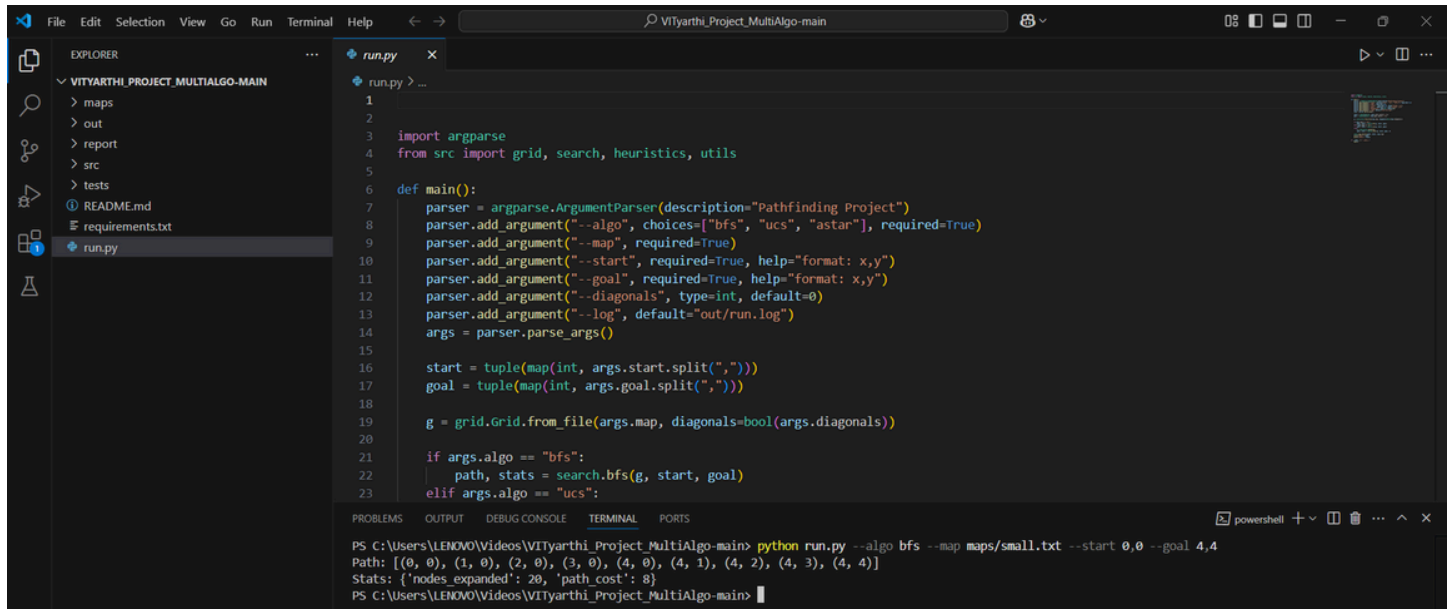


Sample Screenshots

TEST 1 : BFS

- SMALL MAP

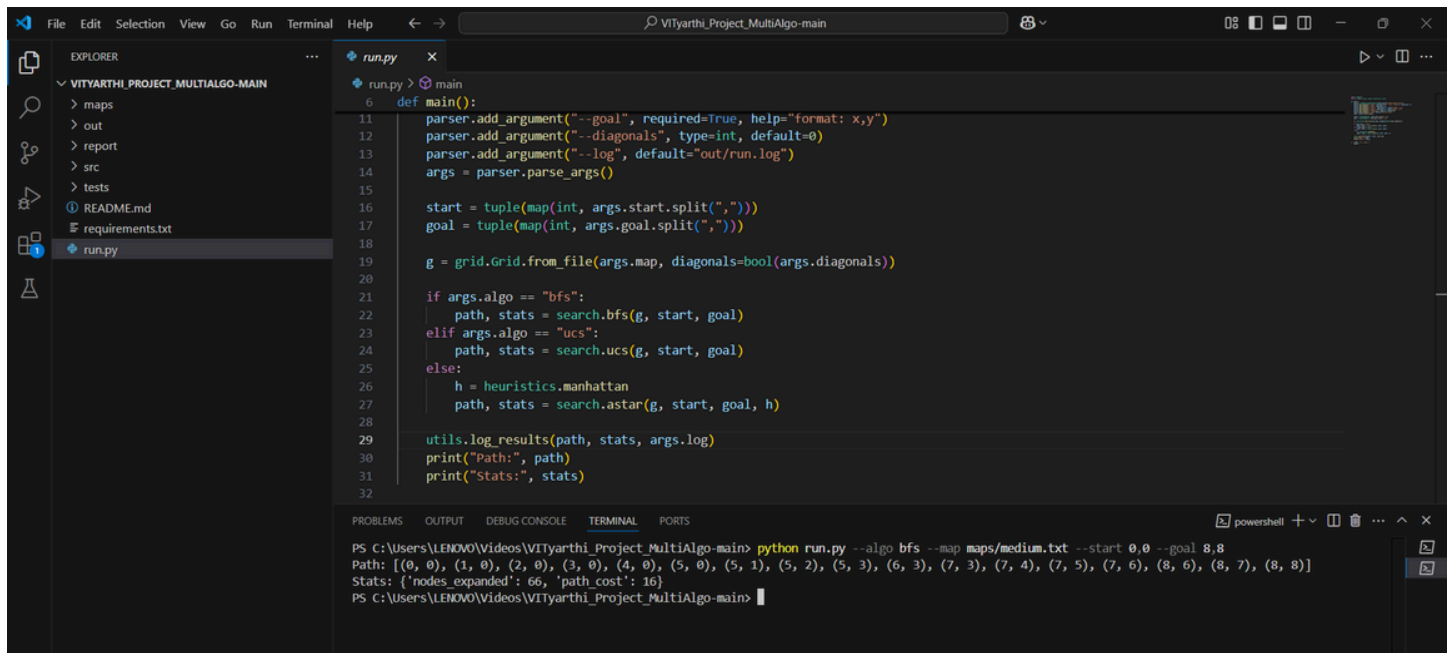


The screenshot shows the Visual Studio Code editor with a project named 'VITYARTHI_PROJECT_MULTIALGO-MAIN'. The Explorer pane on the left shows the project structure with files like 'maps', 'out', 'report', 'src', 'tests', 'README.md', 'requirements.txt', and 'run.py'. The 'run.py' file is open in the editor, showing the following code:

```
1
2
3 import argparse
4 from src import grid, search, heuristics, utils
5
6 def main():
7     parser = argparse.ArgumentParser(description="Pathfinding Project")
8     parser.add_argument("--algo", choices=["bfs", "ucs", "astar"], required=True)
9     parser.add_argument("--map", required=True)
10    parser.add_argument("--start", required=True, help="format: x,y")
11    parser.add_argument("--goal", required=True, help="format: x,y")
12    parser.add_argument("--diagonals", type=int, default=0)
13    parser.add_argument("--log", default="out/run.log")
14    args = parser.parse_args()
15
16    start = tuple(map(int, args.start.split(",")))
17    goal = tuple(map(int, args.goal.split(",")))
18
19    g = grid.Grid.from_file(args.map, diagonals=bool(args.diagonals))
20
21    if args.algo == "bfs":
22        path, stats = search.bfs(g, start, goal)
