

Actividad 2 - Búsqueda y sistemas basados en reglas

Carlos Eduardo Zamora Guzmán

Javier Alberto Leon Rojas

Corporación universitaria iberoamericana

Inteligencia artificial

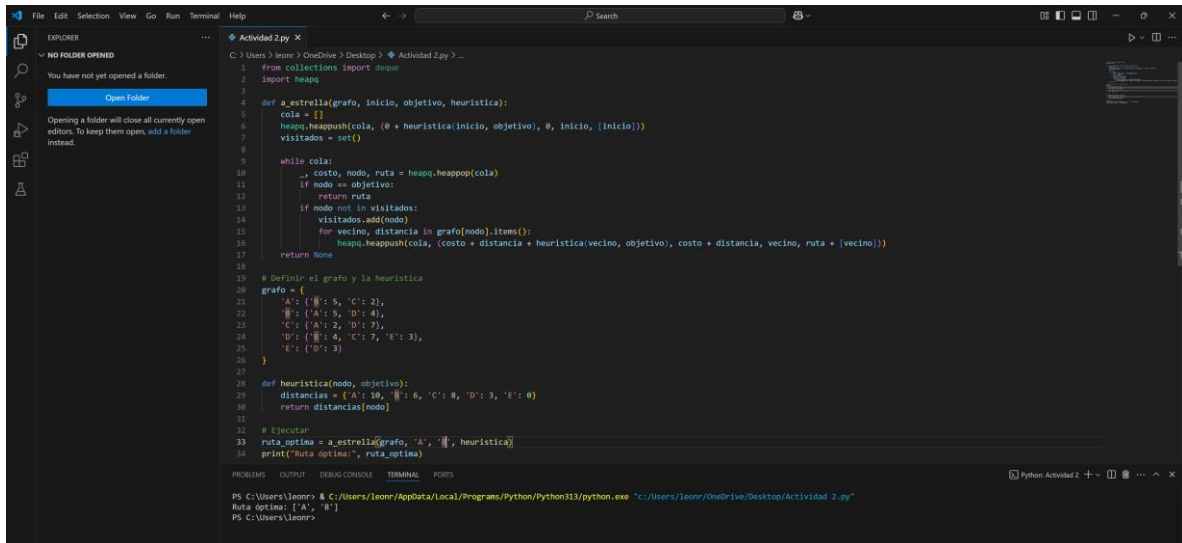
Jorge Castañeda

Bogotá, Colombia

30 de Marzo de 2025

Pruebas realizadas

Ejecución de Ruta punto A al punto B



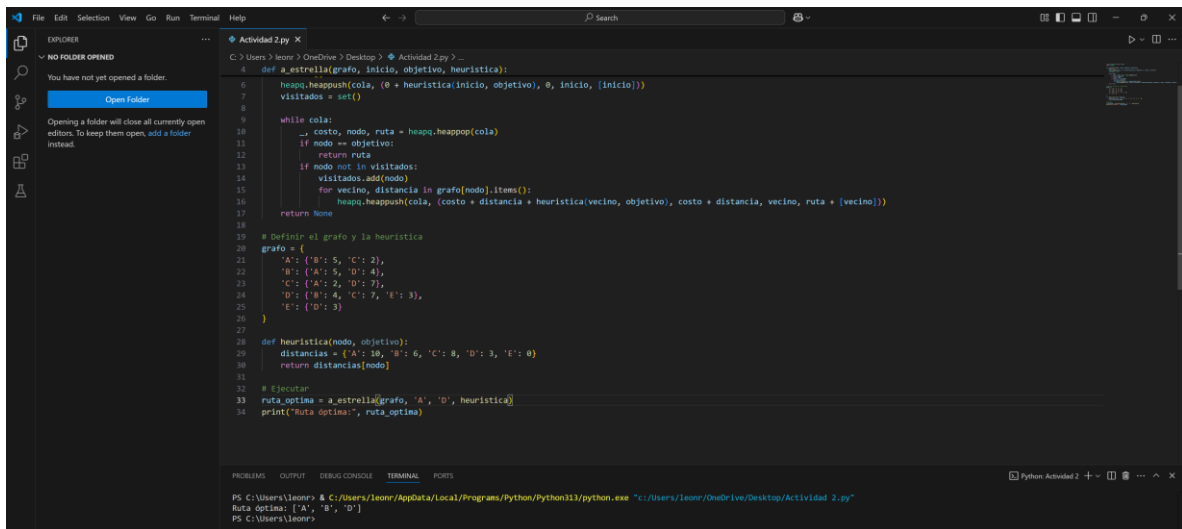
The screenshot shows a VS Code editor with a Python file named 'Actividad 2.py'. The code implements an A* search algorithm to find the shortest path from node 'A' to node 'B' in a graph. The graph is defined as follows:

