

**SCHOOL OF SCIENCE AND TECHNOLOGY**

**DEPARTMENT OF COMPUTING AND INFORMATION SYSTEMS**

**CSC3206 ARTIFICIAL INTELLIGENCE**

**REPORT**

**STUDENT NAME:** THAM YEE JIN, CHAN WEI QI, CHEONG PUI YIEN

**STUDENT ID:** 18003897, 18102855, 18106914

**PROGRAMME:** BSc (HONS) IN COMPUTER SCIENCE

**YEAR / SEMESTER:** YEAR 3 SEMESTER 2

**INSTRUCTIONS TO CANDIDATE**

**IMPORTANT NOTICE**


The University requires students to adhere to submission deadlines for any form of assessment. Penalties are applied in relation to unauthorized late submission of work.

**Academic Honesty Acknowledgement**

"I THAM YEE JIN, CHAN WEI QI, CHEONG PUI YIEN (student name) verify that this paper contains entirely my own work. I have not consulted with any outside person or materials other than what was specified (an interviewee, for example) in the assignment or the syllabus requirements. Further, I have not copied or inadvertently copied ideas, sentences, or paragraphs from another student. I realize the penalties (*refer to page 16, 5.5, Appendix 2, page 44 of the student handbook diploma and undergraduate programme*) for any kind of copying or collaboration on any assignment."

  
THAM YEE JIN  
17 May 2021

  
CHAN WEI QI  
17 May 2021

  
CHEONG PUI YIEN  
17 May 2021

..... (Student's signature / Date)

# 1 Problem Description

The algorithm aims to create two agents with each based on uninformed and informed search. The search algorithms that have been selected to create these agents are Breadth-First Search for uninformed search and Greedy Best-First Search for informed search. These agents are created in a way that they are able to consume one food at any time with either fixed or dynamic snake length. The snake will be able to continue the game up to at least 15 points when it is in the setting of dynamic snake length. Other than that, the snake with dynamic snake length will also be able to achieve points of at least 10 points when more than one food at any one time is being supplied. The challenges solved were summarized as below:

- Challenge 1: One food at any time with fixed snake length up to at least 15 points
- Challenge 2: One food at any time with dynamic snake length up to at least 10 points
- Challenge 3: Two foods at any time with dynamic snake length up to at least 10 points

# 2 Problem Formulation

There are several variables in this snake game that are considered as parameters. These parameters are recognized as the location of the snake, location of the food, and path taken by the snake to reach the food. The location of the snake is the initial state, also known as the parent node, which corresponds to the length of the snake. When the game is set to have dynamic snake length, the variable that stores the locations of the snake will have an increase in length whenever a food is consumed. In such cases, where the snake length is dynamic, the initial state of a node will be the snake head, which is the first coordinate in the snake locations. Next, the location of the food is considered as the goal where the snake has to take actions to reach the goal. The feasible actions of a node are 'North', 'South', 'East', 'West'. The state of each actions is stored as children node of the node, forming transition models. In order to determine if the given state (current location of the snake at any time) has achieved the goal state (location of the food at any time), a goal test is carried out for the nodes. Since the search algorithm that were used in this assignment are Breadth-First Search and Greedy Best-First Search, the goal test is being carried out differently. For Breadth-First Search, the goal test is performed during node generation, while in Greedy Best-First Search, the goal test is applied when the node is selected for expansion.

Another parameter that is being calculated in Greedy Best-First Search is the estimated cost of the path. The estimated cost is calculated using Manhattan distance, between the coordinate of the snake head at that time and the food's coordinate. Each children node will have its cost saved as "estimated distance" which will be used later for the next action decision. Since all the children (possible actions) are appended into the frontier list, sorting is carried out using the "estimated distance" from each node, to arrange all the nodes in ascending order of the cost. The node that has the minimum cost (estimated distance) will be executed as the next action.

### 3 Search Algorithm Implementation

For uninformed search, Breadth-First Search is implemented. For informed search, Greedy Best-First Search is implemented.