23    elif args.algo == "ucs":
```

The terminal at the bottom shows the command executed: `python run.py --algo bfs --map maps/small.txt --start 0,0 --goal 4,4`. The output shows the path found: `Path: [(0, 0), (1, 0), (2, 0), (3, 0), (4, 0), (4, 1), (4, 2), (4, 3), (4, 4)]` and the stats: `Stats: {'nodes expanded': 20, 'path cost': 8}`.

- MEDIUM MAP



The screenshot shows the Visual Studio Code editor with the same project. The 'run.py' file is open, showing the following code:

```
6 def main():
11    parser.add_argument("--goal", required=True, help="format: x,y")
12    parser.add_argument("--diagonals", type=int, default=0)
13    parser.add_argument("--log", default="out/run.log")
14    args = parser.parse_args()
15
16    start = tuple(map(int, args.start.split(",")))
17    goal = tuple(map(int, args.goal.split(",")))
18
19    g = grid.Grid.from_file(args.map, diagonals=bool(args.diagonals))
20
21    if args.algo == "bfs":
22        path, stats = search.bfs(g, start, goal)
23    elif args.algo == "ucs":
24        path, stats = search.ucs(g, start, goal)
25    else:
26        h = heuristics.manhattan
27        path, stats = search.astar(g, start, goal, h)
28
29    utils.log_results(path, stats, args.log)
30    print("Path:", path)
31    print("Stats:", stats)
32
```

