Для разработки интерфейса программного модуля для работы с растровым изображением на языке python, можно использовать следующие библиотеки:

Tkinter - для создания окон, меню и панели инструментов;

Pillow - для работы с изображениями (загрузка, сохранение, копирование, вставка, изменение);

Undo - для реализации отмены действия.

Пример кода для создания интерфейса программного модуля:

import tkinter as tk

from tkinter import filedialog

from tkinter import messagebox

from PIL import Image, ImageTk

from undo import Undo

class ImageProcessor:

def \_\_init\_\_(self, master):

self.master = master

self.master.title("Image Processor")

# создаем меню

menubar = tk.Menu(self.master)

filemenu = tk.Menu(menubar, tearoff=0)

filemenu.add\_command(label="Open", command=self.open\_image)

filemenu.add\_command(label="Save", command=self.save\_image)

menubar.add\_cascade(label="File", menu=filemenu)

self.master.config(menu=menubar)

# создаем строку состояния

self.statusbar = tk.Label(self.master, text="Ready", bd=1, relief=tk.SUNKEN, anchor=tk.W)

self.statusbar.pack(side=tk.BOTTOM, fill=tk.X)

# создаем панель инструментов

toolbar = tk.Frame(self.master)

open\_img = tk.PhotoImage(file="open.gif")

open\_btn = tk.Button(toolbar, image=open\_img, command=self.open\_image)

open\_btn.pack(side=tk.LEFT, padx=2, pady=2)

save\_img = tk.PhotoImage(file="save.gif")

save\_btn = tk.Button(toolbar, image=save\_img, command=self.save\_image)

save\_btn.pack(side=tk.LEFT, padx=2, pady=2)

undo\_img = tk.PhotoImage(file="undo.gif")

undo\_btn = tk.Button(toolbar, image=undo\_img, command=self.undo)

undo\_btn.pack(side=tk.LEFT, padx=2, pady=2)

toolbar.pack(side=tk.TOP, fill=tk.X)

# создаем окна для изображений

self.orig\_image = None

self.proc\_image = None

self.orig\_canvas = tk.Canvas(self.master, width=400, height=400)

self.orig\_canvas.pack(side=tk.LEFT, padx=10, pady=10)

self.proc\_canvas = tk.Canvas(self.master, width=400, height=400)

self.proc\_canvas.pack(side=tk.LEFT, padx=10, pady=10)

# создаем объект для отмены действия

self.undo\_stack = Undo()

def open\_image(self):

filetypes = (("Bitmap files", "\*.bmp"), ("JPEG files", "\*.jpeg"))

filename = filedialog.askopenfilename(title="Open Image", filetypes=filetypes)

if filename:

try:

self.orig\_image = Image.open(filename)

self.show\_image(self.orig\_canvas, self.orig\_image)

except:

messagebox.showerror("Error", "Cannot open image")

def save\_image(self):

if self.proc\_image:

filetypes = (("Bitmap files", "\*.bmp"), ("JPEG files", "\*.jpeg"))

filename = filedialog.asksaveasfilename(title="Save Image", filetypes=filetypes)

if filename:

try:

self.proc\_image.save

self.proc\_image.save(filename)

except:

messagebox.showerror("Error", "Cannot save image")

else:

messagebox.showinfo("Info", "No processed image")

def show\_image(self, canvas, image):

img = ImageTk.PhotoImage(image)

canvas.image = img

canvas.create\_image(0, 0, anchor=tk.NW, image=img)

def copy\_image(self):

if self.orig\_image:

self.clipboard = self.orig\_image.copy()

def copy\_fragment(self, x1, y1, x2, y2):

if self.orig\_image:

self.clipboard = self.orig\_image.crop((x1, y1, x2, y2))

def paste\_image(self):

if self.clipboard:

self.proc\_image = self.orig\_image.copy()

self.proc\_image.paste(self.clipboard, (0, 0))

self.show\_image(self.proc\_canvas, self.proc\_image)

self.undo\_stack.push(self.proc\_image)

def undo(self):

image = self.undo\_stack.pop()

if image:

self.proc\_image = image

self.show\_image(self.proc\_canvas, self.proc\_image)

root = tk.Tk()

app = ImageProcessor(root)

root.mainloop()

В этом примере мы создаем класс ImageProcessor, который содержит все необходимые элементы интерфейса и методы для работы с изображениями. Мы используем библиотеку Tkinter для создания окон, меню, строки состояния и панели инструментов. Для работы с изображениями мы используем библиотеку Pillow. Мы также используем модуль Undo для реализации отмены действия.

В методе \_\_init\_\_ мы создаем все необходимые элементы интерфейса и привязываем к ним соответствующие методы. Метод open\_image загружает изображение из файла, метод save\_image сохраняет обработанное изображение в файл. Методы copy\_image, copy\_fragment и paste\_image реализуют копирование и вставку изображений, а метод undo реализует отмену действия.

Мы также определяем метод show\_image, который отображает изображение на холсте и метод clipborad\_clear, который очищает буфер обмена.

Запустив этот код, мы получим окно с интерфейсом программного модуля для работы с растровым изображением.