STEP 1: Decide What the App Needs to Do

Before writing code, we think like a **program designer**.

Our To-Do app should:

1. Store tasks
2. Run continuously until the user exits
3. Show a menu
4. Take user input
5. Perform actions based on that input

This tells us we need:

- A **list** → to store tasks
- A **loop** → to keep the app running
- **conditionals** → to handle user choices

STEP 2: Create Storage for Tasks

```
tasks = []
```

Why a list?

- Tasks are multiple
- Order matters
- We need to add & remove items

STEP 3: Keep the App Running (Infinite Loop)

```
while True:
```

Why `while True`?

- The app should not stop after one action
- It should run **until the user chooses Exit**

STEP 4: Show the Menu

```
print("\n--- TO DO APP ---")
print("1. Add Task")
print("2. View Tasks")
print("3. Delete Task")
print("4. Exit")
```

Why menu?

- Users need guidance
- Numbers are easier than text commands

`\n` adds a blank line for better readability.

STEP 5: Take User Choice

```
choice = input("Enter your choice (1-4): ")
```

Important teaching point:

- `input()` **always returns a string**
- That's why we compare with `"1"`, `"2"`, not numbers

STEP 6: Handle Choice 1 → Add Task

```
if choice == "1":
    task = input("Enter a new task: ")
    tasks.append(task)
    print("Task added successfully!")
```

What's happening?

1. Ask user for task
2. Store it in `task`
3. Add it to the list using `append()`

```
tasks = ["Study Python"]
```

STEP 7: Handle Choice 2 → View Tasks

```
elif choice == "2":
    if len(tasks) == 0:
        print("No tasks available.")
```

Why check length?

- Looping over an empty list looks bad
- Always validate before displaying

```
else:
    print("\nYour Tasks:")
    for i in range(len(tasks)):
        print(f"{i + 1}. {tasks[i]}")
```

Why `i + 1`?

- Lists start at `0`
- Humans start counting from `1`

STEP 8: Handle Choice 3 → Delete Task

```
elif choice == "3":
```

First check if tasks exist

```
if len(tasks) == 0:
    print("No tasks to delete.")
```

Show tasks again

```
else:
    for i in range(len(tasks)):
        print(f"{i + 1}. {tasks[i]}")
```

Users must **see task numbers** before deleting

Take task number to delete

```
delete_task = int(input("Enter task number to delete: "))
```

This is the **only place** we convert input to `int`
because we need numeric comparison.

Validate the number

```
if delete_task > 0 and delete_task <= len(tasks):
```

Why this check?

- Prevents crashes
- Avoids invalid index errors

Remove the task

```
removed = tasks.pop(delete_task - 1)
print(f"Removed task: {removed}")
```

Why `-1`?

- User gives number starting from `1`
- List index starts from `0`

STEP 9: Handle Choice 4 → Exit App

```
elif choice == "4":
    print("Exiting To-Do App. Goodbye!")
    break
```

Why `break`?

- Break exits the `while True` loop
- App stops cleanly

STEP 10: Handle Invalid Input

```
else:
    print("Invalid choice. Please select 1-4.")
```

This prevents:

- App crashing
- User confusion

```
Final Mental Model for Understanding only

START
 ↓
Show Menu
 ↓
Take Input
 ↓
Check Choice
 ↓
Perform Action
 ↓
```

```
Repeat
```

Python To-Do App