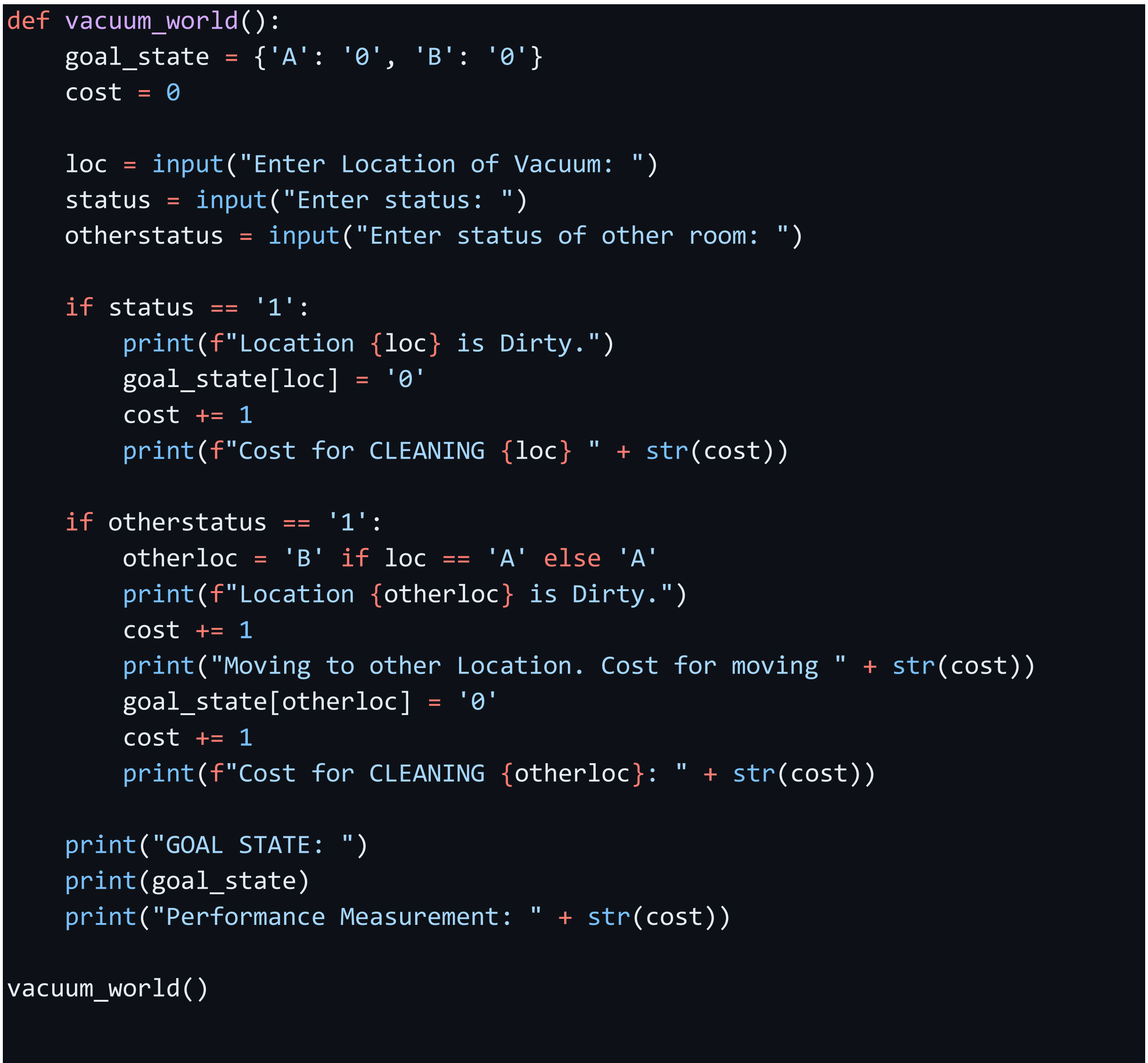
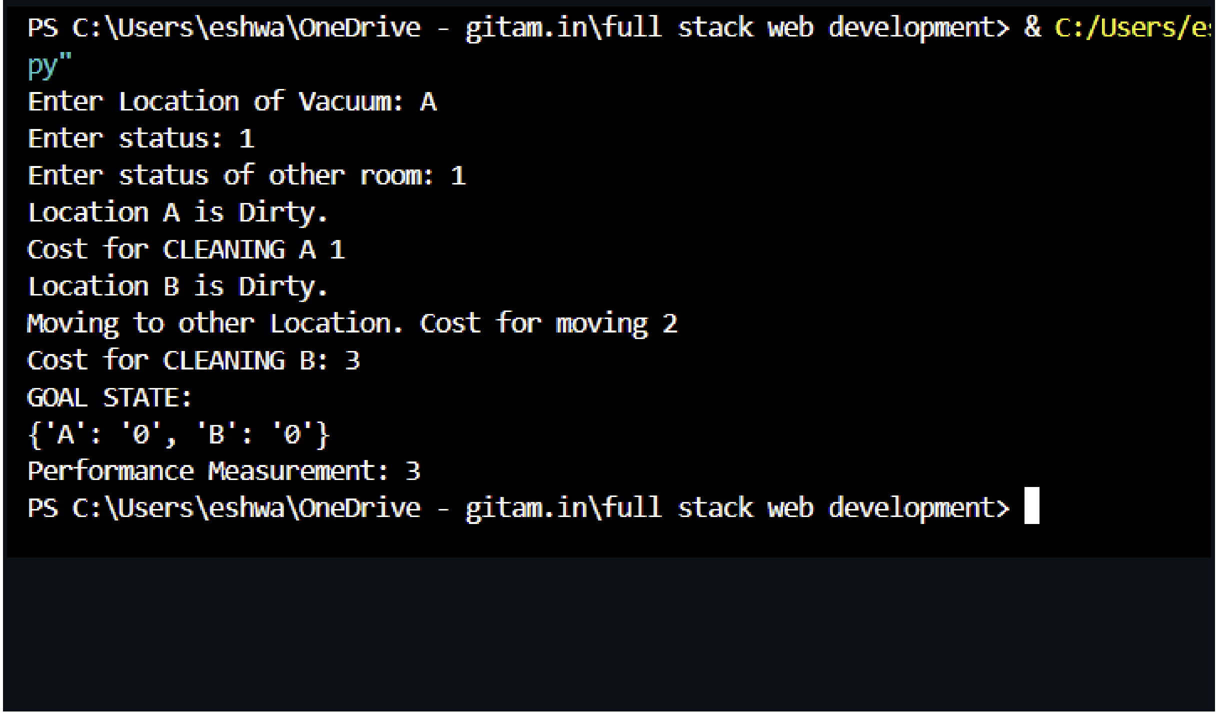
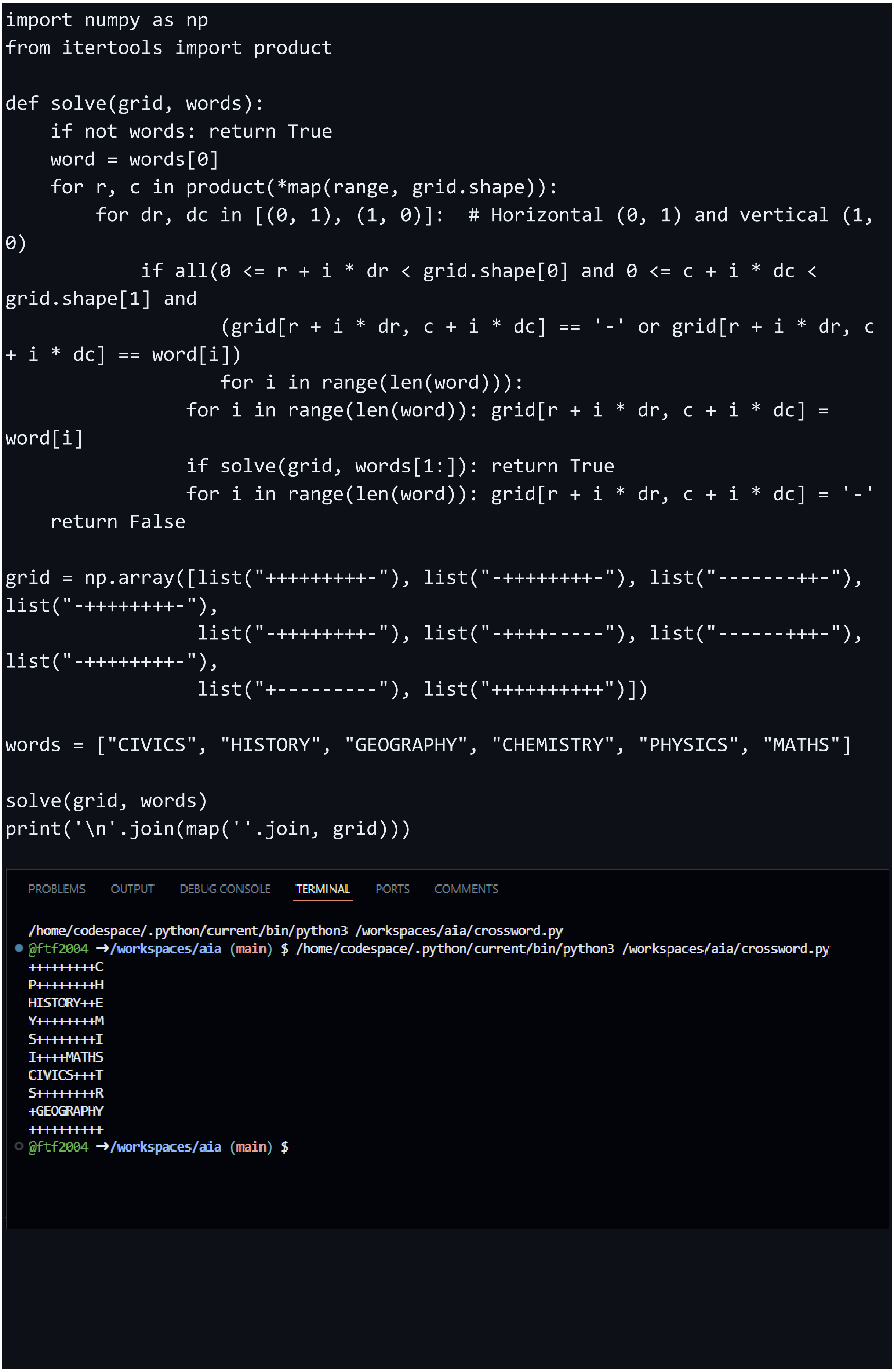


```python
def vacuum_world():
    goal_state = {'A': '0', 'B': '0'}
    cost = 0

    loc = input("Enter Location of Vacuum: ")
    status = input("Enter status: ")
    otherstatus = input("Enter status of other room: ")

    if status == '1':
        print(f"Location {loc} is Dirty.")
        goal_state[loc] = '0'
        cost += 1
        print(f"Cost for CLEANING {loc} " + str(cost))

    if otherstatus == '1':
        otherloc = 'B' if loc == 'A' else 'A'
        print(f"Location {otherloc} is Dirty.")
        cost += 1
        print("Moving to other Location. Cost for moving " + str(cost))
        goal_state[otherloc] = '0'
        cost += 1
        print(f"Cost for CLEANING {otherloc}: " + str(cost))

    print("GOAL STATE: ")
    print(goal_state)
    print("Performance Measurement: " + str(cost))

