1. **What is Tkinter and how is it related to the Tk GUI toolkit?**  
   Tkinter is the standard GUI (Graphical User Interface) library for Python. It is a Python binding to the **Tk GUI toolkit**, which is written in C. Tkinter provides a simple way to create GUI applications.
2. **Why is Tkinter considered a cross-platform GUI toolkit?**  
   Because it works on **Windows, macOS, and Linux** without requiring changes to the code. This is possible because it uses the underlying **Tk** framework, which is also cross-platform.
3. **What are the advantages of using Tkinter for Python GUI development?**
   * Comes bundled with Python (no separate installation on most systems)
   * Lightweight and easy to use
   * Active community and documentation
   * Cross-platform compatibility
   * Quick prototyping for desktop apps
4. **How do you check if Tkinter is installed in your system?**  
   Run the following in a Python shell:

ssimport tkinter

print(tkinter.TkVersion)

1. **What does the tk.Tk() function do in a Tkinter application?**  
   It creates the **main application window** and serves as the **root container** for all other widgets.
2. **What is the purpose of root.mainloop() in a Tkinter program?**  
   It starts the **Tkinter event loop**, which waits for user interactions (like button clicks) and updates the GUI accordingly.
3. **How do you set the window size in Tkinter?**

root.geometry("400x300") # Width x Height

1. **How do you change the title of a Tkinter window?**

root.title("My App Title")

1. **How do you add an icon to a Tkinter window?**

root.iconbitmap("path/to/icon.ico") # Works best with .ico on Windows

**SECTION 2: Widgets – Label, Entry, Text, Button**

1. **What is a widget in Tkinter? Name five commonly used widgets.**  
   A **widget** is a GUI element like a button or text box. Common widgets:
   * Label
   * Button
   * Entry
   * Text
   * Frame
2. **How do you create and display a Label widget in a Tkinter window?**

label = tk.Label(root, text="Hello, World!")

label.pack()

1. **How do you update the text of a Label dynamically?**

label.config(text="New Text")

1. **What is the use of the Entry widget and how do you retrieve its value?**  
   It allows **single-line text input**:

python

CopyEdit

entry = tk.Entry(root)

entry.pack()

value = entry.get()

1. **How do you validate an email address entered into an Entry widget using regex?**

python

CopyEdit

import re

email = entry.get()

if re.match(r"[^@]+@[^@]+\.[^@]+", email):

print("Valid email")

else:

print("Invalid email")

1. **What is the purpose of the Text widget and how is it different from Entry?**  
   Text allows **multi-line input**, while Entry supports only **single-line input**.
2. **How do you insert and retrieve text from a Text widget?**

python

CopyEdit

text\_widget.insert("1.0", "Hello")

content = text\_widget.get("1.0", "end-1c") # 'end-1c' avoids the extra newline

1. **Which method is used to delete all text from an Entry or Text widget?**
   * Entry: entry.delete(0, tk.END)
   * Text: text\_widget.delete("1.0", tk.END)
2. **How do you configure a Button to execute a function on click?**

def say\_hello():

print("Hello")

btn = tk.Button(root, text="Click Me", command=say\_hello)

btn.pack()

1. **What is the difference between .get(), .insert(), and .delete() methods?**

* .get(): Retrieves the content
* .insert(): Adds content at a specific position
* .delete(): Removes content from a widget

**SECTION 3: Layouts – pack(), grid(), place(), Frames**

1. **What are the three layout (geometry) managers in Tkinter?**
   * pack()
   * grid()
   * place()
2. **Explain how the pack() geometry manager works.**  
   It automatically places widgets **relative to each other** (top, bottom, left, right), **stacking** them in the order they are packed.
3. **What does side="left" or fill="x" do in the pack() method?**
   * side="left": Packs the widget to the **left side** of the container
   * fill="x": Makes the widget expand **horizontally**
4. **When should you prefer grid() over pack()?**  
   Use grid() when you need a **table-like (row-column)** layout. It's more flexible than pack() for complex designs.
5. **What are row, column, and sticky options used for in grid()?**
   * row/column: Position in the grid
   * sticky: Aligns widget (e.g., sticky="nsew" sticks it to all sides)
6. **What is the role of the place() method and when should it be used?**  
   place() allows **absolute positioning** of widgets using **x/y coordinates** or relative positioning. Use it when you need **precise control**.
7. **How can you center a widget using place()?**

python

CopyEdit

widget.place(relx=0.5, rely=0.5, anchor='center')

1. **What is a Frame in Tkinter and why is it useful?**  
   A Frame is a container widget used to **group other widgets** together. It helps in **organizing layout**, especially in complex GUIs.
2. **How do you add widgets inside a Frame?**

frame = tk.Frame(root)

frame.pack()

btn = tk.Button(frame, text="Inside Frame")

btn.pack()

1. **Can pack() and grid() be used together in the same container?**  
   **No**. You **cannot mix pack() and grid() in the same parent container** (like root or a Frame). Doing so raises a TclError.