The terminal at the bottom shows the command executed: `python run.py --algo bfs --map maps/medium.txt --start 0,0 --goal 8,8`. The output shows the path found: `Path: [(0, 0), (1, 0), (2, 0), (3, 0), (4, 0), (5, 0), (5, 1), (5, 2), (5, 3), (6, 3), (7, 3), (7, 4), (7, 5), (7, 6), (8, 6), (8, 7), (8, 8)]` and the stats: `Stats: {'nodes expanded': 66, 'path cost': 16}`.

- LARGE MAP

```
run.py
6 def main():
11     parser.add_argument("--goal", required=True, help="format: x,y")
12     parser.add_argument("--diagonals", type=int, default=0)
13     parser.add_argument("--log", default="out/run.log")
14     args = parser.parse_args()
15
16     start = tuple(map(int, args.start.split(",")))
17     goal = tuple(map(int, args.goal.split(",")))
18
19     g = grid.Grid.from_file(args.map, diagonals=bool(args.diagonals))
20
21     if args.algo == "bfs":
22         path, stats = search.bfs(g, start, goal)
23     elif args.algo == "ucs":
24         path, stats = search.ucs(g, start, goal)
25     else:
26         h = heuristics.manhattan
27         path, stats = search.astar(g, start, goal, h)
28
29     utils.log_results(path, stats, args.log)
30     print("Path:", path)
31     print("Stats:", stats)
32
```

```
PS C:\Users\LENOVO\Videos\VTIYarthi_Project_MultiAlgo-main> python run.py --algo bfs --map maps/large.txt --start 0,0 --goal 10,10
Path: [(0, 0), (1, 0), (2, 0), (3, 0), (4, 0), (5, 0), (6, 0), (7, 0), (8, 0), (9, 0), (10, 0), (11, 0), (12, 0), (12, 1), (12, 2), (12, 3), (12, 4), (12, 5), (12, 6), (12, 7), (12, 8), (11, 8), (10, 8), (10, 9), (10, 10)]
Stats: {'nodes_expanded': 139, 'path_cost': 24}
PS C:\Users\LENOVO\Videos\VTIYarthi_Project_MultiAlgo-main>
```

