PD LAB

ASSIGNMENT - 2

Name: Raunak Thanawala

Registration Number: 231070051

Branch: Computer Engineering

Batch: 3

**Aim:-**

To create a contact application in python using tkinter and dictionaries.

**Theory:-**

Dictionary:

A dictionary is a collection which is ordered, changeable and does not allow duplicates. They are used to store data values in key:value pairs. To initialize a dictionary we use curly braces ({}).

Eg. dict = { “name”: “Raunak”, “age”: “19”,   
“Batch”: “Computer”}

Where dict[name] = Raunak, dict[age] = 19 and dict[Batch] = Computer

Tkinter:

Tkinter is Python's basic GUI library used for making cross-platform desktop applications.

We can use various widgets like buttons, labels, text fields, and more to build interactive user interfaces.

It offers three geometry managers (pack, grid, and place) for widget layout.

For Windows and MacOS:

tkinter is pre downloaded in the python installation. We can check if it exists with the command:

python -m tkinter

For Linux:

On debian based Linux OS we have to use the command:

sudo apt-get install python3-tk

**Code and Output:**

import tkinter as tk

from tkinter import messagebox

from tkinter import font as tkfont

x = 0

class ContactsApp:

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

self.root = root

self.root.title("Contacts App")

self.contacts = {}

self.is\_dark\_mode = False

*# Create and place widgets*

self.create\_widgets()

def create\_widgets(self):

*# Name Label and Entry*

tk.Label(self.root, text="Name:").grid(row=1, column=0, padx=10, pady=10)

self.name\_entry = tk.Entry(self.root)

self.name\_entry.grid(row=1, column=1, padx=5, pady=5)

*# Phone Label and Entry*

tk.Label(self.root, text="Phone:").grid(row=2, column=0, padx=10, pady=10)

self.phone\_entry = tk.Entry(self.root)

self.phone\_entry.grid(row=2, column=1, padx=5, pady=5)

*# Add Contact Button*

tk.Button(self.root, text="+", command=self.add\_contact,width=5,height=2).grid(row=1,column=2 , pady=10)

*# Contacts Listbox*

self.contacts\_listbox = tk.Listbox(self.root, width=50, height=1)

self.contacts\_listbox.grid(row=4, column=0, columnspan=4, padx=5, pady=5)

*# Delete Contact Button*

tk.Button(self.root, text="Delete Contact", command=self.delete\_contact).grid(row=5, columnspan=2, pady=5)

self.contact\_count\_label = tk.Label(self.root, text="Number of contacts: 0")

self.contact\_count\_label.grid(row=0, columnspan=2, pady=10)

self.toggle\_button = tk.Button(self.root, text="🌙", command=self.toggle\_mode)

self.toggle\_button.grid(row=0, column=3, pady=10)

def apply\_theme(self):

if self.is\_dark\_mode:

bg\_color = '#2e2e2e'

fg\_color = '#ffffff'

button\_bg = '#444444'

button\_fg = '#ffffff'

listbox\_bg = '#333333'

listbox\_fg = '#ffffff'

else:

bg\_color = '#f0f0f0'

fg\_color = '#000000'

button\_bg = '#e0e0e0'

button\_fg = '#000000'

listbox\_bg = '#ffffff'

listbox\_fg = '#000000'

self.root.configure(bg=bg\_color)

self.toggle\_button.config(bg=button\_bg, fg=button\_fg)

self.contacts\_listbox.config(bg=listbox\_bg, fg=listbox\_fg)

*# Update other widgets*

for widget in self.root.winfo\_children():

if isinstance(widget, tk.Label):

widget.config(bg=bg\_color, fg=fg\_color)

elif isinstance(widget, tk.Entry):

widget.config(bg=bg\_color, fg=fg\_color)

elif isinstance(widget, tk.Button):

widget.config(bg=button\_bg, fg=button\_fg)

def toggle\_mode(self):

self.is\_dark\_mode = not self.is\_dark\_mode

mode\_text = "☀️" if self.is\_dark\_mode else "🌙"

self.toggle\_button.config(text=mode\_text)

self.apply\_theme()

def add\_contact(self):

name = self.name\_entry.get().strip()

phone = self.phone\_entry.get().strip()

if name and phone:

self.contacts[name] = phone

self.update\_contacts\_listbox()

self.name\_entry.delete(0, tk.END)

self.phone\_entry.delete(0, tk.END)

else:

messagebox.showwarning("Input Error", "Both name and phone number are required.")

