j][1]
   b0=pix1[i, j][2]
   r1=r0//100
   r2=(r0-r1*100)//10
```

```
    r3=r0-r1*100-r2*10
    g1=g0//100
    g2=(g0-g1*100)//10
    g3=g0-g1*100-g2*10
    b1=b0//100
    b2=(b0-b1*100)//10
    b3=b0-b1*100-b2*10
    i1=i//1000
    i2=(i-i1*1000)//100
    i3=(i-i1*1000-i2*100)//10
    i4=i-i1*1000-i2*100-i3*10
    j1=j//1000
    j2=(j-j1*1000)//100
    j3=(j-j1*1000-j2*100)//10
    j4=j-j1*1000-j2*100-j3*10
    random1= random.randrange(0, 241, 10)
    random2= random.randrange(0, 241, 10)
    random3= random.randrange(0, 241, 10)
    random4= random.randrange(0, 241, 10)
    random5= random.randrange(0, 241, 10)
    random6= random.randrange(0, 241, 10)
    random7= random.randrange(0, 241, 10)
    random8= random.randrange(0, 241, 10)
    random9= random.randrange(0, 241, 10)
    random10=random.randrange(0, 241, 10)
    random11=random.randrange(0, 241, 10)
    random12=random.randrange(0, 241, 10)
    random13=random.randrange(0, 241, 10)
    random14=random.randrange(0, 241, 10)
    random15=random.randrange(0, 241, 10)
    random16=random.randrange(0, 241, 10)
    random17=random.randrange(0, 241, 10)
    random18=random.randint(2, 255)
    color=(0, 0, 0)
    while color!=(255, 255, 255):
     i5=random.randrange(0, width1*2, 2)
     j5=random.randrange(0, height1*3, 3)
     color=image2.getpixel((i5, j5))
    image2.putpixel((i5, j5), (r1+random1, r2+random2, r3+random3))
    image2.putpixel((i5+1, j5), (g1+random4, g2+random5, g3+random6))
    image2.putpixel((i5, j5+1), (b1+random7, b2+random8, b3+random9))
    image2.putpixel((i5+1, j5+1), (i1+random10, i2+random11, i3+random12))
    image2.putpixel((i5, j5+2), (i4+random13, j1+random14, j2+random15))
    if (i5+1==width1*2-1) and (j5+2==height1*3-1) and (form=='jpg' or form=='jpeg'):
     image2.putpixel((i5+1, j5+2), (j3+random16, j4+random17, key))
    else:
     image2.putpixel((i5+1, j5+2), (j3+random16, j4+random17, random18))
    status=status+1
    bar["value"]=status
    root.update()
elif mode=='CMYK':
 image2=Image.new('CMYK', (width1*2, height1*3), (100, 100, 100, 100))
 for j in range(height1):
```

```
if stop==1:
 break
for i in range(width1):
 if stop==1:
  break
 c0=pix1[i, j][0]
 m0=pix1[i, j][1]
 y0=pix1[i, j][2]
 k0=pix1[i, j][3]
 c1=c0//100
 c2=(c0-c1*100)//10
 c3=c0-c1*100-c2*10
 m1=m0//100
 m2=(m0-m1*100)//10
 m3=m0-m1*100-m2*10
 y1=y0//100
 y2=(y0-y1*100)//10
 y3=y0-y1*100-y2*10
 k1=k0//100
 k2=(k0-k1*100)//10
 k3=k0-k1*100-k2*10
 i1=i//1000
 i2=(i-i1*1000)//100
 i3=(i-i1*1000-i2*100)//10
 i4=i-i1*1000-i2*100-i3*10
 j1=j//1000
 j2=(j-j1*1000)//100
 j3=(j-j1*1000-j2*100)//10
 j4=j-j1*1000-j2*100-j3*10
 random1=random.randrange(0, 91, 10)
 random2=random.randrange(0, 91, 10)
 random3=random.randrange(0, 91, 10)
 random4=random.randrange(0, 91, 10)
 random5=random.randrange(0, 91, 10)
 random6=random.randrange(0, 91, 10)
 random7=random.randrange(0, 91, 10)
 random8=random.randrange(0, 91, 10)
 random9=random.randrange(0, 91, 10)
 random10=random.randrange(0, 91, 10)
 random11=random.randrange(0, 91, 10)
 random12=random.randrange(0, 91, 10)
 random13=random.randrange(0, 91, 10)
 random14=random.randrange(0, 91, 10)
 random15=random.randrange(0, 91, 10)
 random16=random.randrange(0, 91, 10)
 random17=random.randrange(0, 91, 10)
 random18=random.randrange(0, 91, 10)
 random19=random.randrange(0, 91, 10)
 random20=random.randrange(0, 91, 10)
 random21=random.randint(0, 100)
 random22=random.randint(0, 100)
 random23=random.randint(0, 100)
 random24=random.randint(2, 100)
```