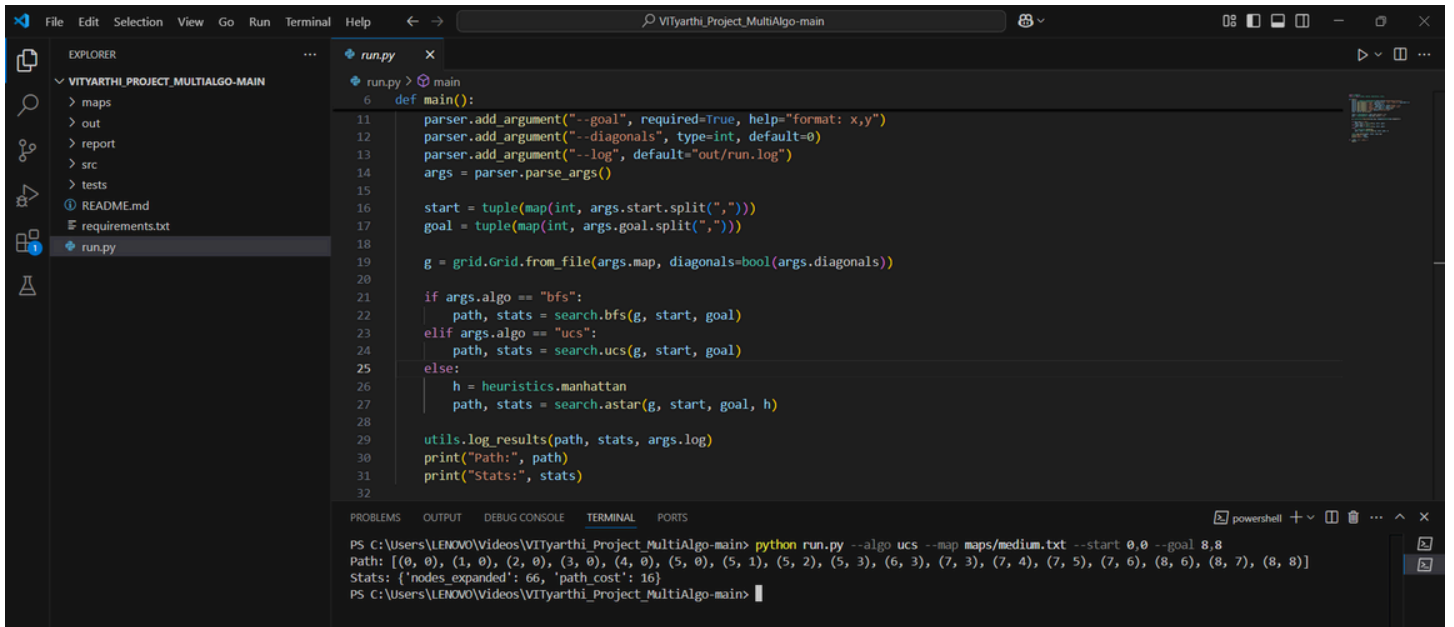
TEST 2: UCS

- SMALL MAP

```
run.py
6 def main():
11     parser.add_argument("--goal", required=True, help="format: x,y")
12     parser.add_argument("--diagonals", type=int, default=0)
13     parser.add_argument("--log", default="out/run.log")
14     args = parser.parse_args()
15
16     start = tuple(map(int, args.start.split(",")))
17     goal = tuple(map(int, args.goal.split(",")))
18
19     g = grid.Grid.from_file(args.map, diagonals=bool(args.diagonals))
20
21     if args.algo == "bfs":
22         path, stats = search.bfs(g, start, goal)
23     elif args.algo == "ucs":
24         path, stats = search.ucs(g, start, goal)
25     else:
26         h = heuristics.manhattan
27         path, stats = search.astar(g, start, goal, h)
28
29     utils.log_results(path, stats, args.log)
30     print("Path:", path)
31     print("Stats:", stats)
32
```

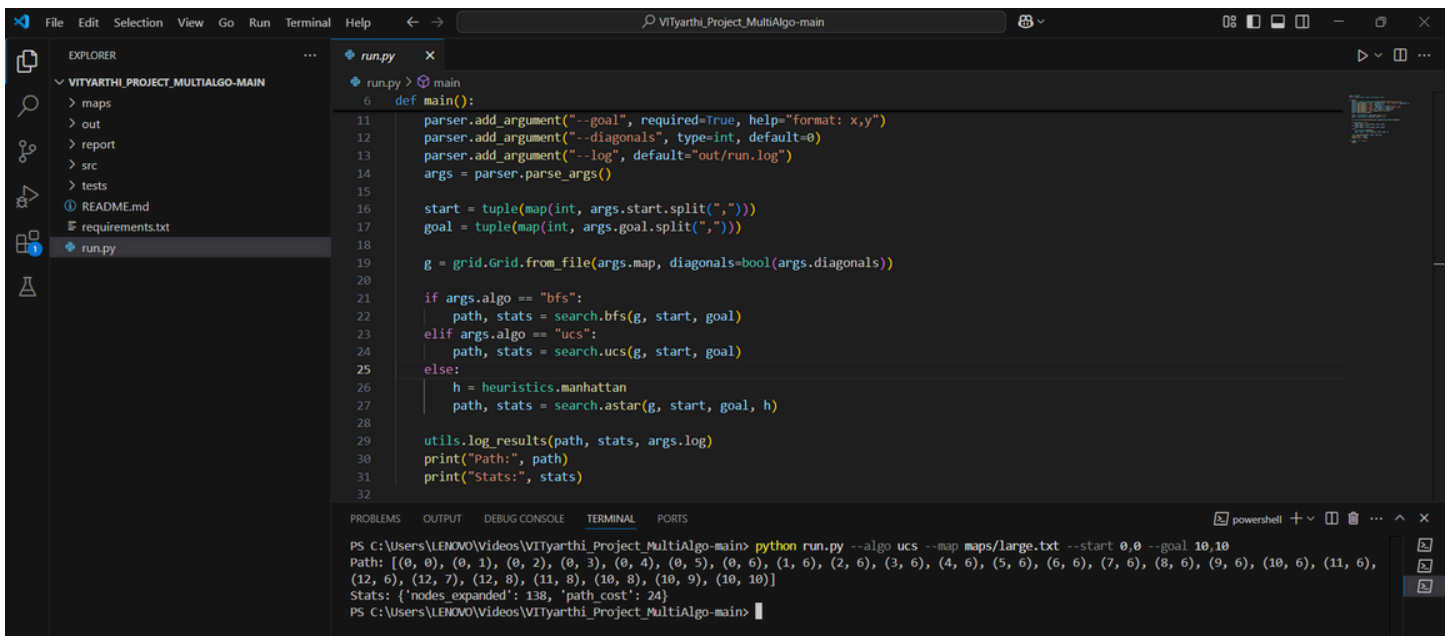
```
PS C:\Users\LENOVO\Videos\VTIYarthi_Project_MultiAlgo-main> python run.py --algo ucs --map maps/small.txt --start 0,0 --goal 4,4
Path: [(0, 0), (0, 1), (0, 2), (0, 3), (0, 4), (1, 4), (2, 4), (3, 4), (4, 4)]
Stats: {'nodes_expanded': 20, 'path_cost': 8}
PS C:\Users\LENOVO\Videos\VTIYarthi_Project_MultiAlgo-main>
```