Both algorithms are implemented by first initializing several variables, including solution, search tree, frontier, explored, snake, food, maze size, root node, and the use of a while loop. In every loop, the expansion of the first element in the frontier will be performed. During each expansion, the possible actions, which do not result in the snake biting itself or hitting the wall, are first obtained and the node information in the search tree will be updated. The child nodes of the node will be added to the frontier if the state has not appeared in the frontier and has not been explored. The child nodes will also be added to the search tree. The frontier is arranged based on increasing breadth in Breadth-First Search, while the frontier is arranged based on increasing estimated distance in Greedy Best-First Search. The goal test is performed when the node is generated in Breadth-First Search, whereas the goal test is performed during expansion in Greedy Best-First Search. The loop will stop either when the goal is found or when the frontier is empty. When goal is not found and frontier is empty, it means that the goal cannot be found, and the snake is unable to reach the food. After the while loop is terminated, the solution will be obtained by backtracking each action's parent node. When the goal is not found, a list of all direction ([ 'n', 's', 'e', 'w' ]) will be added to the solution, to end the game by having the snake biting its tail or hitting the wall.

### 3.1 Breadth-First Search

The flowchart for the implementation of Breadth-First Search is shown below.

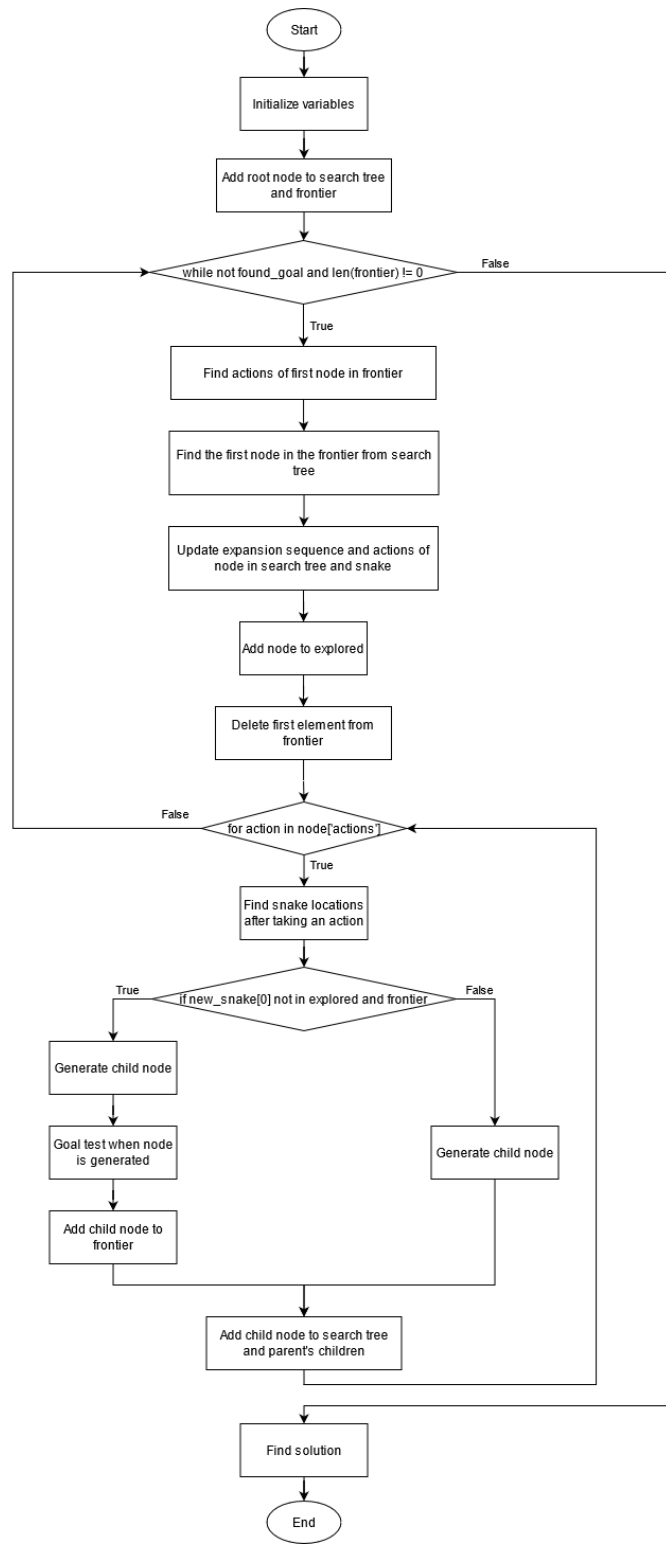


Figure 1: Flowchart for implementation of Breadth-First Search

### 3.2 Greedy Best-First Search

For Greedy Best-First Search, Manhattan distance, which is the distance between the snake head and the food, is used as the estimated distance.

The flowchart for the implementation of Greedy Best-First Search is shown below.

