



ĐẠI HỌC KHOA HỌC TỰ NHIÊN, ĐHQG-HCM
KHOA CÔNG NGHỆ THÔNG TIN

CƠ SỞ TRÍ TUỆ NHÂN TẠO
Báo cáo Đồ án 1

CHỦ ĐỀ: TÌM KIẾM HEURISTIC VỚI A^*

Nhóm thực hiện

- | | |
|--------------------|---------|
| 1. Võ Nhật Vinh | 1612815 |
| 2. Hồng Thanh Hoài | 1612855 |
| 3. Huỳnh Minh Huân | 1612858 |
-

Giáo viên lý thuyết
PGS.TS Lê Hoài Bắc

Giáo viên thực hành
Châu Ngọc Phương

Tháng 10 năm 2018

Lời cảm ơn

Trong quá trình thực hiện đề án này, nhóm chúng em đã nhận được rất nhiều sự giúp đỡ cũng như hỗ trợ từ các thầy cô Trường Đại học Khoa học Tự nhiên, ĐHQG-HCM và các bạn bè trong trường. Nhóm chúng em xin bày tỏ lòng cảm ơn chân thành đến mọi người vì đã hướng dẫn, chỉ bảo rất tận tình.

Đặc biệt, nhóm chúng em xin bày tỏ lòng biết ơn sâu sắc đến các thầy cô khoa Công nghệ thông tin, cụ thể hơn chúng em xin cảm ơn thầy Lê Hoài Bắc và thầy Châu Ngọc Phương đã hỗ trợ, giảng dạy rất kỹ lưỡng để chúng em có thể hoàn thành tốt đề án này.

Một lần nữa, chúng em xin bày tỏ lòng biết ơn sâu sắc đến với các thầy cô và bạn bè.

Tháng 10 năm 2018,
Đại học Khoa học Tự nhiên, ĐHQG-HCM.

Mục lục

Lời cảm ơn	i
1 Giới thiệu nhóm và phân công công việc	1
1.1 Giới thiệu nhóm	1
1.2 Phân công công việc	1
2 Giới thiệu đồ án	2
2.1 Thời gian và công cụ thực hiện	2
2.2 Các bước thực hiện	2
3 Nội dung đồ án	3
3.1 Sơ đồ UML	3
3.2 Các hàm chính	3
3.3 Cấu trúc dữ liệu	4
3.4 Thuật toán chính	4
4 Kiểm thử	5
4.1 Testcase với kích thước nhỏ	5
4.2 Testcase với kích thước lớn	7
4.3 Testcase không có đường đi từ start đến goal	8
5 Đánh giá và tổng kết quá trình	10
5.1 Mức độ hoàn thành của đồ án	10
5.2 Những vấn đề chưa thực hiện được	10
Tài liệu tham khảo	11

1 Giới thiệu nhóm và phân công công việc

1.1 Giới thiệu nhóm

Nhóm gồm 3 thành viên.

STT	Họ và tên	MSSV	Email
1	Võ Nhật Vinh	1612815	nhatvinhvo1998@gmail.com
2	Hồng Thanh Hoài	1612855	hthoai1006@gmail.com
3	Huỳnh Minh Huấn	1612858	minhhuanhuynh289@gmail.com

1.2 Phân công công việc

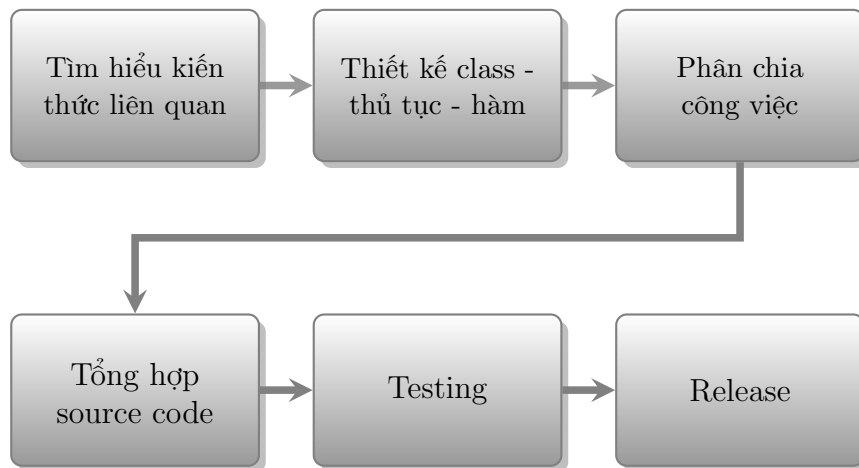
STT	Họ và tên	Công việc	Mức độ hoàn thành
1	Huỳnh Minh Huấn	Hàm <code>astar-search</code>	100%
2	Hồng Thanh Hoài	Hàm <code>main</code> Lớp <code>Cell</code>	100%
3	Võ Nhật Vinh	Lớp <code>PriorityQueue</code> Viết testcases Viết báo cáo	100%

2 Giới thiệu đồ án

2.1 Thời gian và công cụ thực hiện

- Thời gian thực hiện: từ ngày 10/10/2018 đến ngày 22/10/2018.
 - 10/10/2018 – 14/10/2018: Tìm hiểu kiến thức liên quan.
 - 14/10/2018 – 20/10/2018: Thiết kế thuật toán, class, hàm... và tiến hành code.
 - 20/10/2018 – 22/10/2018: Tổng hợp source code, testing, release và viết báo cáo.
- Công cụ làm việc nhóm: Facebook.
- Công cụ quản lý *source code*: Github.
- *Text editor*: Visual Studio Code.