```
   color=(0, 0, 0, 0)
   while color!=(100, 100, 100, 100):
    i5=random.randrange(0, width1*2, 2)
    j5=random.randrange(0, height1*3, 3)
    color = image2.getpixel((i5, j5))
   image2.putpixel((i5, j5), (c1+random1, c2+random2, c3+random3, m1+random4))
   image2.putpixel((i5+1, j5), (m2+random5, m3+random6, y1+random7, y2+random8))
   image2.putpixel((i5, j5+1), (y3+random9, k1+random10, k2+random11, k3+random12))
   image2.putpixel((i5+1, j5+1), (i1+random13, i2+random14, i3+random15, i4+random16))
   image2.putpixel((i5, j5+2), (j1+random17, j2+random18, j3+random19, j4+random20))
   if (i5+1==width1*2-1) and (j5+2==height1*3-1) and (form=='jpg' or form=='jpeg'):
    image2.putpixel((i5+1, j5+2), (random21, random22, random23, key))
   else:
    image2.putpixel((i5+1, j5+2), (random21, random22, random23, random24))
   status=status+1
   bar["value"]=status
   root.update()
if mode=='RGBX':
 image2=Image.new('RGBX', (width1*2, height1*3), (255, 255, 255, 255))
 for j in range(height1):
  if stop==1:
   break
  for i in range(width1):
   if stop==1:
    break
   r0=pix1[i, j][0]
   g0=pix1[i, j][1]
   b0=pix1[i, j][2]
   x0=pix1[i, j][3]
   r1=r0//100
   r2=(r0-r1*100)//10
   r3=r0-r1*100-r2*10
   g1=g0//100
   g2=(g0-g1*100)//10
   g3=g0-g1*100-g2*10
   b1=b0//100
   b2=(b0-b1*100)//10
   b3=b0-b1*100-b2*10
   x1=x0//100
   x2=(x0-x1*100)//10
   x3=x0-x1*100-x2*10
   i1=i//1000
   i2=(i-i1*1000)//100
   i3=(i-i1*1000-i2*100)//10
   i4=i-i1*1000-i2*100-i3*10
   j1=j//1000
   j2=(j-j1*1000)//100
   j3=(j-j1*1000-j2*100)//10
   j4=j-j1*1000-j2*100-j3*10
   random1=random.randrange(0, 241, 10)
   random2=random.randrange(0, 241, 10)
   random3=random.randrange(0, 241, 10)
   random4=random.randrange(0, 241, 10)
```