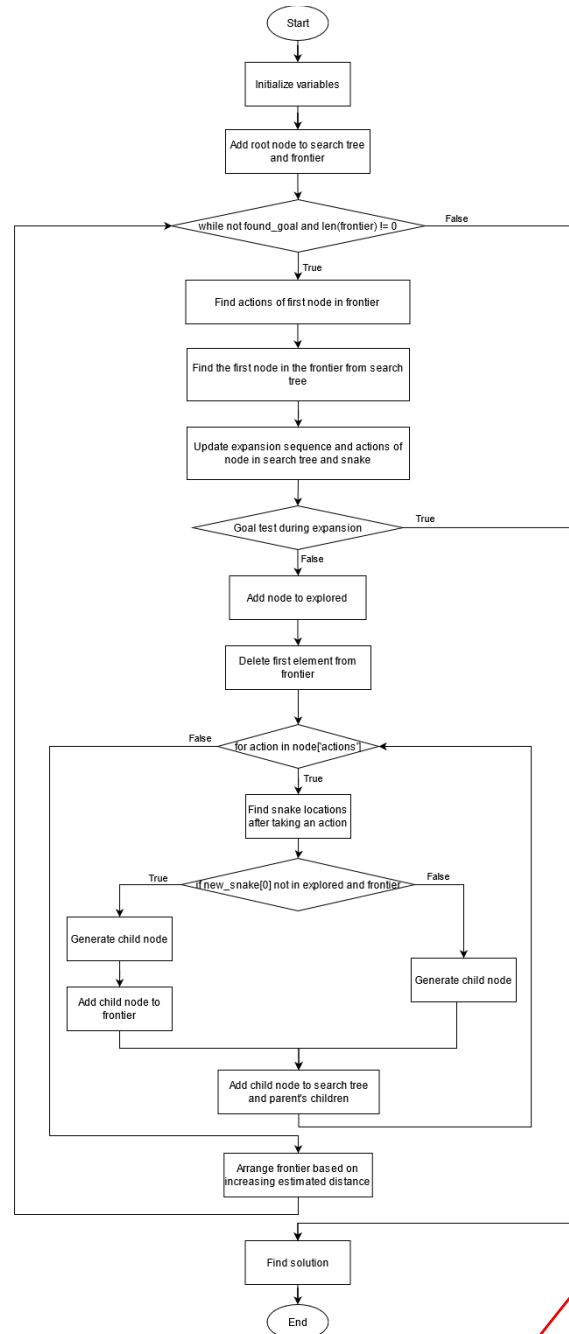


Figure 2: Flowchart for implementation of Greedy Best-First Search

## 4 Results

### 4.1 Challenge 1

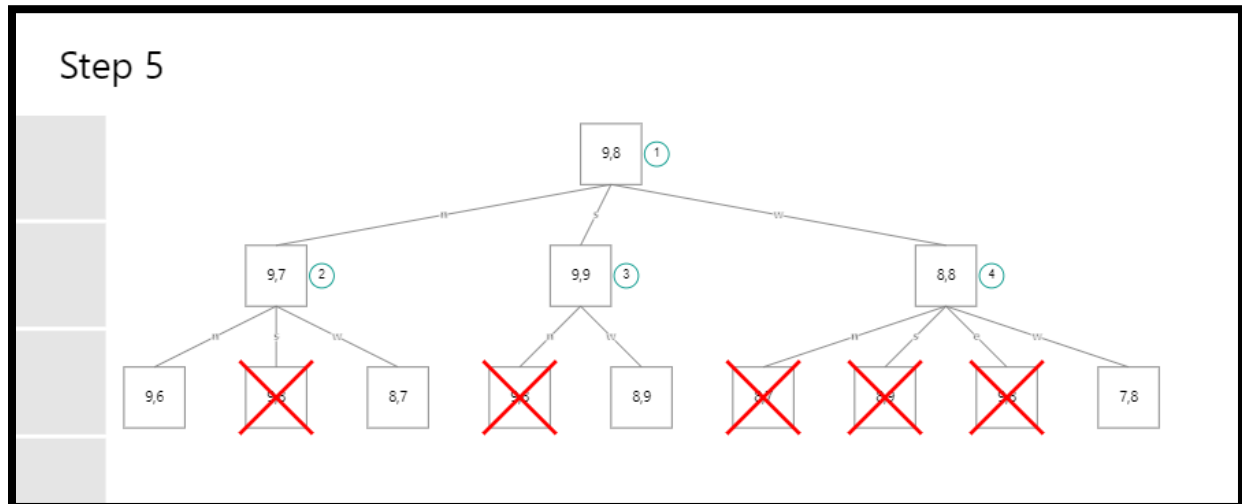


Figure 3.1: Search tree generated for Breadth-First Search Agent with fixed snake length of one

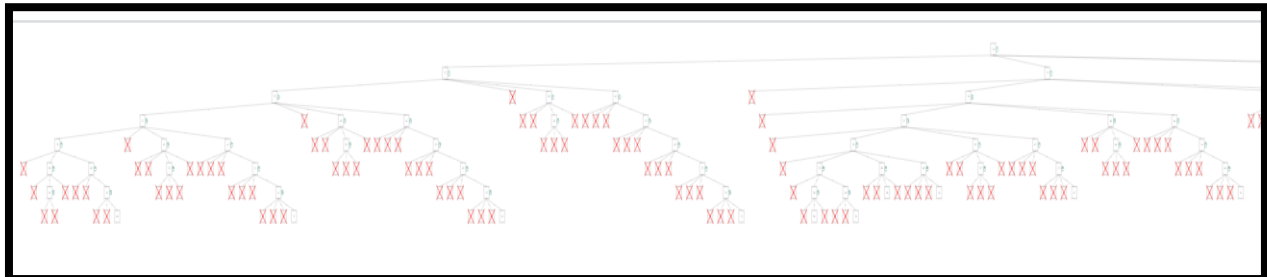


Figure 3.2: Search tree generated for Breadth-First Search Agent with fixed snake length of one

The game is run for 20 rounds with the game set to one food at any time and non-increasing snake length for the Breadth-First Search (BFS) Agent and Greedy Best-First Search (GBFS) Agent. When the starting snake length is set to one, the algorithm is able to achieve at least 15 points in every iteration for both the agents. Since the length of the snake is fixed and the length of snake is one, the game will run infinitely for both the agents as the snake will never hit itself and the maze wall. Both the algorithm performs equally well to hit 15 points. Figure 3.1 shows one of the search trees that is being generated for Breadth-First Search Agent to reach its goal in one of the steps, where the tree expands horizontally before going vertically. Breadth-First Search Agent will expand its nodes by applying the first in first out queue for its frontier and hence the tree showed in Figure 3.2 has a very wide width. The search tree that is being generated by Greedy Best-First Search Agent is shown in Figure 4, where the action that generates the shortest path will be executed next.