- MEDIUM MAP



```
run.py x
run.py > main
6 def main():
11     parser.add_argument("--goal", required=True, help="format: x,y")
12     parser.add_argument("--diagonals", type=int, default=0)
13     parser.add_argument("--log", default="out/run.log")
14     args = parser.parse_args()
15
16     start = tuple(map(int, args.start.split(",")))
17     goal = tuple(map(int, args.goal.split(",")))
18
19     g = grid.Grid.from_file(args.map, diagonals=bool(args.diagonals))
20
21     if args.algo == "bfs":
22         path, stats = search.bfs(g, start, goal)
23     elif args.algo == "ucs":
24         path, stats = search.ucs(g, start, goal)
25     else:
26         h = heuristics.manhattan
27         path, stats = search.astar(g, start, goal, h)
28
29     utils.log_results(path, stats, args.log)
30     print("Path:", path)
31     print("Stats:", stats)
32
PS C:\Users\LENOVO\Videos\VTIyarthi_Project_MultiAlgo-main> python run.py --algo ucs --map maps/medium.txt --start 0,0 --goal 8,8
Path: [(0, 0), (1, 0), (2, 0), (3, 0), (4, 0), (5, 0), (5, 1), (5, 2), (5, 3), (6, 3), (7, 3), (7, 4), (7, 5), (7, 6), (8, 6), (8, 7), (8, 8)]
Stats: {'nodes_expanded': 66, 'path_cost': 16}
PS C:\Users\LENOVO\Videos\VTIyarthi_Project_MultiAlgo-main>
```

- **LARGE MAP**



```
run.py x
run.py > main
6 def main():
11     parser.add_argument("--goal", required=True, help="format: x,y")
12     parser.add_argument("--diagonals", type=int, default=0)
13     parser.add_argument("--log", default="out/run.log")
14     args = parser.parse_args()
15
16     start = tuple(map(int, args.start.split(",")))
17     goal = tuple(map(int, args.goal.split(",")))
18
19     g = grid.Grid.from_file(args.map, diagonals=bool(args.diagonals))
20
21     if args.algo == "bfs":
22         path, stats = search.bfs(g, start, goal)
23     elif args.algo == "ucs":
24         path, stats = search.ucs(g, start, goal)
25     else:
26         h = heuristics.manhattan
27         path, stats = search.astar(g, start, goal, h)
28
29     utils.log_results(path, stats, args.log)
30     print("Path:", path)
31     print("Stats:", stats)
32
PS C:\Users\LENOVO\Videos\VTIyarthi_Project_MultiAlgo-main> python run.py --algo ucs --map maps/large.txt --start 0,0 --goal 10,10
Path: [(0, 0), (0, 1), (0, 2), (0, 3), (0, 4), (0, 5), (0, 6), (1, 6), (2, 6), (3, 6), (4, 6), (5, 6), (6, 6), (7, 6), (8, 6), (9, 6), (10, 6), (11, 6), (12, 6), (12, 7), (12, 8), (11, 8), (10, 8), (10, 9), (10, 10)]
Stats: {'nodes_expanded': 138, 'path_cost': 24}
PS C:\Users\LENOVO\Videos\VTIyarthi_Project_MultiAlgo-main>
```