vacuum_world()
```



```
PS C:\Users\eshwa\OneDrive - gitam.in\full stack web development> & C:/Users/e
py"
Enter Location of Vacuum: A
Enter status: 1
Enter status of other room: 1
Location A is Dirty.
Cost for CLEANING A 1
Location B is Dirty.
Moving to other Location. Cost for moving 2
Cost for CLEANING B: 3
GOAL STATE:
{'A': '0', 'B': '0'}
Performance Measurement: 3
PS C:\Users\eshwa\OneDrive - gitam.in\full stack web development>
```



```
import numpy as np
from itertools import product

def solve(grid, words):
    if not words: return True
    word = words[0]
    for r, c in product(*map(range, grid.shape)):
        for dr, dc in [(0, 1), (1, 0)]:  # Horizontal (0, 1) and vertical (1,
0)
            if all(0 <= r + i * dr < grid.shape[0] and 0 <= c + i * dc <
grid.shape[1] and
                   (grid[r + i * dr, c + i * dc] == '-' or grid[r + i * dr, c
+ i * dc] == word[i])
                   for i in range(len(word))):
                for i in range(len(word)): grid[r + i * dr, c + i * dc] =
word[i]
                if solve(grid, words[1:]): return True
                for i in range(len(word)): grid[r + i * dr, c + i * dc] = '-'
    return False

grid = np.array([list("+++++++++-"), list("-++++++++-"), list("-------++-"),
list("-++++++++-"),
                 list("-++++++++-"), list("-++++-----"), list("------+++-"),
list("-++++++++-"),
                 list("+---------"), list("++++++++++")])

words = ["CIVICS", "HISTORY", "GEOGRAPHY", "CHEMISTRY", "PHYSICS", "MATHS"]

solve(grid, words)
print('\n'.join(map(''.join, grid)))
```

```
/home/codespace/.python/current/bin/python3 /workspaces/aia/crossword.py
@ftf2004 ➜/workspaces/aia (main) $ /home/codespace/.python/current/bin/python3 /workspaces/aia/crossword.py
+++++++++C
P++++++++H
HISTORY++E
Y++++++++M
S++++++++I
I++++MATHS
CIVICS+++T
S++++++++R
+GEOGRAPHY
++++++++++
@ftf2004 ➜/workspaces/aia (main) $
```