Step 6

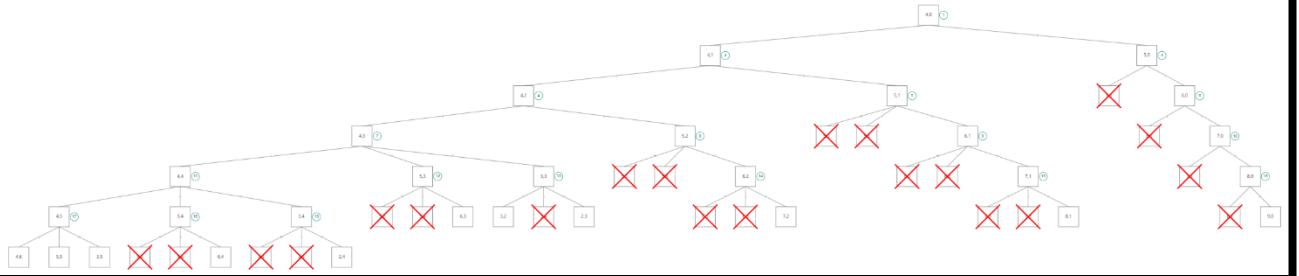


Figure 4: Search tree generated by BFS agent with fixed snake length of six

Hence, both the Breadth-First Search Algorithm and Greedy Best-First Search Algorithm are able to bring the snake to its food and achieve a score of more than 15 points for each iteration when the snake length is 1.

## 4.2 Challenge 2

To evaluate the algorithms, each algorithm was run for 20 rounds with the setting of dynamic snake length and 10 steps per second. The table above shows the scores achieved by the Breadth-First Search agent and Greedy Best-First Search agent in 20 rounds.

Table 1: Scores for Breadth-First Search and Greedy Best-First Search agent

Breadth-First Search Algorithm		Greedy Best-First Search Algorithm	
Round	Score	Round	Score
1	36	1	40
2	29	2	30
3	37	3	27
4	28	4	22
5	41	5	34
6	28	6	31
7	36	7	33
8	37	8	49
9	38	9	24

10	41	10	7
11	53	11	26
12	35	12	28
13	35	13	29
14	40	14	27
15	29	15	36
16	27	16	25
17	30	17	33
18	7	18	19
19	28	19	27
20	36	20	22

For Breadth-First Search, the maximum point scored is 53 (highlighted in yellow), while the minimum point scored is 7 (highlighted in red). For Greedy Best-First Search, the maximum point achieved is 49 (highlighted in yellow), whereas the minimum point achieved is 7 (highlighted in red).

Based on the scores achieved by both agents in 20 rounds, most of the rounds have achieved a score of at least 10 points. However, in round 18 of the Breadth-First Search agent and in round 10 of the Greedy Best-First Search agent, the scores are lower than 10 points. This situation occurs when the food generated is surrounded by the snake or the wall, and this will be further discussed in section 5.2.

### 4.3 Challenge 3

To evaluate the algorithms, each algorithm was run for 20 rounds with the setting of two foods, dynamic snake length, and 10 steps per second. The table below shows the scores obtained in each round for each algorithm.

*Table 2: Scores for Breadth-First Search and Greedy Best-First Search agent*

Breadth-First Search Algorithm	Greedy Best-First Search Algorithm
--------------------------------	------------------------------------



Round	Score	Round	Score
1	20	1	24
2	18	2	24
3	19	3	27
4	40	4	28
5	41	5	23
6	27	6	41
7	34	7	17
8	41	8	24
9	24	9	31
10	40	10	25
11	15	11	29
12	27	12	26
13	28	13	24
14	22	14	41
15	33	15	29
16	32	16	18
17	29	17	27
18	36	18	38
19	13	19	33
20	21	20	28

Based on the scores obtained for 20 rounds, both algorithms are able to achieve at least 10 points with the condition of having 2 foods in a round and dynamic length. In Breadth-First Search, the minimum score obtained was 13 points (highlighted in red) while the maximum score was 41 points (highlighted in yellow). In Greedy Best-First Search, the minimum score obtained was 17 points (highlighted in red) while the maximum score was 41 points (highlighted in yellow).