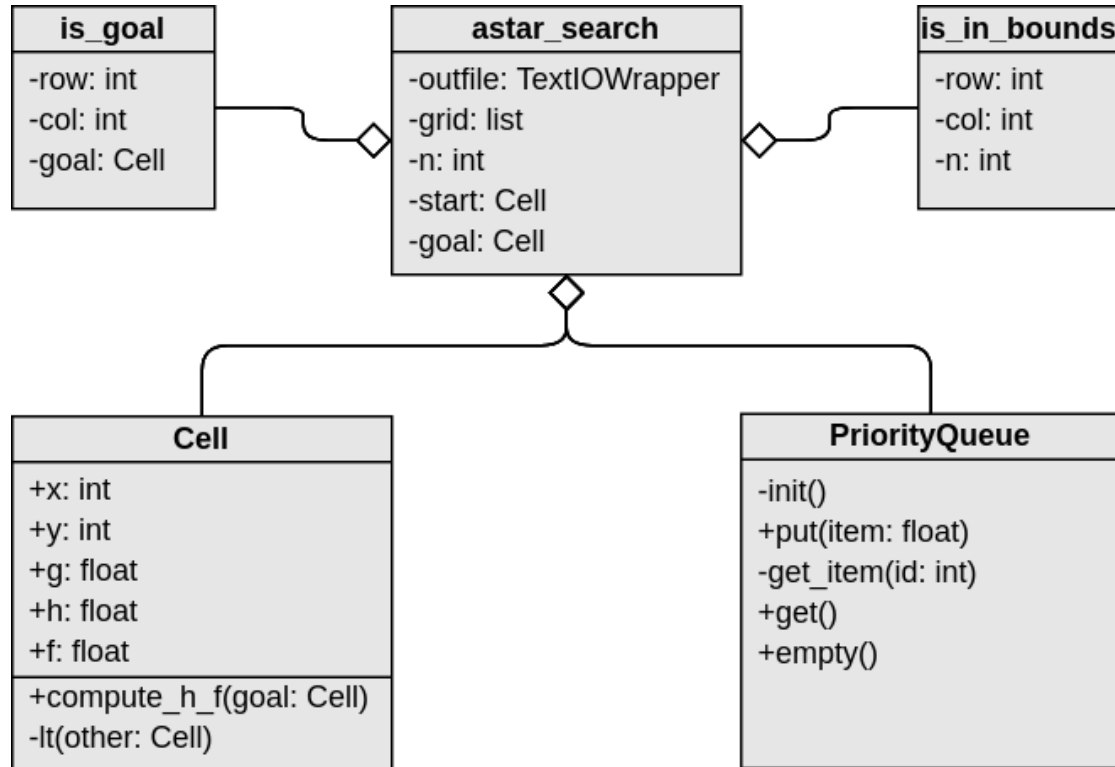
2.2 Các bước thực hiện



3 Nội dung đề án

3.1 Sơ đồ UML

Sơ đồ UML thể hiện thiết kế hàm, class.



3.2 Các hàm chính

- **astar-search**
 - Tham số đầu vào: **outfile** (chứa kết quả), **grid** (chứa bản đồ), **n** (kích thước bản đồ: $n \times n$), **start** - **goal** (điểm xuất phát và điểm đích).
 - Công dụng: Dùng thuật toán A* để tìm đường đi từ một điểm đến một điểm cho trước, trả về `-1` nếu không tìm thấy đường đi.
- **is-in-bounds**
 - Tham số đầu vào: **row** - **col** (tọa độ của ô cần kiểm tra), **n** (kích thước bản đồ).
 - Công dụng: Kiểm tra xem một ô có nằm trong bản đồ hay không.
- **is-goal**
 - Tham số đầu vào: **row** - **col** (tọa độ của ô cần kiểm tra), **goal** (ô đích).
 - Công dụng: Kiểm tra xem một ô có phải là đích hay không.

3.3 Cấu trúc dữ liệu

- Class `Cell` với các thuộc tính:
 - + `x, y`: tọa độ của ô trên bản đồ.
 - + `g`: hàm chi phí đi từ điểm bắt đầu đến điểm hiện hành.
 - + `h`: hàm heuristic ước lượng khoảng cách từ điểm hiện hành đến đích.
 - + `f`: $f = g + h$.
- `PriorityQueue`: hàng đợi ưu tiên.