- Nodes: A, B, C, D, E
- Edges and weights: A-B (5), A-C (2), B-C (4), C-D (2), D-E (3), D-B (4), C-E (3)

The heuristic function calculates the distance from each node to the goal node 'B'. The A* search starts at node 'A' and explores the graph until it reaches node 'B'. The output of the script is:

```
Ruta optima: ['A', 'B']
```

Ejecución de Ruta punto A al punto D

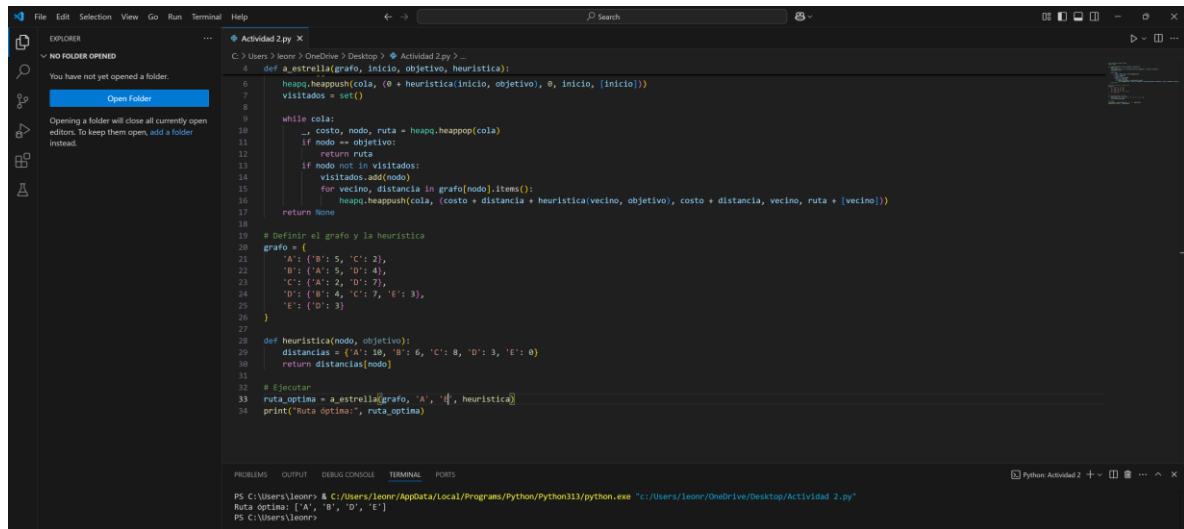


The screenshot shows the same VS Code editor with the 'Actividad 2.py' file. The code is modified to find the shortest path from node 'A' to node 'D'. The graph definition remains the same as in the previous screenshot.

The heuristic function and A* search algorithm are the same, but the goal node is now 'D'. The output of the script is:

```
Ruta optima: ['A', 'B', 'D']
```

Ejecución de Ruta punto A al punto E



```
File Edit Selection View Go Run Terminal Help
Actividad2.py
C:\Users\leone> OneDrive\ Desktop > Actividad2.py
4 def a_estrella(grafo, inicio, objetivo, heuristica):
5     heapq.heappush cola, (0 + heuristica(inicio, objetivo), 0, inicio, [inicio])
6     visitados = set()
7
8     while cola:
9         _, costo, nodo, ruta = heapq.heappop(cola)
10        if nodo == objetivo:
11            return ruta
12        if nodo not in visitados:
13            visitados.add(nodo)
14            for vecino, distancia in grafo[nodo].items():
15                heapq.heappush(cola, (costo + distancia + heuristica(vecino, objetivo), costo + distancia, vecino, ruta + [vecino]))
16        return None
17
18 # Definir el grafo y la heuristica
19 grafo = {
20     'A': {'B': 5, 'C': 2},
21     'B': {'A': 5, 'D': 4},
22     'C': {'A': 2, 'D': 7},
23     'D': {'B': 4, 'C': 7, 'E': 3},
24     'E': {'D': 3}
25 }
26
27 def heuristica(nodo, objetivo):
28     distancias = {'A': 10, 'B': 6, 'C': 8, 'D': 3, 'E': 0}
29     return distancias[nodo]
30
31 # Ejecutar
32 ruta_optima = a_estrella(grafo, 'A', 'E', heuristica)
33 print("Ruta optima:", ruta_optima)
```