TEST 3 : A* (A STAR) WITH DIAGONALS

- **SMALL MAP**

```
File Edit Selection View Go Run Terminal Help
VITYarthi_Project_MultiAlgo-main

EXPLORER
VITYARTH_PROJECT_MULTIALGO-MAIN
  maps
  out
  report
  src
  tests
  README.md
  requirements.txt
  run.py

run.py
6 def main():
11     parser.add_argument("--goal", required=True, help="format: x,y")
12     parser.add_argument("--diagonals", type=int, default=0)
13     parser.add_argument("--log", default="out/run.log")
14     args = parser.parse_args()
15
16     start = tuple(map(int, args.start.split(",")))
17     goal = tuple(map(int, args.goal.split(",")))
18
19     g = grid.Grid.from_file(args.map, diagonals=bool(args.diagonals))
20
21     if args.algo == "bfs":
22         path, stats = search.bfs(g, start, goal)
23     elif args.algo == "ucs":
24         path, stats = search.ucs(g, start, goal)
25     else:
26         h = heuristics.manhattan
27         path, stats = search.astar(g, start, goal, h)
28
29     utils.log_results(path, stats, args.log)
30     print("Path:", path)
31     print("Stats:", stats)
32
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS C:\Users\LENOVO\Videos\VITYarthi_Project_MultiAlgo-main> python run.py --algo astar --map maps/small.txt --start 0,0 --goal 4,4 --diagonals 1
Path: [(0, 0), (0, 1), (1, 2), (2, 3), (3, 4), (4, 4)]
Stats: {'nodes_expanded': 6, 'path_cost': 5}
PS C:\Users\LENOVO\Videos\VITYarthi_Project_MultiAlgo-main>
```

• MEDIUM MAP

```
File Edit Selection View Go Run Terminal Help
VITYarthi_Project_MultiAlgo-main

EXPLORER
VITYARTH_PROJECT_MULTIALGO-MAIN
  maps
  out
  report
  src
  tests
  README.md
  requirements.txt
  run.py

run.py
6 def main():
11     parser.add_argument("--goal", required=True, help="format: x,y")
12     parser.add_argument("--diagonals", type=int, default=0)
13     parser.add_argument("--log", default="out/run.log")
14     args = parser.parse_args()
15
16     start = tuple(map(int, args.start.split(",")))
17     goal = tuple(map(int, args.goal.split(",")))
18
19     g = grid.Grid.from_file(args.map, diagonals=bool(args.diagonals))
20
21     if args.algo == "bfs":
22         path, stats = search.bfs(g, start, goal)
23     elif args.algo == "ucs":
24         path, stats = search.ucs(g, start, goal)
25     else:
26         h = heuristics.manhattan
27         path, stats = search.astar(g, start, goal, h)
28
29     utils.log_results(path, stats, args.log)
30     print("Path:", path)
31     print("Stats:", stats)
32
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS C:\Users\LENOVO\Videos\VITYarthi_Project_MultiAlgo-main> python run.py --algo astar --map maps/medium.txt --start 0,0 --goal 8,8 --diagonals 1
Path: [(0, 0), (0, 1), (1, 2), (2, 2), (3, 3), (3, 4), (2, 5), (3, 6), (4, 7), (5, 8), (6, 8), (7, 9), (8, 8)]
Stats: {'nodes_expanded': 16, 'path_cost': 12}
PS C:\Users\LENOVO\Videos\VITYarthi_Project_MultiAlgo-main>
```

