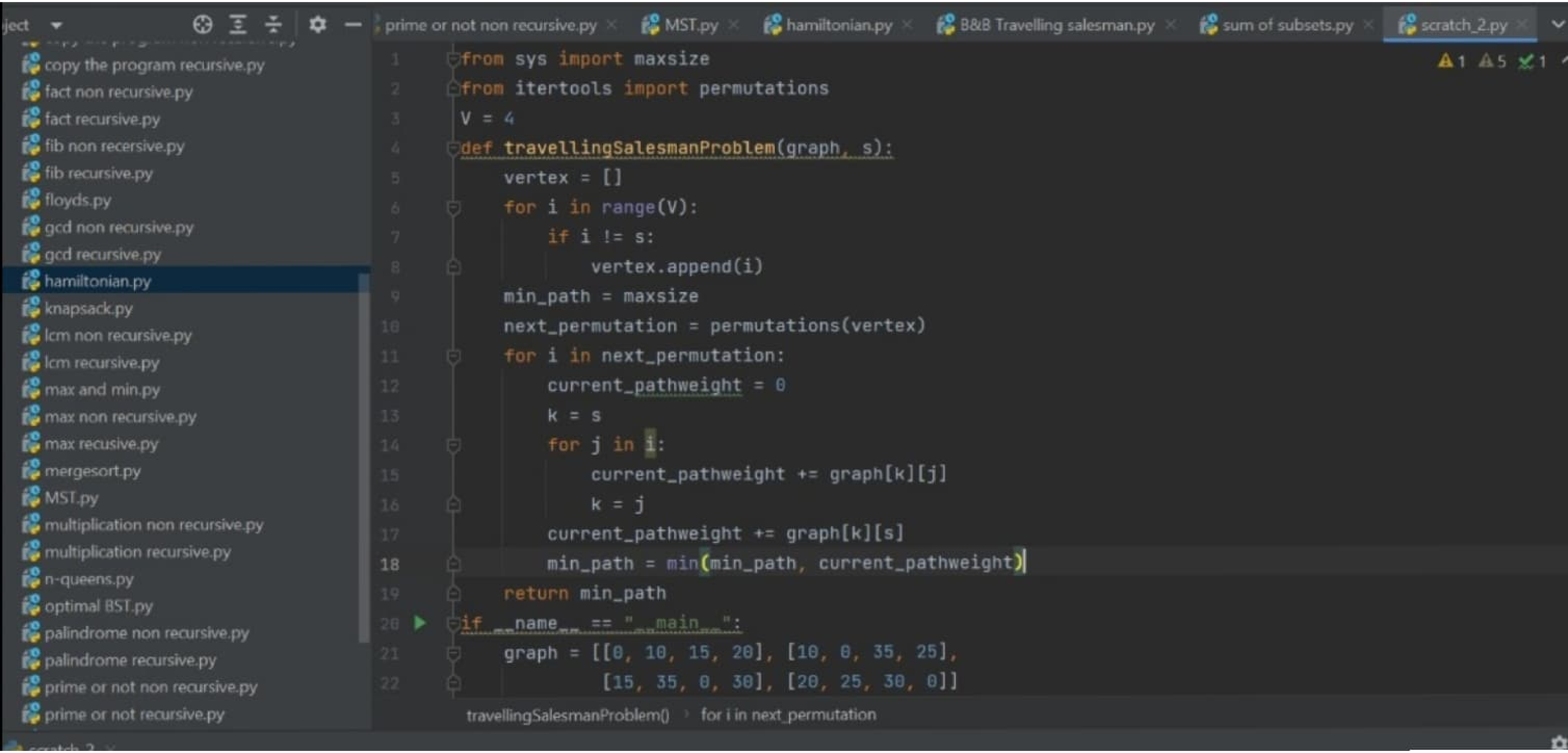
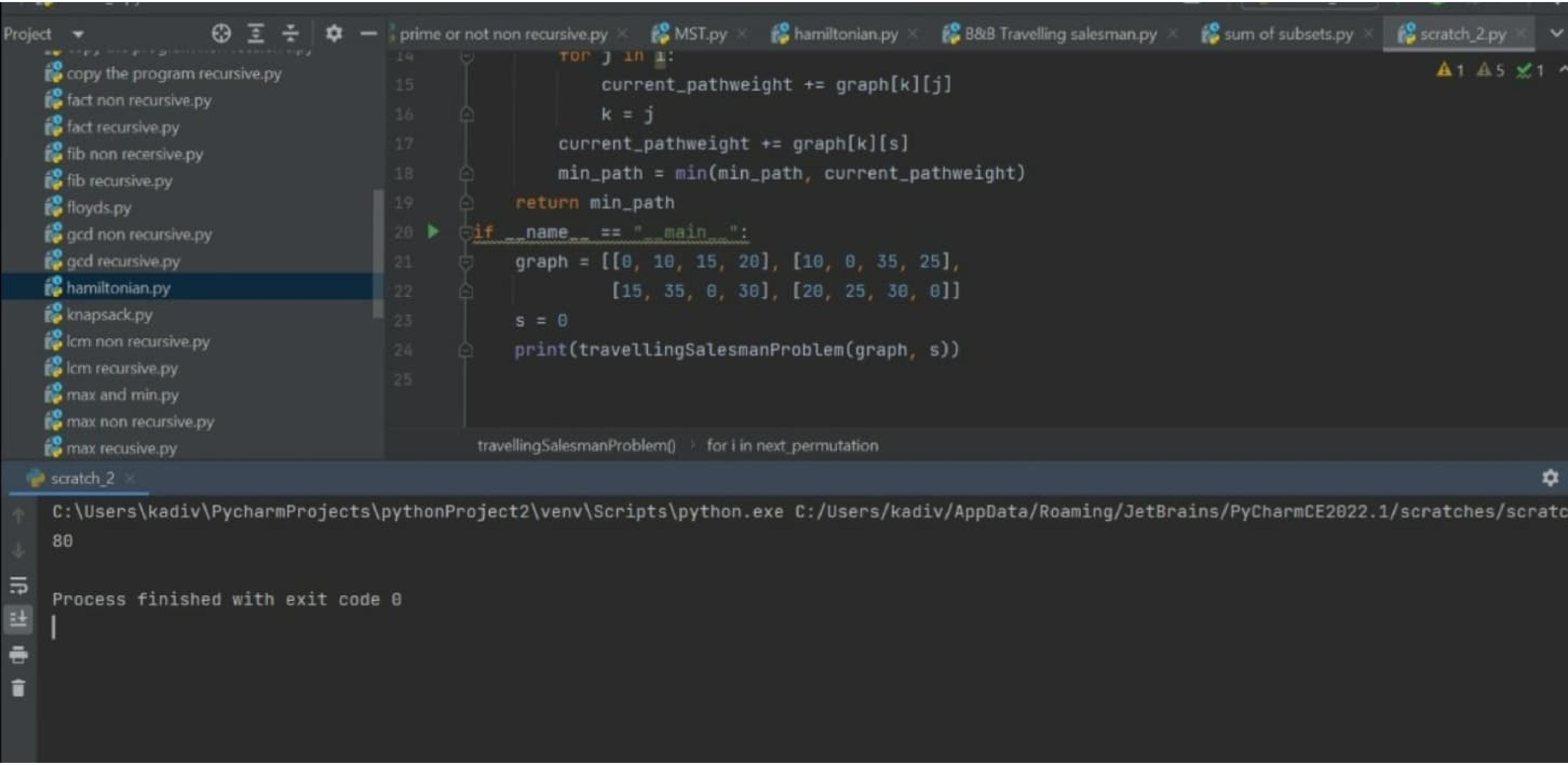