3.4 Thuật toán chính

Thuật toán chính sử dụng là tìm kiếm A.*

- Bước 1: Nếu `start` và `goal` không nằm trong bản đồ hoặc là vật cản thì kết thúc thuật toán.
- Bước 2: Khởi tạo `open-list`, khởi tạo `closed-list` với tất cả giá trị là `False`, và khởi tạo `parent-list` với tất cả các giá trị có tọa độ `(-1, -1)` và `f = FLOAT-MAX`. Đưa `start` vào `open-list`.
- Bước 3: Lặp khi `open-list` không rỗng:
 - Lấy ra node có giá trị `f` thấp nhất trong `open-list`, gọi là `q`. Gán `True` cho vị trí của `q` trong `close list`.
 - Nếu `q` là `goal`, tiến hành quay lui thông qua `parent-list` để tìm đường đi và kết thúc thuật toán.
 - Khởi tạo 8 successor xung quanh lân cận 8 của `q`.
 - Với mỗi successor:
 - + Nếu successor nằm trong bản đồ, không phải là vật cản, và giá trị trong `closed-list` là `False` thì tiến hành tính `g, h, f` cho successor này.
 - + Nếu `parent` của successor này có `f = FLOAT-MAX` hoặc `f` lớn hơn `f` của successor, ta gán `parent` mới cho successor với tọa độ là tọa độ của `q`, `f` là `f` của successor. Và thêm successor này vào `open-list`.
- Bước 4: Nếu `open-list` đã rỗng mà chưa tìm thấy đường đi thì kết thúc thuật toán.

4 Kiểm thử

Sử dụng 10 testcase với ba đặc trưng khác nhau là: kích thước nhỏ, kích thước lớn, và không có đường đi.

4.1 Testcase với kích thước nhỏ

The screenshot shows a code editor with two files: `input.txt` and `output.txt`. The `input.txt` file contains the following text:

```
You, 4 days ago | 1 author (You)
1 7
2 0 0
3 6 6
4 0 0 0 0 0 0 1
5 0 0 0 0 0 1 1
6 1 1 0 0 0 0 1
7 0 1 1 0 0 0 0
8 0 1 1 0 0 1 0
9 0 1 0 0 0 1 0
10 0 0 0 0 0 0 0
```

The `output.txt` file contains the following text:

```
1 8
2 (0, 0) (1, 1) (2, 2) (3, 3) (4, 4) (5, 4) (6, 5) (6, 6)
3 S - - - - 0
4 - 0 - - - 0 0
5 0 0 0 - - 0
6 - 0 0 0 - -
7 - 0 0 - 0 0 -
8 - 0 - - 0 0 -
9 - - - - 0 G
```

Hình 1: Testcase 8x8.

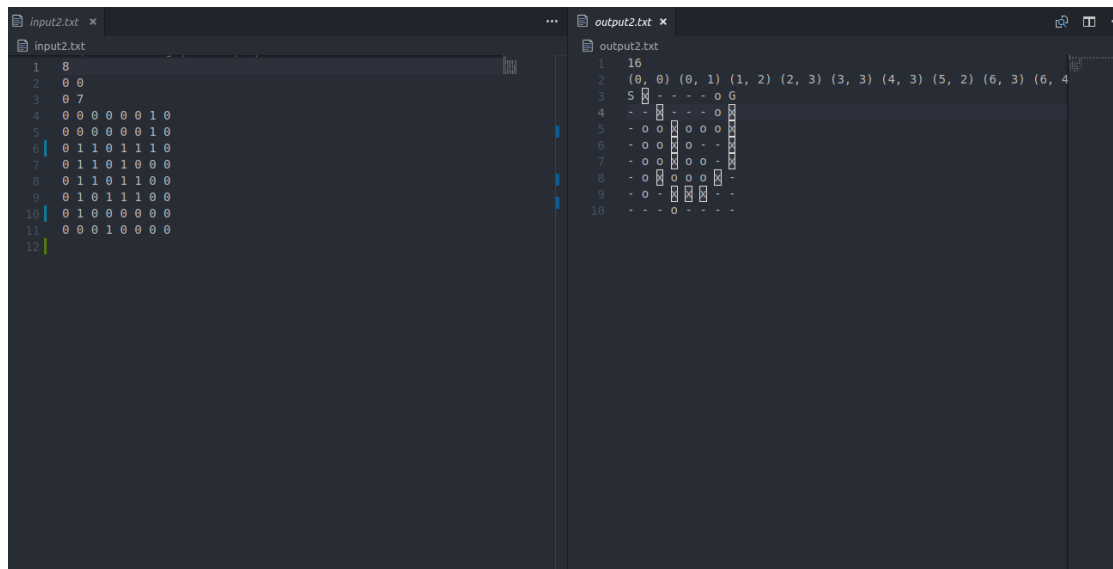
The screenshot shows a code editor with two files: `input1.txt` and `output1.txt`. The `input1.txt` file contains the following text:

```
1 8
2 0 0
3 7 0
4 0 0 0 0 0 0 1 1
5 0 0 0 0 0 1 1 0
6 1 1 0 1 0 0 1 0
7 0 1 1 0 0 0 0 0
8 0 1 1 0 0 1 0 0
9 0 1 0 1 1 1 0 0
10 0 0 0 0 0 0 0 0
11 0 0 0 0 0 0 0 0
12
```

The `output1.txt` file contains the following text:

```
1 8
2 (0, 0) (1, 1) (2, 2) (3, 3) (4, 3) (5, 2) (6, 1) (7, 0)
3 S - - - - 0 0
4 - 0 - - - 0 0 -
5 0 0 0 0 - 0 0 -
6 - 0 0 0 0 - - -
7 - 0 0 0 0 0 - -
8 - 0 0 0 0 0 0 -
9 - 0 0 0 0 0 0 -
10 G - - - - -
```

Hình 2: Testcase 8x8.