• LARGE MAP

The screenshot shows the Visual Studio Code interface with the file explorer on the left displaying the project structure. The main editor shows the code for `run.py`, which defines a `main` function. The function uses a command-line parser to handle arguments for goal, diagonals, log file, start, goal, and diagonals. It then creates a grid from a file and performs a search using the A* algorithm. The terminal at the bottom shows the command `python run.py --algo astar --map maps/large.txt --start 0,0 --goal 10,10 --diagonals 1` and the resulting path and stats.

```
def main():
    parser.add_argument("--goal", required=True, help="format: x,y")
    parser.add_argument("--diagonals", type=int, default=0)
    parser.add_argument("--log", default="out/run.log")
    args = parser.parse_args()

    start = tuple(map(int, args.start.split(",")))
    goal = tuple(map(int, args.goal.split(",")))

    g = grid.Grid.from_file(args.map, diagonals=bool(args.diagonals))

    if args.algo == "bfs":
        path, stats = search.bfs(g, start, goal)
    elif args.algo == "ucs":
        path, stats = search.ucs(g, start, goal)
    else:
        h = heuristics.manhattan
        path, stats = search.astar(g, start, goal, h)

    utils.log_results(path, stats, args.log)
    print("Path:", path)
    print("Stats:", stats)
```

```
PS C:\Users\LENOVO\Videos\VITYarthi_Project_MultiAlgo-main> python run.py --algo astar --map maps/large.txt --start 0,0 --goal 10,10 --diagonals 1
Path: [(0, 0), (0, 1), (1, 2), (2, 3), (3, 4), (4, 4), (5, 4), (6, 5), (7, 6), (8, 7), (9, 6), (10, 6), (11, 6), (12, 7), (11, 8), (10, 9), (10, 10)]
Stats: {'nodes_expanded': 61, 'path_cost': 16}
PS C:\Users\LENOVO\Videos\VITYarthi_Project_MultiAlgo-main>
```

TEST 4 : BFS ON LARGE MAP WITH LOGGING

The screenshot shows the Visual Studio Code interface with the file explorer on the left displaying the project structure. The main editor shows the code for `run.py`, which defines a `main` function. The function uses a command-line parser to handle arguments for goal, diagonals, log file, start, goal, and diagonals. It then creates a grid from a file and performs a search using the BFS algorithm. The terminal at the bottom shows the command `python run.py --algo bfs --map maps/dynamic.txt --start 0,0 --goal 5,5` and the resulting path and stats.

```
def main():
    parser.add_argument("--goal", required=True, help="format: x,y")
    parser.add_argument("--diagonals", type=int, default=0)
    parser.add_argument("--log", default="out/run.log")
    args = parser.parse_args()

    start = tuple(map(int, args.start.split(",")))
    goal = tuple(map(int, args.goal.split(",")))

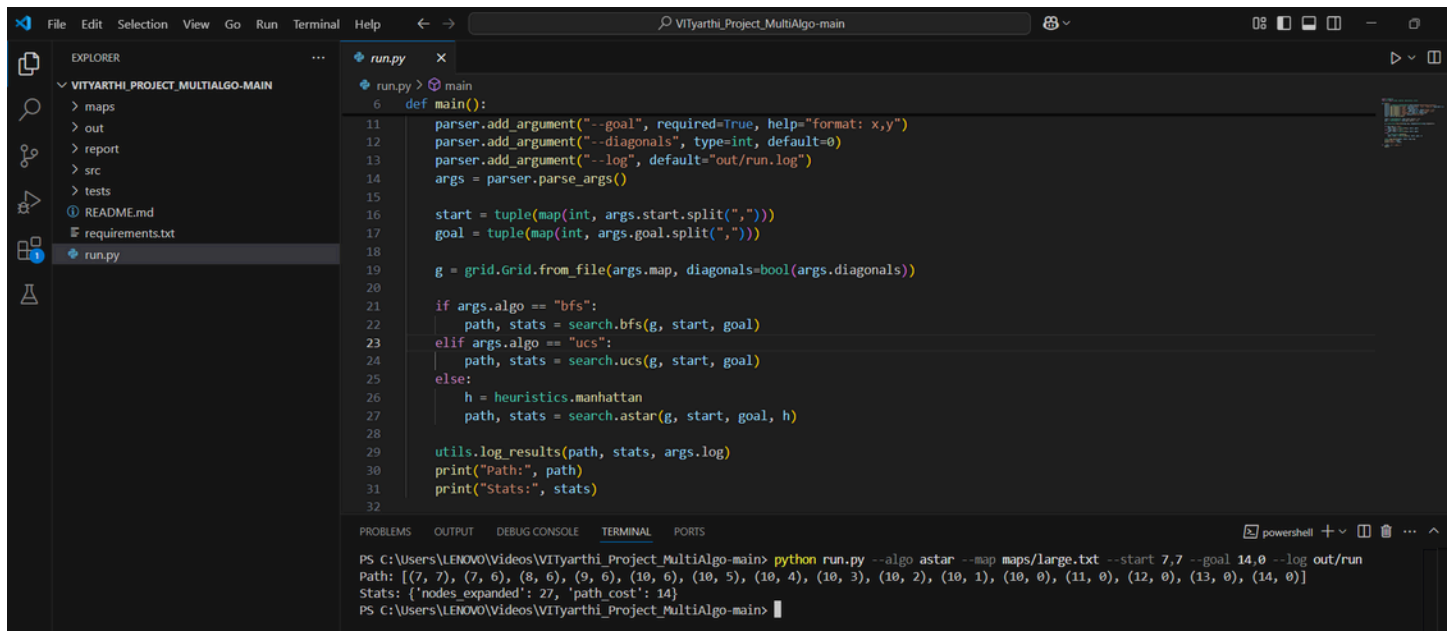
    g = grid.Grid.from_file(args.map, diagonals=bool(args.diagonals))

    if args.algo == "bfs":
        path, stats = search.bfs(g, start, goal)
    elif args.algo == "ucs":
        path, stats = search.ucs(g, start, goal)
    else:
        h = heuristics.manhattan
        path, stats = search.astar(g, start, goal, h)

    utils.log_results(path, stats, args.log)
    print("Path:", path)
    print("Stats:", stats)
```

```
PS C:\Users\LENOVO\Videos\VITYarthi_Project_MultiAlgo-main> python run.py --algo bfs --map maps/dynamic.txt --start 0,0 --goal 5,5
Path: [(0, 0), (1, 0), (2, 0), (3, 0), (4, 0), (5, 0), (5, 1), (5, 2), (5, 3), (5, 4), (5, 5)]
Stats: {'nodes_expanded': 42, 'path_cost': 10}
PS C:\Users\LENOVO\Videos\VITYarthi_Project_MultiAlgo-main>
```

