# Name: Muhammad Ashir

**Roll No: 053**

# Explanation of To-Do List Program (OOP in Python)

This document explains the Python program for a To-Do List that was written using Object-Oriented Programming (OOP). The explanations are kept simple, human-friendly, and beginner-oriented so that anyone can understand the logic.

## 1. Defining the Class (ToDoList)

Here is the first part of the code:

*class ToDoList:  
def \_\_init\_\_(self):  
self.tasks = []  
  
def add\_task(self, task):  
self.tasks.append(task)  
  
def show\_tasks(self):  
if not self.tasks:  
print("No tasks yet!")  
else:  
print("\nYour Tasks:")  
for i, t in enumerate(self.tasks, 1):  
print(i, "-", t)  
  
def remove\_task(self, number):  
if 0 < number <= len(self.tasks):  
print("Removed:", self.tasks.pop(number - 1))  
else:  
print("Invalid number!")*

## Explanation:

- A class named `ToDoList` is created. A class is like a blueprint to organize related code together.  
- Inside the class:  
• `\_\_init\_\_`: This is the constructor. It runs automatically when the object is created. Here, it creates an empty list called `tasks` where we will store all the to-do items.  
• `add\_task`: This method takes a new task and adds it to the list.  
• `show\_tasks`: This method displays all tasks. If the list is empty, it says 'No tasks yet!'. Otherwise, it prints tasks with numbers.  
• `remove\_task`: This removes a task if the given number is valid. If not, it shows 'Invalid number!'.  
This class helps us keep all to-do related features (add, view, remove) neatly inside one place.

## 2. Running the To-Do List Program

Here is the second part of the code:

todo = ToDoList()  
  
while True:  
print("\n--- TO-DO LIST ---")  
print("1. Add Task")  
print("2. View Tasks")  
print("3. Remove Task")  
print("4. Exit")  
  
choice = input("Enter choice: ")  
  
if choice == '1':  
todo.add\_task(input("Enter task: "))  
elif choice == '2':  
todo.show\_tasks()  
elif choice == '3':  
num = int(input("Enter task number: "))  
todo.remove\_task(num)  
elif choice == '4':  
print("Goodbye!")  
break

else:  
print("Invalid choice!")

## Explanation:

- First, we create an object of the class: `todo = ToDoList()`. This means we now have a working To-Do List.  
- A `while True:` loop is used to keep the program running until the user chooses to exit.  
- The program shows a menu:  
1. Add Task → asks the user to type a task and saves it.  
2. View Tasks → shows all tasks stored so far.  
3. Remove Task → asks which task number to delete.  
4. Exit → ends the program.  
- If the user types something wrong, it shows 'Invalid choice!'.  
  
This loop makes the program interactive, allowing the user to keep adding or removing tasks until they choose to quit.

## 3. Why This Program is Useful:

This program is a simple example of how Object-Oriented Programming (OOP) makes our code clean and organized. Instead of writing everything in one place, we put related actions (add, view, remove tasks) inside a class. This makes it easier to understand, reuse, and expand later.  
  
In real life, to-do list applications are used to stay productive and remember important tasks. Even though this version runs in the console, the same logic can be used to build bigger apps with graphics or web versions.

## Output:

A screenshot of a computer

AI-generated content may be incorrect.