The screenshot shows a code editor with two tabs: `input2.txt` and `output2.txt`. The `input2.txt` tab contains an 8x8 grid of numbers (0-9) representing a test case. The `output2.txt` tab shows the result of a search algorithm, including a path of coordinates and a grid of numbers.

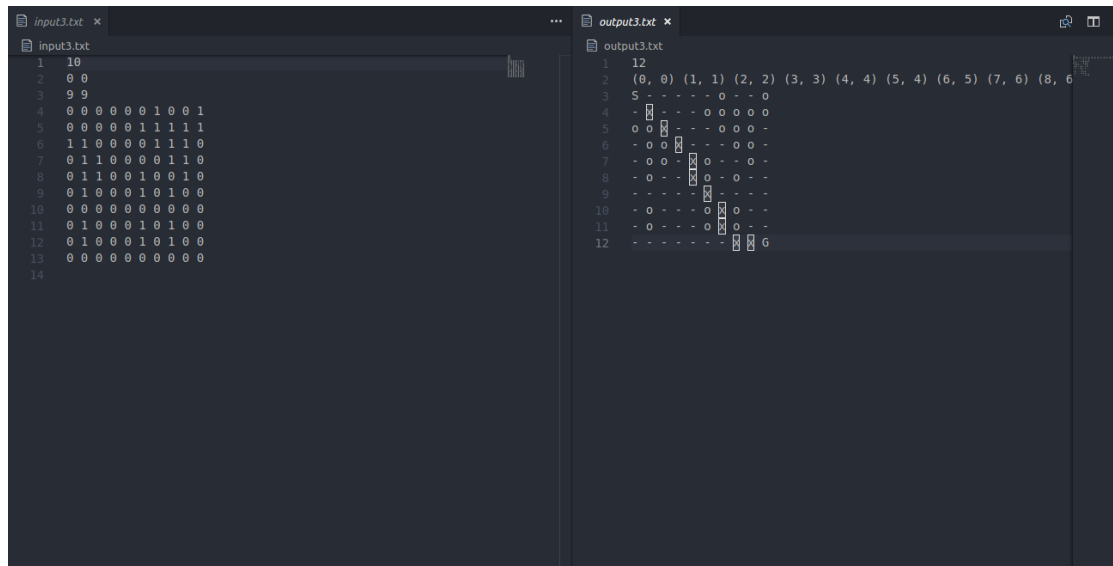
```

input2.txt
1 8
2 0 0
3 0 7
4 0 0 0 0 0 1 0
5 0 0 0 0 0 0 1 0
6 0 1 1 0 1 1 1 0
7 0 1 1 0 1 0 0 0
8 0 1 1 0 1 1 0 0
9 0 1 0 1 1 1 0 0
10 0 1 0 0 0 0 0 0
11 0 0 0 1 0 0 0 0
12

output2.txt
1 16
2 (0, 0) (0, 1) (1, 2) (2, 3) (3, 3) (4, 3) (5, 2) (6, 3) (6, 4)
3 S - - - - - 0 G
4 - - - - - 0
5 - 0 0 0 0 0 0
6 - 0 0 0 0 0 -
7 - 0 0 0 0 0 -
8 - 0 0 0 0 0 -
9 - 0 - - - - -
10 - - - 0 - - -

```

Hình 3: Testcase 8x8.