```
   random5=random.randrange(0, 241, 10)
   random6=random.randrange(0, 241, 10)
   random7=random.randrange(0, 241, 10)
   random8=random.randrange(0, 241, 10)
   random9=random.randrange(0, 241, 10)
   random10=random.randrange(0, 241, 10)
   random11=random.randrange(0, 241, 10)
   random12=random.randrange(0, 241, 10)
   random13=random.randrange(0, 241, 10)
   random14=random.randrange(0, 241, 10)
   random15=random.randrange(0, 241, 10)
   random16=random.randrange(0, 241, 10)
   random17=random.randrange(0, 241, 10)
   random18=random.randrange(0, 241, 10)
   random19=random.randrange(0, 241, 10)
   random20=random.randrange(0, 241, 10)
   random21=random.randint(0, 255)
   random22=random.randint(0, 255)
   random23=random.randint(0, 255)
   random24=random.randint(2, 255)
   color=(0, 0, 0, 0)
   while color!=(255, 255, 255, 255):
    i5=random.randrange(0, width1*2, 2)
    j5=random.randrange(0, height1*3, 3)
    color=image2.getpixel((i5, j5))
   image2.putpixel((i5, j5), (r1+random1, r2+random2, r3+random3, x1+random4))
   image2.putpixel((i5+1, j5), (g1+random5, g2+random6, g3+random7, x2+random8))
   image2.putpixel((i5, j5+1), (b1+random9, b2+random10, b3+random11, x3+random12))
   image2.putpixel((i5+1, j5+1), (i1+random13, i2+random14, i3+random15, i4+random16))
   image2.putpixel((i5, j5+2), (j1+random17, j2+random18, j3+random19, j4+random20))
   if (i5+1==width1*2-1) and (j5+2==height1*3-1) and (form=='jpg' or form=='jpeg'):
    image2.putpixel((i5+1, j5+2), (random21, random22, random23, h))
   else:
    image2.putpixel((i5+1, j5+2), (random21, random22, random23, random24))
   status = status+1
   bar["value"]=status
   root.update()
if stop==0:
 if (form=='jpeg' or form=='jpg') and (mode=='RGB' or mode=='RGBA' or mode=='RGBX'):
  image2.save(sa+'/'+name+'.png', 'PNG')
  del image1
  os.remove('image_delete.png')
 elif (form=='jpeg' or form=='jpg') and (mode=='CMYK'):
  image2.save(sa+'/'+name+'.tiff', 'TIFF')
  del image1
  os.remove('image_delete.tiff')
 elif (form=='png'):
  image2.save(sa+'/'+name+'.png', 'PNG')
 elif (form=='tiff'):
  image2.save(sa+'/'+name+'.tiff', 'TIFF')
 elif (form=='tif'):
  image2.save(sa+'/'+name+'.tif', 'TIFF')
 elif form=='bmp':
```

```
     image2.save(sa+'/'+name+'.bmp', 'BMP')
    elif form=='BMP':
     image2.save(sa+'/'+name+'.bmp', 'BMP')
   else:
    bar["value"]=size
    root.update()
    if (form=='jpeg' or form=='jpg') and (mode=='RGB' or mode=='RGBA' or mode=='RGBX'):
     del image1
     os.remove('image_delete.png')
    elif (form=='jpeg' or form=='jpg') and (mode=='CMYK'):
     del image1
     os.remove('image_delete.tiff')
  but_stop.destroy()
  but_ready = Button(root,text="ГОТОВО. НАЖМИТЕ ЧТОБЫ
ПРОДОЛЖИТЬ",bg="green",fg="black",font='arial 9 bold')
  but_ready.place(x=320,y=170,width=300,height=30)
  but_ready.bind("<Button-1>",ready)
  but_shifr=Button(root,text="НАЧАТЬ ШИФРОВАНИЕ",bg="yellow",fg="black",font='arial 12 bold')
  but_shifr.place(x=10,y=92,width=210,height=50)
  but_shifr["relief"]='raised'
  but_shifr.bind("<Button-1>",shifr)
def deshifr(event):
 global but_stop
 global stop
 global but_fail
 global but_deshifr
 global in_progress
 global but_ready
 global lab_name_3
 if status_1==1 and status_2==1 and status_3==1 and in_progress==0:
  stop=0
  in_progress=1
  but_deshifr["relief"]='sunken'
  but_file["bg"]='yellow'
  but_catalog["bg"]='yellow'
  but_shifr["bg"]='yellow'
  but_deshifr["bg"]='yellow'
  lab_name_1["bg"]='yellow'
  lab_name_3=Label(root,text=name,bg="white",fg="green")
  lab_name_3.place(x=485,y=59,width=200,height=18)
  but_stop=Button(root,text="ОТМЕНА",bg="red",fg="black",font='arial 9 bold')
  but_stop.place(x=430,y=170,width=80,height=30)
  but_stop.bind("<Button-1>",stoper)
  image1=Image.open(op)
  mode=image1.mode
  width1=image1.size[0]
  height1=image1.size[1]
  pix1=image1.load()
  key_place=image1.getpixel((width1-1, height1-1))
  size=width1*height1//6
  bar["maximum"]=size
  status=0
  k=-1
```

