

AI ASSISTED CODING

LAB EXAM-2

NAME : R.SURYANARAYANA

HT.no : 2403A52038

BATCH : 03

+

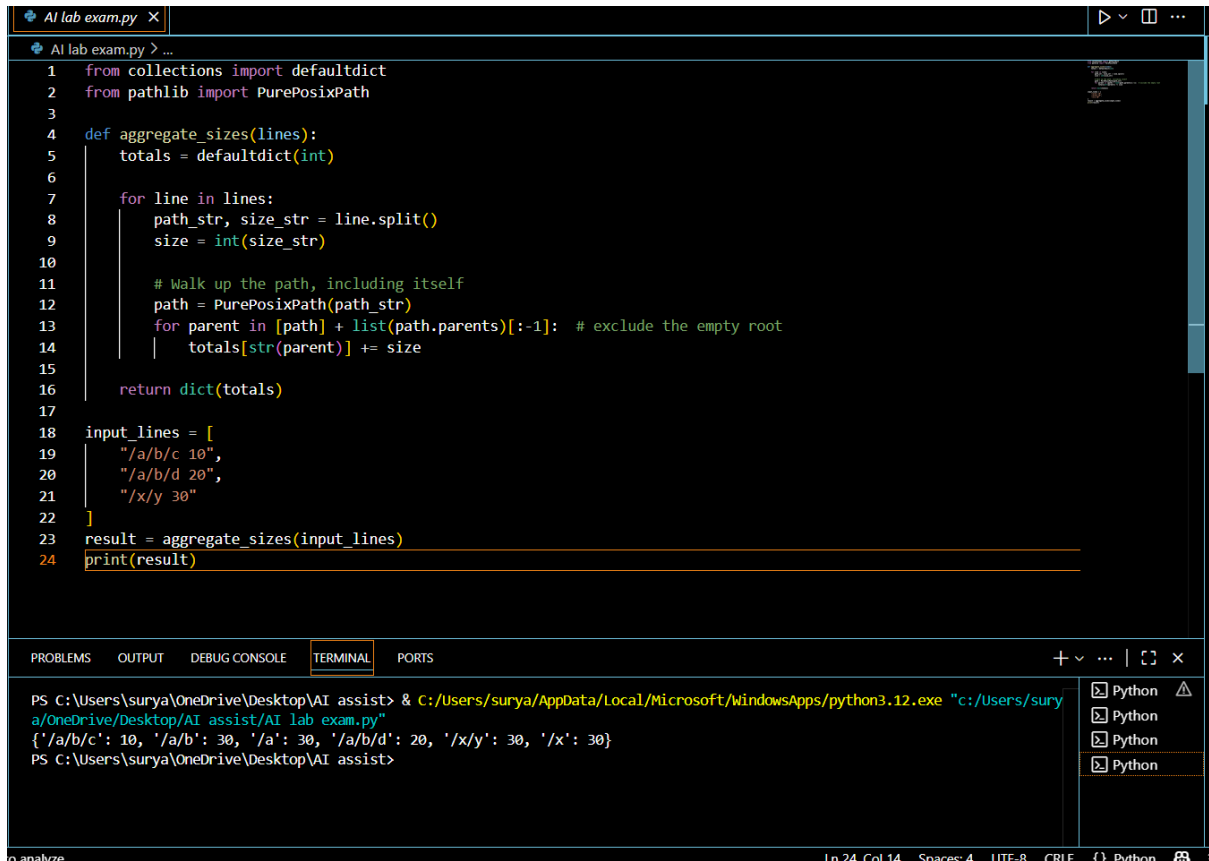
Q1. Compute directory sizes from listing

- **Task :** Aggregate sizes from a flat listing of `path size` lines so that each parent includes all descendants.

Prompt:

Explain in simple words how to find total sizes of folders when we only have a list like "/a 10", "/a/b 5", "/a/c 7". Describe how to add each item's size to its folder and all its parent folders so every parent shows the total size of everything inside it. No code—just the idea.

CODE :



```
1 from collections import defaultdict
2 from pathlib import PurePosixPath
3
4 def aggregate_sizes(lines):
5     totals = defaultdict(int)
6
7     for line in lines:
8         path_str, size_str = line.split()
9         size = int(size_str)
10
11         # Walk up the path, including itself
12         path = PurePosixPath(path_str)
13         for parent in [path] + list(path.parents)[:1]: # exclude the empty root
14             totals[str(parent)] += size
15
16     return dict(totals)
17
18 input_lines = [
19     "/a/b/c 10",
20     "/a/b/d 20",
21     "/x/y 30"
22 ]
23 result = aggregate_sizes(input_lines)
24 print(result)
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS C:\Users\surya\OneDrive\Desktop\AI assist> & C:/Users/surya/AppData/Local/Microsoft/WindowsApps/python3.12.exe "c:/Users/surya/OneDrive/Desktop/AI assist/AI lab exam.py"
{'/a/b/c': 10, '/a/b/d': 20, '/a': 30, '/a/b/d': 20, '/x/y': 30, '/x': 30}
PS C:\Users\surya\OneDrive\Desktop\AI assist>
```

Observation:

The code reads each path size line, adds the size to that path and all its parent folders, and finally returns a dictionary of total sizes. This makes every parent directory show the sum of its own size plus all of its children.
Example: `['/a 10', '/a/b 5', '/a/c 7']` → `{'/a': 22, '/a/b': 5, '/a/c': 7}`.

Q2. Compute simple sentiment score

Task : Compute total sentiment using: good=+1, great=+2, bad=-1.

Prompt : Write Python code that reads a sentence and calculates a sentiment score.

Use this lexicon: good = +1, great = +2, bad = -1.

Tokenize by spaces, remove basic punctuation, and return the total score as an integer.

CODE :



```
lab8.4.py x
lab8.4.py > ...
1 import re
2
3 def sentiment_score(text: str) -> int:
4     # simple lowercase & remove punctuation
5     clean = re.sub(r"^[^\w\s]", "", text.lower())
6     tokens = clean.split()
7
8     lexicon = {"good": 1, "great": 2, "bad": -1}
9     return sum(lexicon.get(token, 0) for token in tokens)
10
11 score = sentiment_score("This is a great and good day.")
12 print(f"Sentiment Score: {score}")
```

PROBLEMS OUTPUT DEBUG CONSOLE **TERMINAL** PORTS

a/OneDrive/Desktop/AI assist/lab8.4.py
Sentiment Score: 3
PS C:\Users\surya\OneDrive\Desktop\AI assist>

Python Python Python Python

Observation:

The program cleans the input text by removing punctuation and lowering case, splits it into words, and adds scores using the lexicon (good = +1, great = +2, bad = -1). It returns the total sentiment as an integer.