Weihao Fu

CSD325 10.2 Assignment

Requirement 5:

A screenshot of a computer screen

AI-generated content may be incorrect.

Question 6:

1. A screenshot of a computer screen

   AI-generated content may be incorrect.
2. A screenshot of a computer

   AI-generated content may be incorrect.A screenshot of a computer

   AI-generated content may be incorrect.
3. A screenshot of a computer program

   AI-generated content may be incorrect.

Code:

import tkinter as tk

import tkinter.messagebox as msg

from tkinter import Menu

class Todo(tk.Tk):

def \_\_init\_\_(self, tasks=None):

super().\_\_init\_\_()

if not tasks:

self.tasks = []

else:

self.tasks = tasks

self.tasks\_canvas = tk.Canvas(self)

self.tasks\_frame = tk.Frame(self.tasks\_canvas)

self.text\_frame = tk.Frame(self)

self.scrollbar = tk.Scrollbar(self.tasks\_canvas, orient="vertical", command=self.tasks\_canvas.yview)

self.tasks\_canvas.configure(yscrollcommand=self.scrollbar.set)

self.title("Fu - To-Do App v2")

self.geometry("300x400")

#Exit File

menu\_bar = Menu(self)

self.config(menu=menu\_bar)

#File menu

file\_menu = Menu(menu\_bar, tearoff = 0)

menu\_bar.add\_cascade(label="File", menu = file\_menu)

file\_menu.add\_command(label="Exit", command= self.destroy)

self.task\_create = tk.Text(self.text\_frame, height=3, bg="white", fg="black")

self.tasks\_canvas.pack(side=tk.TOP, fill=tk.BOTH, expand=1)

self.scrollbar.pack(side=tk.RIGHT, fill=tk.Y)

self.canvas\_frame = self.tasks\_canvas.create\_window((0, 0), window=self.tasks\_frame, anchor="n")

self.task\_create.pack(side=tk.BOTTOM, fill=tk.X)

self.text\_frame.pack(side=tk.BOTTOM, fill=tk.X)

self.task\_create.focus\_set()

todo1 = tk.Label(self.tasks\_frame, text="--- Add Items Here ---", bg="orange2", fg="thistle4", pady=10)

todo1.bind("<Button-2>", self.remove\_task)

self.tasks.append(todo1)

for task in self.tasks:

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

self.bind("<Return>", self.add\_task)

self.bind("<Configure>", self.on\_frame\_configure)

self.bind\_all("<MouseWheel>", self.mouse\_scroll)

self.bind\_all("<Button-4>", self.mouse\_scroll)

self.bind\_all("<Button-5>", self.mouse\_scroll)

self.tasks\_canvas.bind("<Configure>", self.task\_width)

self.colour\_schemes = [{"bg": "LightSkyBlue1", "fg": "thistle4"}, {"bg": "OliveDrab2", "fg": "thistle4"}]

def add\_task(self, event=None):

task\_text = (f'{self.task\_create.get(1.0,tk.END).strip()}\n Right-Clicked for Deletion')

if len(task\_text) > 0:

new\_task = tk.Label(self.tasks\_frame, text=task\_text, pady=10)

self.set\_task\_colour(len(self.tasks), new\_task)

new\_task.bind("<Button-1>", self.remove\_task)

new\_task.pack(side=tk.TOP, fill=tk.X)

self.tasks.append(new\_task)

self.task\_create.delete(1.0, tk.END)

def remove\_task(self, event):

task = event.widget

if msg.askyesno("Really Delete?", "Delete " + task.cget("text") + "?"):

self.tasks.remove(event.widget)

event.widget.destroy()

self.recolour\_tasks()

def recolour\_tasks(self):

for index, task in enumerate(self.tasks):

self.set\_task\_colour(index, task)

def set\_task\_colour(self, position, task):

\_, task\_style\_choice = divmod(position, 2)

my\_scheme\_choice = self.colour\_schemes[task\_style\_choice]

task.configure(bg=my\_scheme\_choice["bg"])

task.configure(fg=my\_scheme\_choice["fg"])

def on\_frame\_configure(self, event=None):

self.tasks\_canvas.configure(scrollregion=self.tasks\_canvas.bbox("all"))

def task\_width(self, event):

canvas\_width = event.width

self.tasks\_canvas.itemconfig(self.canvas\_frame, width = canvas\_width)

def mouse\_scroll(self, event):

if event.delta:

self.tasks\_canvas.yview\_scroll(int(-1\*(event.delta/120)), "units")

else:

if event.num == 5:

move = 1

else:

move = -1

self.tasks\_canvas.yview\_scroll(move, "units")

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

todo = Todo()

todo.mainloop()