ject
prime or not non recursive.py
MST.py
hamiltonian.py
B&B Travelling salesman.py
sum of subsets.py
scratch_2.py
copy the program recursive.py
fact non recursive.py
fact recursive.py
fib non recersive.py
fib recursive.py
floyds.py
gcd non recursive.py
gcd recursive.py
hamiltonian.py
knapsack.py
lcm non recursive.py
lcm recursive.py
max and min.py
max non recursive.py
max recusive.py
mergesort.py
MST.py
multiplication non recursive.py
multiplication recursive.py
n-queens.py
optimal BST.py
palindrome non recursive.py
palindrome recursive.py
prime or not non recursive.py
prime or not recursive.py
⚠1 ⚠5 ✓1
travellingSalesmanProblem() › for i in next_permutation

```python
from sys import maxsize
from itertools import permutations
V = 4
def travellingSalesmanProblem(graph, s):
    vertex = []
    for i in range(V):
        if i != s:
            vertex.append(i)
    min_path = maxsize
    next_permutation = permutations(vertex)
    for i in next_permutation:
        current_pathweight = 0
        k = s
        for j in i:
            current_pathweight += graph[k][j]
            k = j
        current_pathweight += graph[k][s]
        min_path = min(min_path, current_pathweight)
    return min_path
if __name__ == "__main__":
    graph = [[0, 10, 15, 20], [10, 0, 35, 25],
             [15, 35, 0, 30], [20, 25, 30, 0]]
```


Project
prime or not non recursive.py
MST.py
hamiltonian.py
B&B Travelling salesman.py
sum of subsets.py
scratch_2.py
copy the program recursive.py
fact non recursive.py
fact recursive.py
fib non recersive.py
fib recursive.py
floyds.py
gcd non recursive.py
gcd recursive.py
hamiltonian.py
knapsack.py
lcm non recursive.py
lcm recursive.py
max and min.py
max non recursive.py
max recusive.py
travellingSalesmanProblem() › for i in next_permutation
scratch_2

```python
        for j in i:
            current_pathweight += graph[k][j]
            k = j
        current_pathweight += graph[k][s]
        min_path = min(min_path, current_pathweight)
    return min_path
if __name__ == "__main__":
    graph = [[0, 10, 15, 20], [10, 0, 35, 25],
             [15, 35, 0, 30], [20, 25, 30, 0]]
    s = 0
    print(travellingSalesmanProblem(graph, s))
```

```
C:\Users\kadiv\PycharmProjects\pythonProject2\venv\Scripts\python.exe C:/Users/kadiv/AppData/Roaming/JetBrains/PyCharmCE2022.1/scratches/scratc
80

Process finished with exit code 0
```

Scanned with OKEN Scanner