```
try:
 if (mode=='CMYK'):
  image2=Image.new('CMYK', (width1//2, height1//3), (100, 100, 100, 100))
  key=key_place[3]
 elif (mode=='RGB'):
  image2=Image.new('RGB', (width1//2, height1//3), (255, 255, 255))
  key=key_place[2]
 elif (mode=='RGBA'):
  image2=Image.new('RGBA', (width1//2, height1//3), (255, 255, 255, 255))
  key=key_place[3]
 elif (mode=='RGBX'):
  image2=Image.new('RGBX', (width1//2, height1//3), (255, 255, 255, 255))
  key=key_place[3]
 if mode=='RGBA':
  for j in range(height1):
   k=k+1
   p=-1
   if stop==1:
    break
   for i in range(width1):
    p=p+1
    if stop==1:
     break
    if (k%3==0) and (p%2==0):
     r0=pix1[i, j][0]%10
     g0=pix1[i, j][1]%10
     b0=pix1[i, j][2]%10
     a0=pix1[i, j][3]%10
     r1=pix1[i+1, j][0]%10
     g1=pix1[i+1, j][1]%10
     b1=pix1[i+1, j][2]%10
     a1=pix1[i+1, j][3]%10
     r2=pix1[i, j+1][0]%10
     g2=pix1[i, j+1][1]%10
     b2=pix1[i, j+1][2]%10
     a2=pix1[i, j+1][3]%10
     r3=pix1[i+1, j+1][0]%10
     g3=pix1[i+1, j+1][1]%10
     b3=pix1[i+1, j+1][2]%10
     a3=pix1[i+1, j+1][3]%10
     r4=pix1[i, j+2][0]%10
     g4=pix1[i, j+2][1]%10
     b4=pix1[i, j+2][2]%10
     a4=pix1[i, j+2][3]%10
     r5=pix1[i+1, j+2][0]%10
     g5=pix1[i+1, j+2][1]%10
     b5=pix1[i+1, j+2][2]%10
     a5=pix1[i+1, j+2][3]%10
     i1=r3*1000+g3*100+b3*10+a3
     j1=r4*1000+g4*100+b4*10+a4
     red=r0*100+g0*10+b0
     green=a0*100+r1*10+g1
     blue=b1*100+a1*10+r2
```

```
    alpha=g2*100+b2*10+a2
    image2.putpixel((i1, j1), (red, green, blue, alpha))
    status=status+1
    bar["value"]=status
    root.update()
elif mode=='RGB':
 for j in range(height1):
  k=k+1
  p=-1
  if stop==1:
   break
  for i in range(width1):
   p=p+1
   if stop==1:
    break
   if (k%3==0) and (p%2==0):
    r0=pix1[i, j][0]%10
    g0=pix1[i, j][1]%10
    b0=pix1[i, j][2]%10
    r1=pix1[i+1, j][0]%10
    g1=pix1[i+1, j][1]%10
    b1=pix1[i+1, j][2]%10
    r2=pix1[i, j+1][0]%10
    g2=pix1[i, j+1][1]%10
    b2=pix1[i, j+1][2]%10
    r3=pix1[i+1, j+1][0]%10
    g3=pix1[i+1, j+1][1]%10
    b3=pix1[i+1, j+1][2]%10
    r4=pix1[i, j+2][0]%10
    g4=pix1[i, j+2][1]%10
    b4=pix1[i, j+2][2]%10
    r5=pix1[i+1, j+2][0]%10
    g5=pix1[i+1, j+2][1]%10
    b5=pix1[i+1, j+2][2]%10
    i1=r3*1000+g3*100+b3*10+r4
    j1=g4*1000+b4*100+r5*10+g5
    red=r0*100+g0*10+b0
    green=r1*100+g1*10+b1
    blue=r2*100+g2*10+b2
    image2.putpixel((i1, j1), (red, green, blue))
    status=status+1
    bar["value"]=status
    root.update()
elif mode=='CMYK':
 for j in range(height1):
  k=k+1
  p=-1
  if stop==1:
   break
  for i in range(width1):
   p=p+1
   if stop==1:
    break
```