The screenshot shows a code editor with two tabs: `input3.txt` and `output3.txt`. The `input3.txt` tab contains a 10x10 grid of numbers (0-9) representing a test case. The `output3.txt` tab shows the result of a search algorithm, including a path of coordinates and a grid of numbers.

```

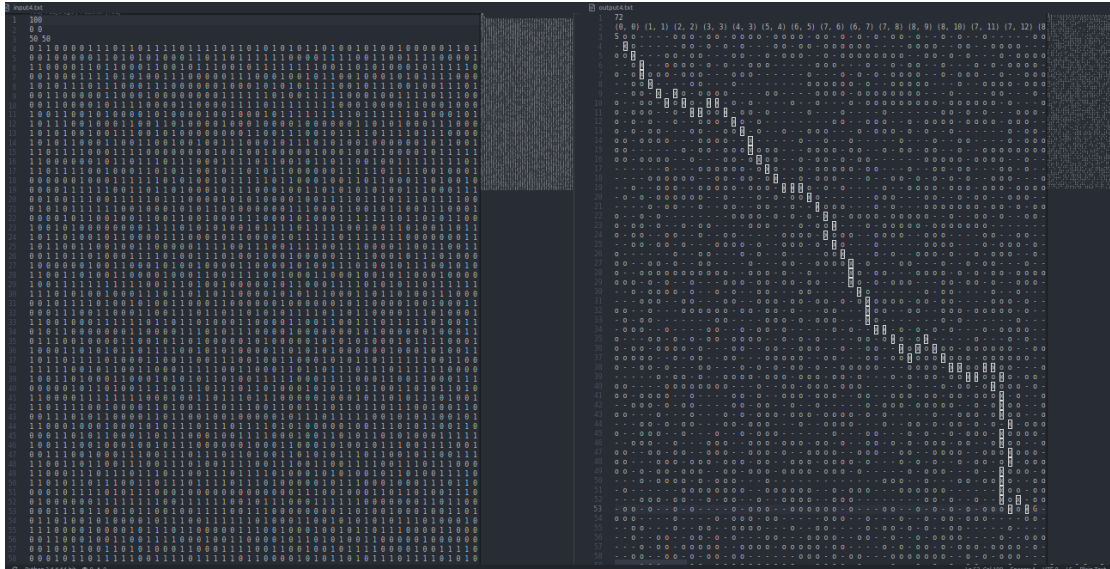
input3.txt
1 10
2 0 0
3 9 9
4 0 0 0 0 0 0 1 0 0 1
5 0 0 0 0 0 1 1 1 1 1
6 1 1 0 0 0 0 1 1 1 0
7 0 1 1 0 0 0 0 1 1 0
8 0 1 1 0 0 1 0 0 1 0
9 0 1 0 0 0 1 0 1 0 0
10 0 0 0 0 0 0 0 0 0 0
11 0 1 0 0 0 1 0 1 0 0
12 0 1 0 0 0 1 0 1 0 0
13 0 0 0 0 0 0 0 0 0 0
14

output3.txt
1 12
2 (0, 0) (1, 1) (2, 2) (3, 3) (4, 4) (5, 4) (6, 5) (7, 6) (8, 6)
3 S - - - - - 0 - - 0
4 - - - - - 0 0 0 0 0
5 0 0 - - - - - 0 0 0
6 - 0 0 - - - - - 0 0
7 - 0 0 - - - - - 0 -
8 - 0 - - - - - 0 -
9 - - - - - - - - -
10 - 0 - - - - - 0 -
11 - 0 - - - - - 0 -
12 - - - - - - - - G

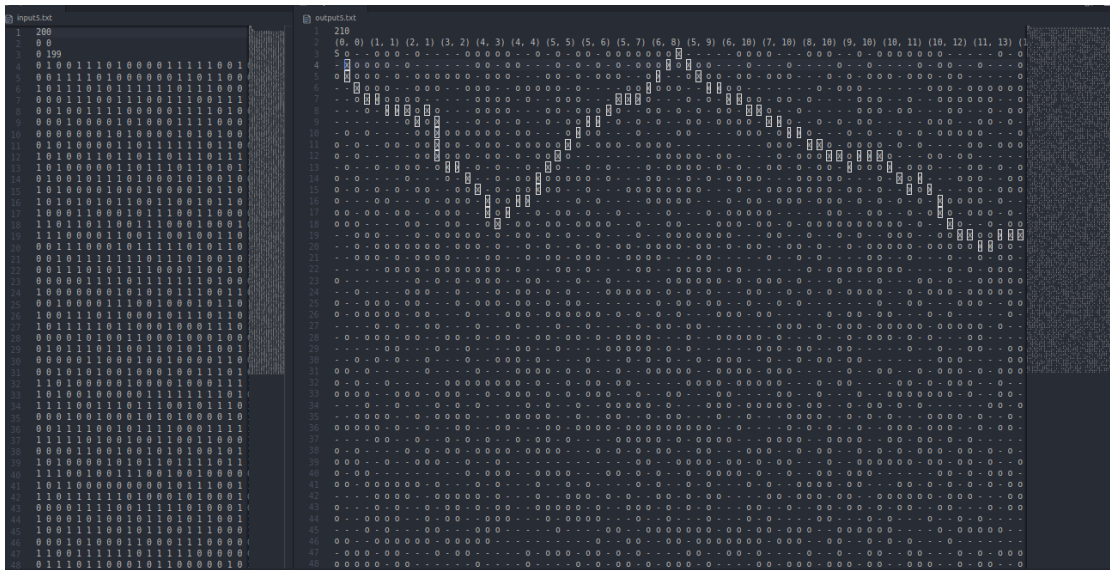
```

Hình 4: Testcase 10x10.