## 5 Discussion

Breadth-First Search Agent performs slightly slower than Greedy Best-First Search Agent for all the challenges. This is due to the nature of Breadth-First Search in searching the goal node using the First-In First-Out queue system whereby the children that are first appended into the frontier must be explored before going into the descendants' level to search for goal node. Once a node is generated, the goal test is carried. If the node is not the goal, the next node to explore will be the one that is situated first in the queue. Hence, the search tree for Breadth-First Search might go very wide, increasing time taken to find the path. On the other hand, Greedy Best-First Search Agent performs slightly faster than Breadth-First Search Agent as the algorithm will only expand the path that has the lowest Manhattan distance. Greedy Best-First Search will expand the current node before choosing the next path, in other words, goal test for Greedy Best-First Search is being carried out during node expansion instead of node generation.

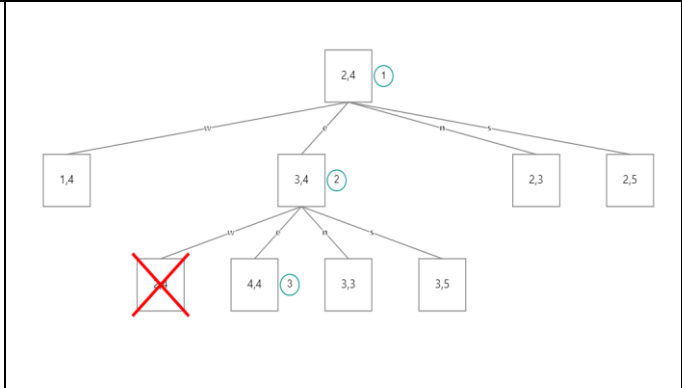
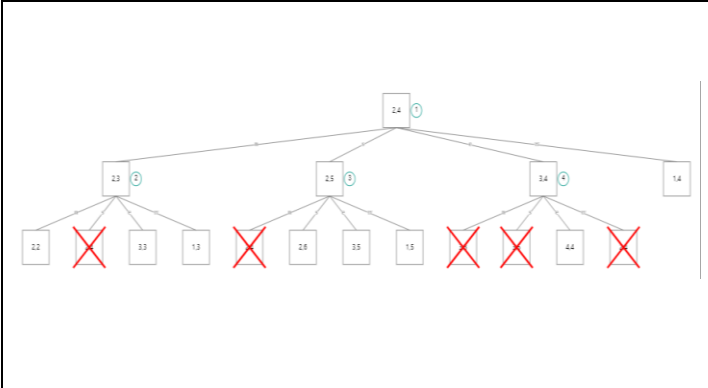
### 5.1 Snake with Fixed Length

The search trees generated for both the Breadth-First Search and Greedy Best-First Search algorithm with fixed snake length are shown in Table 3. The location of the snake and food are set to be the same to generate the search tree shown in the table. The search tree for BFS will always expand wider than search tree from GBFS.

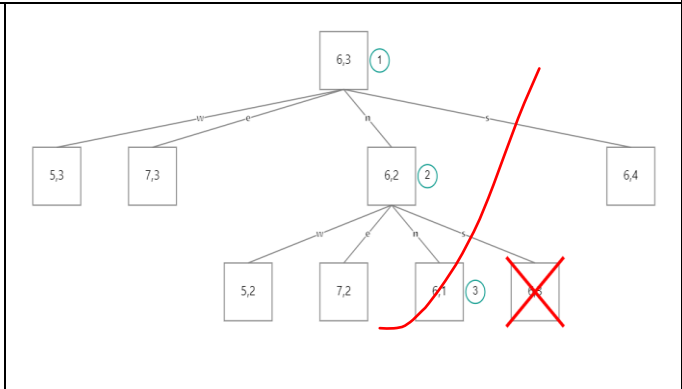
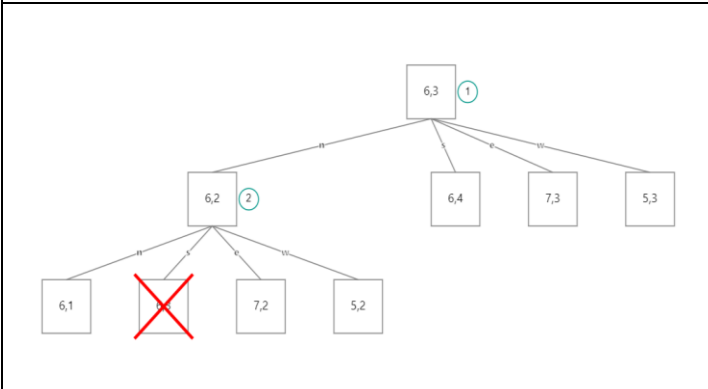
Table 3: Search Tree Comparison of BFS and GFS with fixed snake length

