Julekha Khatoon

100937821

Assignment 2

# Part 1: Identify the Problem

## Problem 1: File Size Checker for Directory

**Description:** The presence of zero-sized or empty files is a common problem when handling files on a computer or server. These files may collect over time because of failed downloads, incomplete file transfers, or errors during file production. It is critical to discover and maintain these empty files on a regular basis since they consume disk space and may cause difficulty when organizing or backing up data. For example, an automated system may download files, but if the file is empty, it has no use and can be safely removed. Users can simply wipe away superfluous files by writing a Python application that checks for zero-sized files in a specified directory.

**Why This Is Useful:** This approach is useful for anyone who manages a big number of files, including developers, system administrators, and even individual users who organize their file directories. It will enable them to swiftly identify files that are not producing useful data and take appropriate steps, such as deletion or further examination. This can assist in freeing up storage space and ensure that useless files do not take up valuable disk space.

**Difficulty Level:** This task is reasonably basic; the main challenge is dealing with the file system, which necessitates the usage of libraries such as os for file manipulation. The key learning objectives will be to understand file paths, check file sizes, and handle edge circumstances (such as folders that do not exist or are empty). Overall, the coding solution will make use of basic Python control structures such as loops and conditionals, as well as the built-in os library.  
  
**Libraries Are Expected to Use:**  
**os** Library: To communicate with the file system, obtain file paths, and determine file size.

## Problem 2: To-Do List with Deadline Tracking

**Description:** To stay organized and remember chores and due dates, many people utilize to-do lists. However, without a method to track deadlines and prioritize work, managing them can become hard. A Python program that allows users to input tasks, specify deadlines, and then sort or filter tasks by deadline or priority can be useful. The application will accept user input for each task (name, deadline date, priority), save the tasks, and show them in chronological order based on their deadlines. It will also allow the user to mark completed tasks and delete them.

**Why This Is Useful**: This is a very practical and beneficial problem-solving exercise for anyone who struggles with task management. A to-do list software with deadline monitoring can help students, project managers, and even individuals manage personal activities. Python's built-in libraries can help you manage and arrange jobs more efficiently. This will not only increase productivity but will also allow you to practice working with data structures such as lists and dictionaries, as well as sorting algorithms.

**Difficulty Level:** This problem is appropriate for a beginner-level project because it requires managing data entries and performing operations on them. The difficulty is to save, sort, and filter tasks, as well as manage user inputs and ensure data persistence (e.g., saving tasks even after the software closes). Overall, the problem is simple, but it takes knowledge of lists, dictionaries, and how to handle user input in Python.  
  
**Libraries Are Expected to Use:**  
**datetime** library: Use dates and deadlines to categorize jobs.

Built-in Python functions: To process user input and manage lists or dictionaries.