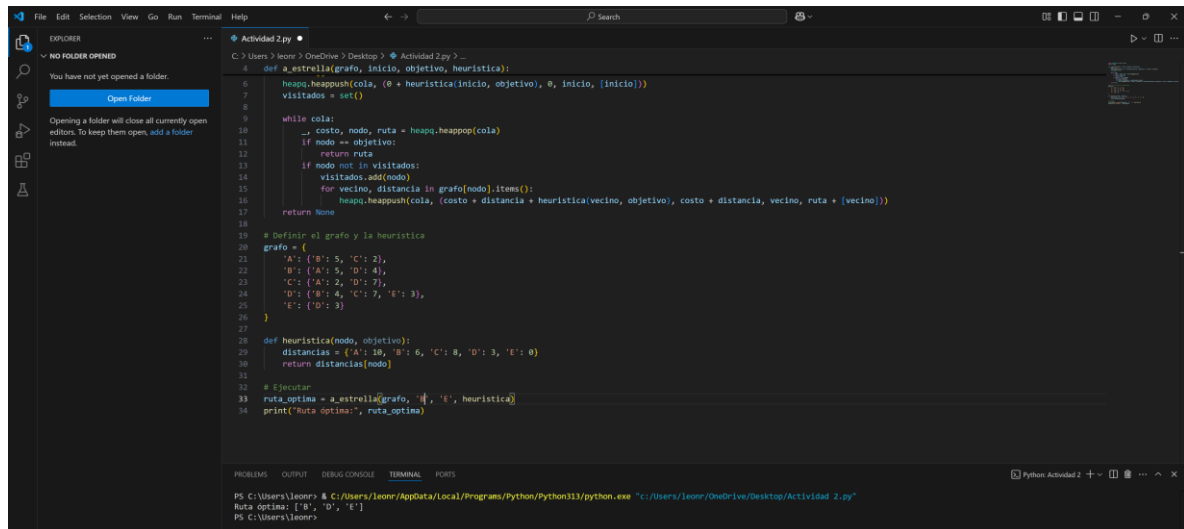
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

PS C:\Users\leone> & C:\Users\leone\AppData\Local\Programs\Python\Python311\python.exe "C:\Users\leone\OneDrive\Desktop\Actividad 2.py"

Ruta optima: ['A', 'B', 'D', 'E']

PS C:\Users\leone>

Ejecución de Ruta punto B al punto E



```
File Edit Selection View Go Run Terminal Help
Actividad2.py
C:\Users\leone> OneDrive\ Desktop > Actividad2.py
4 def a_estrella(grafo, inicio, objetivo, heuristica):
5     heapq.heappush cola, (0 + heuristica(inicio, objetivo), 0, inicio, [inicio])
6     visitados = set()
7
8     while cola:
9         _, costo, nodo, ruta = heapq.heappop(cola)
10        if nodo == objetivo:
11            return ruta
12        if nodo not in visitados:
13            visitados.add(nodo)
14            for vecino, distancia in grafo[nodo].items():
15                heapq.heappush(cola, (costo + distancia + heuristica(vecino, objetivo), costo + distancia, vecino, ruta + [vecino]))
16        return None
17
18 # Definir el grafo y la heuristica
19 grafo = {
20     'A': {'B': 5, 'C': 2},
21     'B': {'A': 5, 'D': 4},
22     'C': {'A': 2, 'D': 7},
23     'D': {'B': 4, 'C': 7, 'E': 3},
24     'E': {'D': 3}
25 }
26
27 def heuristica(nodo, objetivo):
28     distancias = {'A': 10, 'B': 6, 'C': 8, 'D': 3, 'E': 0}
29     return distancias[nodo]
30
31 # Ejecutar
32 ruta_optima = a_estrella(grafo, 'B', 'E', heuristica)
33 print("Ruta optima:", ruta_optima)
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

PS C:\Users\leone> & C:\Users\leone\AppData\Local\Programs\Python\Python311\python.exe "C:\Users\leone\OneDrive\Desktop\Actividad 2.py"

Ruta optima: ['B', 'D', 'E']

PS C:\Users\leone>

Link Video: <https://www.youtube.com/watch?v=HPTSgYV63BQ>

Link GitHub: <https://github.com/javileon10/Actividad-2---B-squeda-y-sistemas-basados-en-reglas>