self.update\_contact\_count()

def update\_contacts\_listbox(self):

self.contacts\_listbox.delete(0, tk.END)

for name, phone in sorted(self.contacts.items()):

self.contacts\_listbox.insert(tk.END, f"{name}: {phone}")

*# Adjust the height of the Listbox based on the number of contacts*

num\_items = len(self.contacts)

max\_visible\_items = 10

new\_height = min(num\_items, max\_visible\_items)

self.contacts\_listbox.config(height=new\_height)

def update\_contact\_count(self):

num\_contacts = len(self.contacts)

self.contact\_count\_label.config(text=f"Number of contacts: {num\_contacts}")

def delete\_contact(self):

selected\_index = self.contacts\_listbox.curselection()

if selected\_index:

selected\_contact = self.contacts\_listbox.get(selected\_index[0])

name = selected\_contact.split(':')[0].strip()

del self.contacts[name]

self.update\_contacts\_listbox()

else:

messagebox.showwarning("Selection Error", "Please select a contact to delete.")

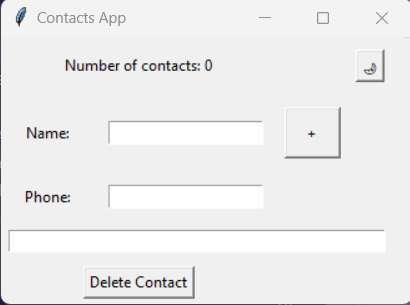
self.update\_contact\_count()

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

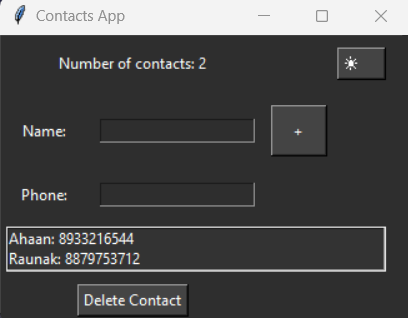
root = tk.Tk()

app = ContactsApp(root)

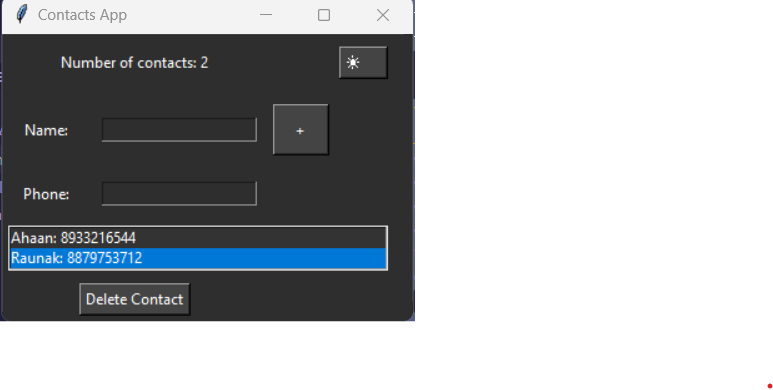
root.mainloop()



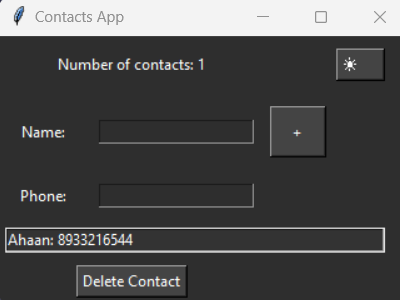
Basic Output



Output with 2 Contacts saved and Dark mode enabled



Selecting a contact so we can delete it



Contacts after Deleting the Contact

**Conclusion:**

Thus we have written a program to store contacts in a dictionary and then have made a Contacts application using tkinter.