```
    if (k%3==0) and (p%2==0):
     c0=pix1[i, j][0]%10
     m0=pix1[i, j][1]%10
     y0=pix1[i, j][2]%10
     k0=pix1[i, j][3]%10
     c1=pix1[i+1, j][0]%10
     m1=pix1[i+1, j][1]%10
     y1=pix1[i+1, j][2]%10
     k1=pix1[i+1, j][3]%10
     c2=pix1[i, j+1][0]%10
     m2=pix1[i, j+1][1]%10
     y2=pix1[i, j+1][2]%10
     k2=pix1[i, j+1][3]%10
     c3=pix1[i+1, j+1][0]%10
     m3=pix1[i+1, j+1][1]%10
     y3=pix1[i+1, j+1][2]%10
     k3=pix1[i+1, j+1][3]%10
     c4=pix1[i, j+2][0]%10
     m4=pix1[i, j+2][1]%10
     y4=pix1[i, j+2][2]%10
     k4=pix1[i, j+2][3]%10
     c5=pix1[i+1, j+2][0]%10
     m5=pix1[i+1, j+2][1]%10
     y5=pix1[i+1, j+2][2]%10
     k5=pix1[i+1, j+2][3]%10
     i1=c3*1000+m3*100+y3*10+k3
     j1=c4*1000+m4*100+y4*10+k4
     c6=c0*100+m0*10+y0
     m6=k0*100+c1*10+m1
     y6=y1*100+k1*10+c2
     k6=m2*100+y2*10+k2
     image2.putpixel((i1, j1), (c6, m6, y6, k6))
     status=status+1
     bar["value"]=status
     root.update()
elif mode=='RGBX':
 for j in range(height1):
  k=k+1
  p=-1
  if stop==1:
   break
  for i in range(width1):
   p=p+1
   if stop==1:
    break
   if (k%3==0) and (p%2==0):
    r0=pix1[i, j][0]%10
    g0=pix1[i, j][1]%10
    b0=pix1[i, j][2]%10
    x0=pix1[i, j][3]%10
    r1=pix1[i+1, j][0]%10
    g1=pix1[i+1, j][1]%10
    b1=pix1[i+1, j][2]%10
```

```
      x1=pix1[i+1, j][3]%10
      r2=pix1[i, j+1][0]%10
      g2=pix1[i, j+1][1]%10
      b2=pix1[i, j+1][2]%10
      x2=pix1[i, j+1][3]%10
      r3=pix1[i+1, j+1][0]%10
      g3=pix1[i+1, j+1][1]%10
      b3=pix1[i+1, j+1][2]%10
      x3=pix1[i+1, j+1][3]%10
      r4=pix1[i, j+2][0]%10
      g4=pix1[i, j+2][1]%10
      b4=pix1[i, j+2][2]%10
      x4=pix1[i, j+2][3]%10
      r5=pix1[i+1, j+2][0]%10
      g5=pix1[i+1, j+2][1]%10
      b5=pix1[i+1, j+2][2]%10
      x5=pix1[i+1, j+2][3]%10
      i1=r3*1000+g3*100+b3*10+x3
      j1=r4*1000+g4*100+b4*10+x4
      red=r0*100+g0*10+b0
      green=x0*100+r1*10+g1
      blue=b1*100+x1*10+r2
      x=g2*100+b2*10+x2
      image2.putpixel((i1, j1), (red, green, blue, x))
      status=status+1
      bar["value"]=status
      root.update()
   if stop==0:
    if (form=='png' or form=='tiff') and key==0:
     image2.save(sa+'/'+name+'.jpg', 'JPEG')
    elif (form=='png' or form=='tiff') and key==1:
     image2.save(sa+'/'+name+'.jpeg', 'JPEG')
    elif (form=='png') and key>1:
     image2.save(sa+'/'+name+'.png', 'PNG')
    elif (form=='tiff') and key>1:
     image2.save(sa+'/'+name+'.tiff', 'TIFF')
    elif (form=='tif'):
     image2.save(sa+'/'+name+'.tif', 'TIFF')
    elif form=='bmp':
     image2.save(sa+'/'+name+'.bmp', 'BMP')
   else:
    bar["value"]=size
    root.update()
  except Exception:
   but_fail=Button(root,text="ОШИБКА.                НАЖМИТЕ                ЧТОБЫ
ПРОДОЛЖИТЬ",bg="red",fg="black",font='arial 9 bold')
   but_fail.place(x=320,y=170,width=300,height=30)
   but_fail.bind("<Button-1>",fail)
   bar["value"]=size
  else:
   but_ready=Button(root,text="ГОТОВО.                НАЖМИТЕ                ЧТОБЫ
ПРОДОЛЖИТЬ",bg="green",fg="black",font='arial 9 bold')
   but_ready.place(x=320,y=170,width=300,height=30)
```

