COSE361 인공지능 실습 1

제출기한:2021 년 4월 1일 목요일 오후 1:59PM 까지

(2:00PM 제출시 지각 제출로 간주)

학번: 2016160311 이름: 이재윤

- 1. Euclidean distance 을 계산하는 함수 heuristic euclidean 를 생성하시오.
- 2. greedy_best_first_search, a_star_search 의 heuristic 함수를 1 번에서 만든 함수로 교체하시오.
- 3. grid3 그리드에 dijkstra_search, greedy_best_first_search, a_star_search 를 적용하고 draw_grid 함수를 이용하여 그리드를 출력하시오. (start3, goal3 사용)
- 4. 위 코드에서 생성된 각 path 의 스크린샷을 첨부하시오.
 - dijkstra_search

2)	gree	dy_	_bes	st_f	ir	st_s	ear	rch							
							•		•	•					
					0	0	0	0	0	0	0	0	0	0	Z
					0	###	###	###	###	###	###	###	####		
					0	0	0	0	0	0	0		###		
											0		###		
										0	0		###		
									0	0			###		
								0	0				###		
							0	0					###		
						0	0						###		
		0	0	0	0	0							###		
	Α	0	###	###	###		###	###	###	###	###	###	###		
	~~~	~~		~~~	~~		~~~	~~~	~~~	~~~	~~~	~~		·~~	~~~
3)															
٠,	a_st	ar	_sea	irch	1										
<i>J</i> )	a_st	ar_	_sea	rch	) 										
<i>J</i> )	a_st 	ar_ 	_sea 	rch 	) 										
,	a_st 	ar_ 	_sea 	rch 	•		· ·	· ·	· ·	· ·	· ·		· · ·		 · ·
′)	a_st	ar_ 	_sea 	arch	@		· · @	· · @	· · @	· · @	@		@		Z
	a_st	ar_ 	_sea	arch									####		Z
<i>J</i> ,	a_st	ar_	_sea	erch	0 0								###	@	Z
,	a_st	ar_	_sea	erch	0 0 0								### ### ###	@	. Z
<i>J</i> ,	a_st	ar_ 	_sea	arch									### ### ### ###		 Z
	a_st	ar	_sea	arch									### ### ### ###		. Z
	a_st	ar_ 	_sea	erch 									### ### ### ###		
	a_st	ar_ 	_sea										### ### ### ###		. Z
	a_st	ar_  - - - - - - - - - - - - - - - - -											**** *** *** *** *** ***		 Z
													### ### ### ### ###		
						### - - - - -	### - - - - -	****	### - - - - -	### - - - - -	### - - - - -	****	### ### ### ### ### ###		 · · · · · ·
						### - - - - -	### - - - - -	****	### - - - - -	### - - - - -	### - - - - -	****	***** **** **** **** **** **** ***		·
						### - - - - -	### - - - - -	****	### - - - - -	### - - - - -	### - - - - -	****	***** **** **** **** **** **** ***		 ·
						### - - - - -	### - - - - -	****	### - - - - -	### - - - - -	### - - - - -	****	***** **** **** **** **** **** ***		

5. implementation.py 파일의 grid3 그리드 코드를 복사하여 grid4 를 생성하시오. grid4 는 디귿자인 벽 중 한군데에 구멍이 뚫려 있어야 합니다. (구멍 위치 – 177 인덱스)

6.	gri	d4 ⊐	니기.	드에	di	jks	tra	_se	arc	h,	gre	edy	_b	est_	fi	rst_	searc	h,		
	. c	tan	503	nch	르	- 저ડ	로하	_ 	ln au	ıar	ni d	하스	_ _르	이윤	삵	വ –	- 1리드를	르 추	려하기	U O
																~ı —	ㅗ니 — E	2 2 .	7 917	۱۳.
	(het	ıristic	: 임	수 [	Eucl	ide	an, s	stai	rt3	, go	oal:	3 Д	내년	로 사	용)					
7.	위 =	코드0	ᅦ서	생성	성된	각	pat	h º	스.	크린	l샷{	을 침	부	하시.	오.					
		dijk					•													
	1)			a_3																
						0	0	0	0	0	0	0	0	0	0	Z				
						0	###	###	###	###	###	###	###	####						
						0								###						
						0								###						
						0								###						
						0								###						
					0	0							Ī	###						
				0	0	e.					·	Ċ		###		Ċ				
		•	0	0	(u	•	•	•	•	•	•	•	•	###		•				
				Įu	•	•	•	•	•	•	•	•	•	###	•	•				
		0	0												•	•				
		Α	•	###	###	###	+###	###	###	###	###	****	###	####	•	•				
		•	•	•	•	•	•	•	•	•	•	•	•	•	•	•				
		•	٠	٠	•	•	•	٠	٠	٠	•	•	•	•	•	•				
		~~~	~~~		~~~	~~~		~~~		NNN		.~~	~~	~~~	~~~					
	2)	gree	dy_	bes	t_f	irs	st_s	ear	rch											
		•	•	•	•	•	•	•	•	•	•	•	•	•	•	•				
		•	•	•	•	•	•	•	•	•	•	•	•	•						
		•	•	•	•	٠									0	Z				
		•	•	٠	•	٠	###	###	###	###	###	###		###	0	•				
		•	•	٠	•	٠	•	•	٠	•	٠	•	•	###	0	•				
														###	0					
											0	0	0	###	0					
										0	0		0	###	0					
									0	0			0	###	0					
								0	0				0	###	0					
							0	0					0	###	0					
			0	0	0	0	0						0	0	0					
		Α	-	-	-	-	-	###	###	###	###	###	_	###	-					
																_				

3) a_star_search

				0	0	0	0	0	0	0	0	0	0	Z
				0	###	###	###	###	###	###	###	###		
				0								###		
				0								###		
				0								###		
				0								###		
			0	0								###		
		0	0									###		
	0	0										###		
0	0													
Α		###	###	###	###	###	###	###	###	###	###	###		
·~~	~~		~~~	~~~	~~~	~~~	~~~	~~~	~~~	~~~	~~~	.~~~	·~~	~~~

- 8. 위에서 밝혀진 Dijkstra's Algorithm, Greedy Best-First Search, A* Search 을 통해 찾아진 path 들을 optimality 관점으로 비교 및 분석하시오.
 - → Dijkstra's Algorithm 과 A* Search 는 optimality 가 보장되므로 주어진 조건 내에서 최단거리의 path 를 찾아내어 나타내고 있지만 Greedy Best-First Search 의 경우 optimality 가 보장되지 않기 때문에 시작지점에서 목표지점까지 가는 path 가 최소 거리가 되지 않는 것을 확인할 수 있다. 주어진 조건에 따르면 path 의 cost 가 전부 동일하므로 Dijkstra's Algorithm 으로 구한 경로가 A* Search 로 얻은 경로와 동일하게 나오는 것을 확인할 수 있다. 벽에 구멍을 뚫었을 때, Greedy Best-First Search 의 경로는 구멍을 뚫기 전보다는 더 빠른 경로를 찾아내었다. 하지만 Dijkstra's Algorithm 과 A* Search 의 경로는 구멍을 뚫기 전과 후의 가장 optimal 한 경로가 동일하기 때문에 경로가 변하지 않았음을 확인할 수 있다.