import customtkinter as ctk

class User\_Turtle():

def \_\_init\_\_(self, port\_manager, turtle\_sim, text\_output):

self.port\_manager = port\_manager

self.turtle\_sim = turtle\_sim

self.text\_output = text\_output

self.compile\_mode =True

self.commands\_list = []

def find\_simulation\_scale(self):

def forward(self, number):

def right(self, number):

def left(self, number):

def down(self):

def up(self):

def message(self, message):

import customtkinter as ctk

from time import sleep

import user\_turtle as t

class Code\_Handler():

def \_\_init\_\_(self, port\_manager):

self.port\_manager = port\_manager

def handle\_code(self, code\_input, turtle, text\_output):

import customtkinter as ctk

import tkinter as tk

import idlelib.colorizer as ic

import idlelib.percolator as ip

from tkinter import scrolledtext

class Code\_Input():

def \_\_init\_\_(self, root, clear\_text):

self.code\_frame= ctk.CTkFrame(root, fg\_color="transparent")

self.textbox = scrolledtext.ScrolledText

#checks every second

cdg = ic.ColorDelegator()

#not supported by ctk.CtkTextbox so switched to tk.Text

ip.Percolator(self.textbox).insertfilter(cdg)

def get\_code(self):string

def set\_code(self, new):

def insert\_code(self, new):

def change\_textsize(self, size):

def clear(self):

#resize code elements to meet new window size requirements

def resize(self, new\_width, new\_height):

import customtkinter as ctk

class Connection\_Show():

def make\_connection\_layout(self, top\_bar\_frame): ctk.CTkFrame

def update\_states(self, usb\_connection, allow\_writing):

import customtkinter as ctk

import tkinter as tk

import glob

import re

class File\_Handler():

def \_\_init\_\_(self, get\_text, set\_text, insert\_text):

self.set\_text = set\_text

self.get\_text = get\_text

self.insert\_text = insert\_text

def load(self):

def save(self):

def get\_available\_inserts(self, shape\_type): filenames string[]

def load\_insert\_text(self, value, folder):

import tkinter as tk

import customtkinter as ctk

from tkhtmlview import HTMLScrolledText

from tkhtmlview import RenderHTML

class Info\_Page():

def \_\_init\_\_(self, top\_bar\_frame):

def display\_page(self):

import customtkinter as ctk

import tkinter as tk

import turtle\_simulation as ts

import code\_input

import top\_menu

import code\_handler

import port\_manager

import connection\_state

class Main():

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

# make window

root = ctk.CTk()

# specify window size

self.width, self.height=800,600

top\_frame = ctk.CTkFrame(root, fg\_color="#007D02", corner\_radius=0)

connection\_states = connection\_state.Connection\_Show()

connection\_layout = connection\_states.make\_connection\_layout(top\_frame)

self.port\_manager = port\_manager.port\_manager(connection\_states)

self.code\_handler = code\_handler.Code\_Handler(self.port\_manager)

self.code\_input = code\_input.Code\_Input

self.right\_frame= ctk.CTkFrame

#shows output errors on label

self.text\_output = ctk.CTkTextbox

self.top\_menu = top\_menu.Top\_Menu

self.turtle\_view = ts.Turtle\_Simulation(self.right\_frame)

self.button\_frame= ctk.CTkFrame(self.right\_frame, fg\_color="transparent")

self.run\_button = ctk.CTkButton

self.stop\_button = ctk.CTkButton

def on\_closing(self, root):

def button\_setup(self, frame, text, command, state): ctk.CTkButton

def change\_textsize(self, size):

def buttons\_layout(self, horizontal):

def clear\_text(self):

def run\_text(self):

def stop(self):

if \_\_name\_\_=="\_\_main\_\_":

Main()

import serial.tools.list\_ports

import serial

import time

import threading

class port\_manager():

def \_\_init\_\_(self, connection\_states):

self.connection\_states = connection\_states

self.setup =False

self.usb\_connection = False

self.allow\_writing = False

self.up =0.5

self.down = 0.3

self.port = None

self.port\_name =None

self.sent = 0

#number of commands the buffer can take at a time

self.buffer\_number = 10

#current values saved to the EEPROM

self.saved\_settings = None

def change\_port(self):

def set\_port(self, port\_name):

def get\_port\_names(self): return String, port\_names[]

def read\_port(self):

def close\_port(self):

def send\_command(self, command):

def get\_settings(self): self.saved\_settings:string

import tkinter as tk

import customtkinter as ctk

import re

import time

import math

from PIL import Image

class setup\_wizard():

def \_\_init\_\_(self, port\_manager):

self.circle\_steps = 4096

self.port\_manager = port\_manager

self.settings = {}

self.main\_button\_height = 0.8

self.paragraph\_font = ("Roboto", 14)

self.pop\_up\_size = 500

self.turtle\_image = ctk.CTkImage

self.diameter\_image = ctk.CTkImage

self.axle\_images = ctk.CTkImage[]

self.axle\_image\_first=True

def get\_settings(self, settings):

def make\_title(self, title\_text):

def make\_paragraph(self, label\_text):

def make\_button(self, frame, b\_text, b\_command): ctk.CTkButton

def setup\_wizard(self):

def check\_backlash\_start(self):

def check\_backlash\_main(self):

def backlash\_forward(self):

def backlash\_moved(self):

def end\_backlash(self):

def check\_wheel\_diameter(self):

def draw\_diameter\_button(self, first=False):

def draw\_for\_diameter(self):

def diameter\_length\_check(self):

def calculate\_wheel\_diameter(self, length):

def check\_axle\_length(self):

def set\_axle\_image(self):

def make\_axle\_length\_inputs(self):

def draw\_for\_axle(self):

def calculate\_axle\_length(self):

def validate\_input(self, length): bool

def save\_all(self):

import customtkinter as ctk

import tkinter as tk

import file\_handler as fh

import information\_page as ip

import setup\_wizard as sw

import sys

class Top\_Menu():

def \_\_init\_\_(self, root, port\_manager, top\_bar\_frame, connection\_frame, change\_textsize, set\_text, get\_text, insert\_text):

file\_handler = fh.File\_Handler(get\_text, set\_text, insert\_text)

self.port\_manager = port\_manager

self.up = 0.5 #stuck till can reliabily move less

self.down=0.3

setup\_wizard = sw.setup\_wizard(port\_manager)

fonts = [8, 10, 11, 12, 14, 16, 18, 20, 24, 28, 32, 36]

ip.Info\_Page(top\_bar\_frame)

#self.refresh\_button.pack(side=ctk.LEFT, padx=3)

#show which port is currently connected to

self.port\_label = ctk.CTkLabel

def make\_top\_menu\_button(self, top\_bar\_frame, text): return tk.Menubutton

def make\_connection\_dropdown(self):

def optionmenu\_callback(self, choice):

def refresh\_ports\_list(self):

def set\_pen\_view(self):

def down\_slider\_event(self, value):

def save(self):

import turtle

import customtkinter as ctk

import tkinter as tk

import re

import math

class Turtle\_Simulation():

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

self.angle = 90

self.size\_used=[0,0,0,0]

self.canvas = tk.Canvas(root, width=500, height=500)

self.canvas.pack(side=ctk.TOP)

self.screen = turtle.TurtleScreen

self.terry = turtle.RawTurtle

self.paused = False

self.scale = 1

self.reset\_button: ctk.CTkButton

def run\_code(self, code, output\_label):

def stop\_turtle(self):

def get\_paused(self): self.paused

def reset(self):

#resize turtle canvas

def resize(self, width, height):

def work\_out\_scale(self, lines):