```
   but_ready.bind("<Button-1>",ready)
  finally:
   but_stop.destroy()
   but_deshifr=Button(root,text="НАЧАТЬ  ДЕШИФРОВКУ",bg="yellow",fg="black",font='arial  12
bold')
   but_deshifr.place(x=10,y=150,width=210,height=50)
   but_deshifr.bind("<Button-1>",deshifr)
   but_deshifr["relief"]='raised'
root.title('ШИФРОВАНИЕ')
root.geometry('+400+200')
root.maxsize(700, 300)
root.minsize(700, 300)
place_1=Frame(root,width=700,heigh=300,bg='LightBlue1')
place_1.place(x=0,y=0)
border_1=Frame(root,width=5,heigh=208,bg='black')
border_1.place(x=230,y=0)
border_2=Frame(root,width=5,heigh=80,bg='black')
border_2.place(x=465,y=0)
border_3=Frame(root,width=700,heigh=5,bg='black')
border_3.place(x=0,y=80)
border_4=Frame(root,width=700,heigh=5,bg='black')
border_4.place(x=0,y=207)
but_file=Button(root,text="ВЫБРАТЬ  ИЗОБРАЖЕНИЕ",bg="SlateBlue2",fg="black",font='arial  9
bold')
but_file.place(x=15,y=5,width=200,height=50)
but_catalog=Button(root,text="ВЫБРАТЬ ПУТЬ",bg="SlateBlue2",fg="black",font='arial 9 bold')
but_catalog.place(x=250,y=5,width=200,height=50)
but_shifr=Button(root,text="НАЧАТЬ ШИФРОВАНИЕ",bg="red",fg="black",font='arial 12 bold')
but_shifr.place(x=10,y=92,width=210,height=50)
but_deshifr=Button(root,text="НАЧАТЬ ДЕШИФРОВКУ",bg="red",fg="black",font='arial 12 bold')
but_deshifr.place(x=10,y=150,width=210,height=50)
but_about=Button(root,text="О ПРОГРАММЕ",bg="SlateBlue2",fg="black",font='arial 12 bold')
but_about.place(x=10,y=220,width=330,height=70)
but_instruction=Button(root,text="ИНСТРУКЦИЯ",bg="SlateBlue2",fg="black",font='arial 12 bold')
but_instruction.place(x=360,y=220,width=330,height=70)
lab_name_1=Label(root,text="ИМЯ  БУДУЩЕГО  ФАЙЛА",bg="SlateBlue2",fg="black",font='arial  9
bold')
lab_name_1.place(x=485,y=5,width=200,height=25)
lab_file=Label(root,text="<ПУСТО>",bg="white",fg="red")
lab_file.place(x=15,y=59,width=200,height=18)
lab_catalog=Label(root,text="<ПУСТО>",bg="white",fg="red")
lab_catalog.place(x=250,y=59,width=200,height=18)
lab_progress=Label(root,text="ПРОГРЕСС:",bg="LightBlue1",fg="black",font='arial 14 bold')
lab_progress.place(x=410,y=90,width=120,height=20)
lab_name_2=Label(root,text="",bg="LightBlue1",fg="green")
lab_name_2.place(x=485,y=32,width=200,height=25)
ent_name=Entry(root,bg="white",fg="black")
ent_name.place(x=485,y=59,width=200,height=18)
bar=Progressbar(root,orient='horizontal',length=455,mode='determinate',value=0)
bar.place(x=240,y=138)
but_file.bind("<Button-1>",file)
but_catalog.bind("<Button-1>",catalog)
but_shifr.bind("<Button-1>",shifr)
```

```
but_deshifr.bind("<Button-1>",deshifr)
but_instruction.bind("<Button-1>",instruction)
but_about.bind("<Button-1>",about)
ent_name.bind("<KeyRelease>",name)
root.protocol('WM_DELETE_WINDOW',away)
root.mainloop()
```