TEST 5: A* ON LARGE MAP, LOG TO CUSTOM FILE



The image shows a Visual Studio Code editor window with a project named "VITYARTHI_PROJECT_MultiAlgo-main". The Explorer sidebar on the left shows the project structure with files like maps, out, report, src, tests, README.md, requirements.txt, and run.py. The main editor displays the code in run.py, which is a Python script for a pathfinding algorithm. The script uses argparse for command-line arguments, a Grid class for the map, and search algorithms (bfs, ucs, astar) to find a path from a start point to a goal point. The terminal at the bottom shows the command to run the script and its output, which includes the path found and the statistics of the search.

```
run.py
6 def main():
11     parser.add_argument("--goal", required=True, help="format: x,y")
12     parser.add_argument("--diagonals", type=int, default=0)
13     parser.add_argument("--log", default="out/run.log")
14     args = parser.parse_args()
15
16     start = tuple(map(int, args.start.split(",")))
17     goal = tuple(map(int, args.goal.split(",")))
18
19     g = grid.Grid.from_file(args.map, diagonals=bool(args.diagonals))
20
21     if args.algo == "bfs":
22         path, stats = search.bfs(g, start, goal)
23     elif args.algo == "ucs":
24         path, stats = search.ucs(g, start, goal)
25     else:
26         h = heuristics.manhattan
27         path, stats = search.astar(g, start, goal, h)
28
29     utils.log_results(path, stats, args.log)
30     print("Path:", path)
31     print("Stats:", stats)
32
```

Terminal Output:

```
PS C:\Users\LENOVO\Videos\VITYarthi_Project_MultiAlgo-main> python run.py --algo astar --map maps/large.txt --start 7,7 --goal 14,0 --log out/run
Path: [(7, 7), (7, 6), (8, 6), (9, 6), (10, 6), (10, 5), (10, 4), (10, 3), (10, 2), (10, 1), (10, 0), (11, 0), (12, 0), (13, 0), (14, 0)]
Stats: {'nodes_expanded': 27, 'path_cost': 14}
PS C:\Users\LENOVO\Videos\VITYarthi_Project_MultiAlgo-main>
```

NAME : K.BALA YASWANTH

REG.NO : 24MIM01243