Breadth-First Search (Uninformed Search)	Greedy Best- First Search (Informed Search)
Snake location: [0,5] and Food location: [1,6]	

Snake location: [2,4] and Food location: [4,4]
--



Snake location: [6,3] and Food location: [6,1]
--



er hit itself or hi

## 5.2 Snake with Dynamic Length

To compare Breadth-First Search and Greedy Best-First Search algorithms for dynamic snake length, different snake locations and food locations are tested. Below shows comparison between algorithms through their generated search tree using different pairs of locations of snake and food.

Table 4: Comparisons of search trees between Breadth-First Search and Greedy Best-First Search agents using different pairs of snake and food locations

Breadth-First Search (Uninformed Search)	Greedy Best-First Search (Informed Search)
Snake location: [[5,0], [4,0], [3,0]] and Food location: [4,1]	
Snake location: [[4,4], [3,4], [2,4]] and Food location: [5,5]	
Snake location: [[1,2], [0,2]] and Food location: [2,2]	

However, as mentioned in the result section, there is a possibility that both algorithms do not reach at least 10 points. An example of the situation is shown in Figure 5 below. The food generated is surrounded by the snake. The solution returned by both Breadth-First Search and Greedy Best-First Search algorithm is ['w', 'n']. Although this is the solution that takes the least steps to reach the food, it leads to dead end. In other words, the snake has no way to move to the next food regardless of where it is placed. Therefore, lesser points are scored.

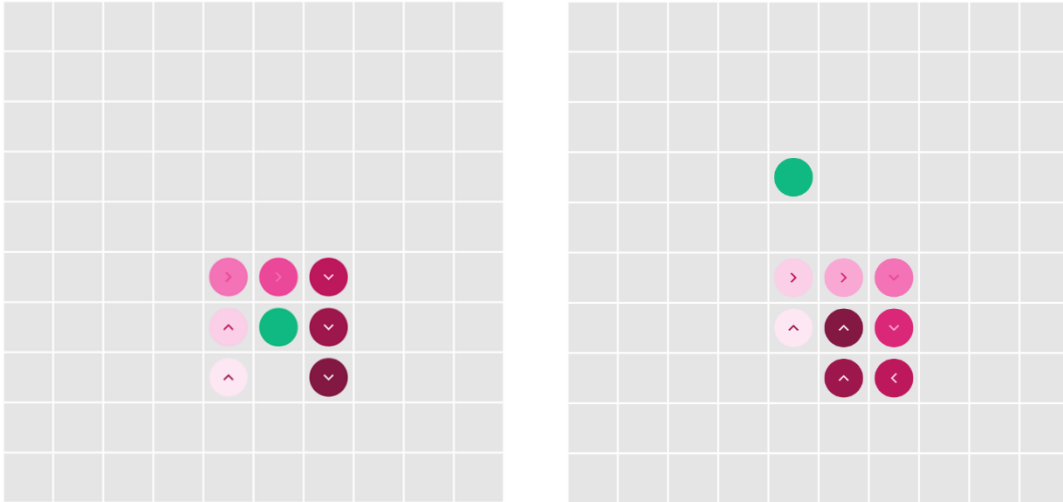


Figure 5: Food generated, and solution returned