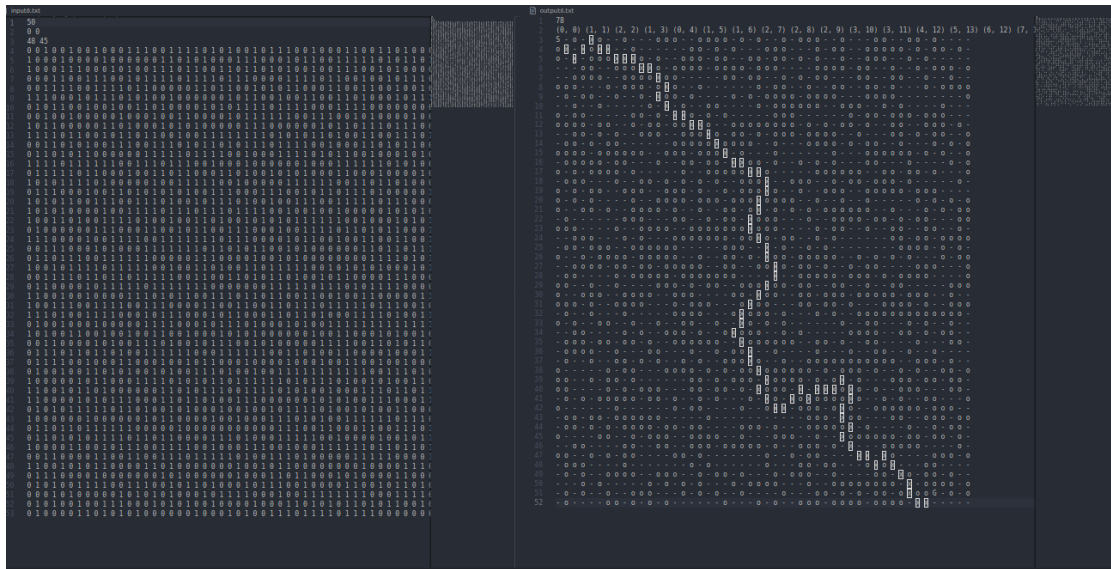
4.2 Testcase với kích thước lớn



Hình 5: Testcase 100x100.

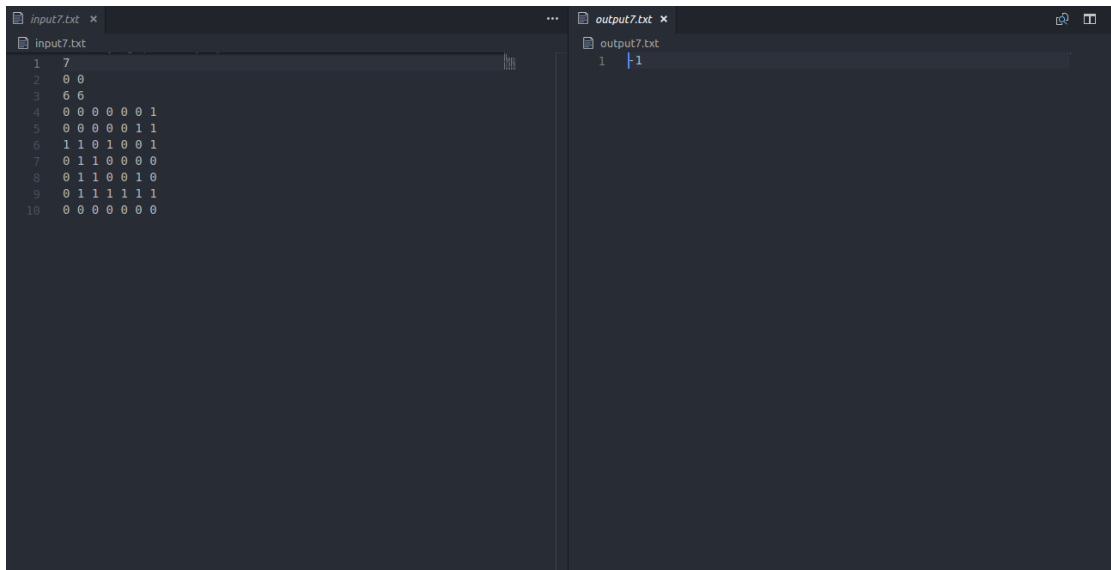


Hình 6: Testcase 200x200.

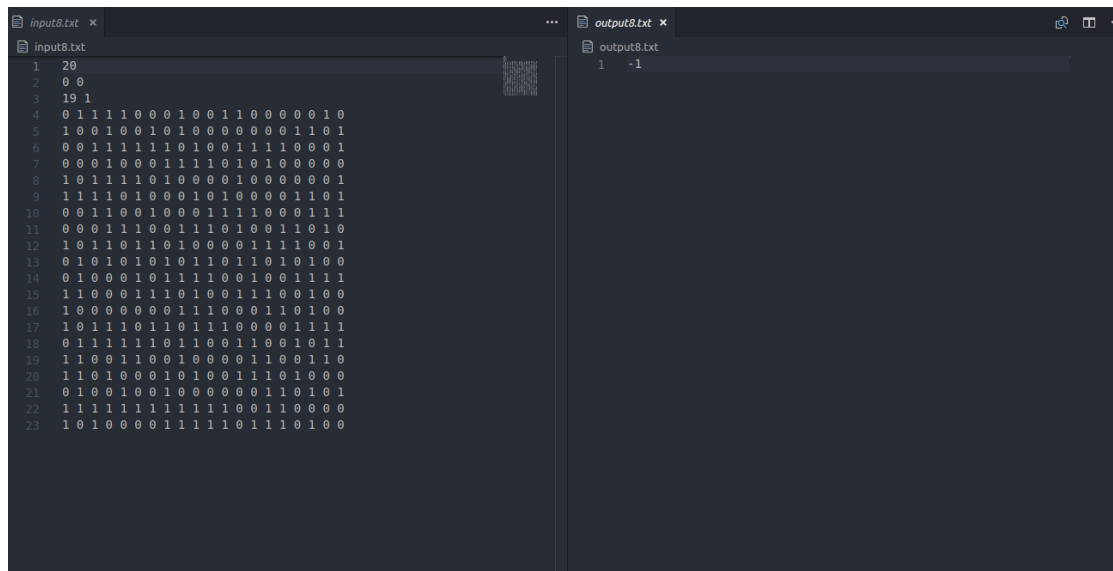


Hình 7: Testcase 50x50.

4.3 Testcase không có đường đi từ start đến goal



Hình 8: Testcase 7x7.



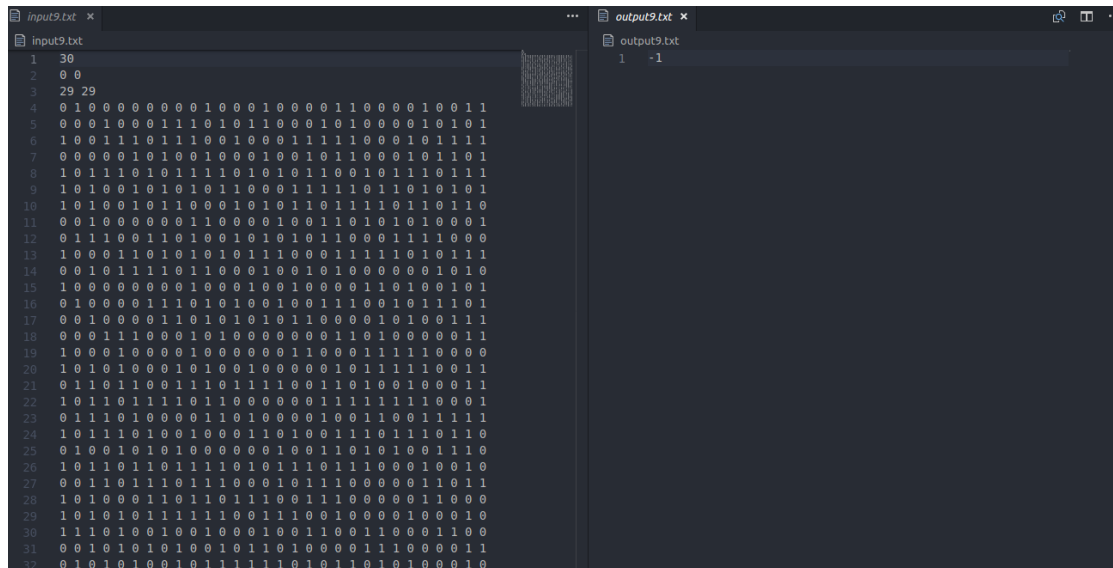
```

input8.txt
1 20
2 0 0
3 19 1
4 0 1 1 1 1 0 0 0 1 0 0 1 1 0 0 0 0 0 1 0
5 1 0 0 1 0 0 1 0 1 0 0 0 0 0 0 0 1 1 0 1
6 0 0 1 1 1 1 1 1 0 1 0 0 1 1 1 1 0 0 0 1
7 0 0 0 1 0 0 0 1 1 1 1 0 1 0 1 0 0 0 0 0
8 1 0 1 1 1 1 0 1 0 0 0 0 1 0 0 0 0 0 0 1
9 1 1 1 1 0 1 0 0 0 1 0 1 0 0 0 0 1 1 0 1
10 0 0 1 1 0 0 1 0 0 0 1 1 1 1 0 0 0 1 1 1
11 0 0 0 1 1 1 0 0 1 1 1 0 1 0 0 1 1 0 1 0
12 1 0 1 1 0 1 1 0 1 0 0 0 0 1 1 1 1 0 0 1
13 0 1 0 1 0 1 0 1 0 1 1 0 1 1 0 1 0 1 0 0
14 0 1 0 0 0 1 0 1 1 1 1 0 0 1 0 0 1 1 1 1
15 1 1 0 0 0 1 1 1 0 1 0 0 1 1 1 0 0 1 0 0
16 1 0 0 0 0 0 0 0 1 1 1 0 0 0 0 1 1 0 1 0 0
17 1 0 1 1 1 0 1 1 0 1 1 1 0 0 0 0 1 1 1 1
18 0 1 1 1 1 1 1 0 1 1 0 0 1 1 0 0 1 0 1 1
19 1 1 0 0 1 1 0 0 1 0 0 0 0 0 1 1 0 0 1 1 0
20 1 1 0 1 0 0 0 1 0 1 0 0 0 1 1 1 0 1 0 0 0
21 0 1 0 0 1 0 0 1 0 0 0 0 0 0 0 1 1 0 1 0 1
22 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 1 1 0 0 0 0
23 1 0 1 0 0 0 0 1 1 1 1 1 0 1 1 1 0 1 0 1 0

output8.txt
1 -1

```

Hình 9: Testcase 20x20.



```

input9.txt
1 30
2 0 0
3 29 29
4 0 1 0 0 0 0 0 0 0 0 1 0 0 0 1 0 0 0 0 1 1 0 0 0 0 1 0 0 1 1
5 0 0 0 1 0 0 0 1 1 1 0 1 0 1 1 0 0 0 1 0 1 0 0 0 0 1 0 1 0 1
6 1 0 0 1 1 1 0 1 1 1 0 0 1 0 0 0 1 1 1 1 0 0 0 1 0 1 1 1 1
7 0 0 0 0 0 1 0 1 0 0 1 0 0 0 1 0 0 1 0 1 1 0 0 0 1 0 1 1 0 1
8 1 0 1 1 1 0 1 0 1 1 1 1 0 1 0 1 0 1 1 0 0 1 0 1 1 1 0 1 1 1
9 1 0 1 0 0 1 0 1 0 1 0 1 1 0 0 0 1 1 1 1 1 0 1 1 0 1 0 1 0 1
10 1 0 1 0 0 1 0 1 1 0 0 0 1 0 1 0 1 1 0 1 1 1 1 0 1 1 0 1 1 0
11 0 0 1 0 0 0 0 0 0 1 1 0 0 0 0 1 0 0 1 1 0 1 0 1 0 1 0 0 0 1
12 0 1 1 1 0 0 1 1 0 1 0 0 1 0 1 0 1 0 1 1 0 0 0 1 1 1 1 0 0 0
13 1 0 0 0 1 1 0 1 0 1 0 1 0 1 1 1 0 0 0 1 1 1 1 1 0 1 0 1 1 1
14 0 0 1 0 1 1 1 1 0 1 1 0 0 0 1 0 0 1 0 1 0 0 0 0 0 0 1 0 1 0
15 1 0 0 0 0 0 0 0 0 1 0 0 0 1 0 0 1 0 0 0 0 1 1 0 1 0 0 1 0 1
16 0 1 0 0 0 0 1 1 1 0 1 0 1 0 0 1 0 0 1 1 1 0 0 1 0 1 1 1 0 1
17 0 0 1 0 0 0 0 1 1 0 1 0 1 0 1 0 1 1 0 0 0 0 1 0 1 0 0 1 1 1
18 0 0 0 1 1 1 0 0 0 1 0 1 0 0 0 0 0 0 0 1 1 0 1 0 0 0 0 0 1 1
19 1 0 0 0 1 0 0 0 0 1 0 0 0 0 0 0 1 1 0 0 0 1 1 1 1 1 0 0 0 0
20 1 0 1 0 1 0 0 0 1 0 1 0 0 1 0 0 0 0 0 1 0 1 1 1 1 1 0 0 1 1
21 0 1 1 0 1 1 0 0 1 1 1 0 1 1 1 1 0 0 1 1 0 1 0 0 0 1 0 0 0 1 1
22 1 0 1 1 0 1 1 1 1 0 1 1 0 0 0 0 0 0 1 1 1 1 1 1 1 1 0 0 0 1
23 0 1 1 1 0 1 0 0 0 0 1 1 0 1 0 0 0 0 1 0 0 1 1 0 0 1 1 1 1 1
24 1 0 1 1 1 0 1 0 0 1 0 0 0 0 1 1 0 1 0 0 1 1 1 0 1 1 1 0 1 1 0
25 0 1 0 0 1 0 1 0 1 0 0 0 0 0 0 0 1 0 0 1 1 0 1 0 1 0 0 1 1 1 0
26 1 0 1 1 0 1 1 0 1 1 1 1 0 1 0 1 1 1 0 1 1 1 0 0 0 0 1 0 0 1 0
27 0 0 1 1 0 1 1 1 0 1 1 1 0 0 0 1 0 1 1 1 0 0 0 0 0 1 1 0 1 1
28 1 0 1 0 0 0 1 1 0 1 1 0 1 1 1 0 0 0 1 1 1 0 0 0 0 0 1 1 0 0 0
29 1 0 1 0 1 0 1 1 1 1 1 1 0 0 0 1 1 0 0 1 0 0 0 0 0 1 0 0 0 1 0
30 1 1 1 0 1 0 0 1 0 0 1 0 0 0 1 0 0 1 1 0 0 1 1 0 0 0 1 1 0 0
31 0 0 1 0 1 0 1 0 1 0 0 1 0 1 1 0 1 0 0 0 0 1 1 1 0 0 0 0 1 1
32 0 1 0 1 0 1 0 0 1 0 1 1 1 1 1 1 0 1 0 1 1 0 1 0 1 0 0 0 0 1 0

```

Hình 10: Testcase 30x30.

5 Đánh giá và tổng kết quá trình

5.1 Mức độ hoàn thành của đề án

STT	Nội dung	Hoàn thành
1	Tìm hiểu kỹ kiến thức, phân chia công việc rõ ràng.	100%
2	Vẽ sơ đồ UML, mô tả cấu trúc dữ liệu, thuật toán cài đặt.	100%
3	Trình bày code sạch sẽ, comment và docstring đầy đủ.	100%
4	Sử dụng testcase với ba đặc trưng khác nhau.	100%
5	Kết quả đúng với yêu cầu đề án.	100%
Mức độ hoàn thành tổng thể của đề án:		100%

5.2 Những vấn đề chưa thực hiện được

Nhóm đã thực hiện đầy đủ những yêu cầu của đề án. Trong quá trình hoàn thành đề án có phát sinh những vấn đề gây khó khăn nhưng nhóm đã giải quyết được.

Tài liệu

- [1] Red Blob Games, *Introduction to A**.
- [2] Rosetta Code, *A* search algorithm*.
- [3] Wikipedia, *A* search algorithm*.
- [4] PGS.TS Lê Hoài Bắc, *Slide bài giảng về Heuristic Search*.
- [5] PyTutorials, *Convert PY to EXE*.