Introduction to Artificial Intelligence.
Assignment One.

Polina Turishcheva. Group 3.

Assumptions:

- Start point is always (0,0)
- The field edges are (0,0), (0, 9), (9, 0), (9,9) - inclusively
- Diagonal throws are valid
- Only one throw per search run is valid
- We were not asked to create successful algorithms it terms of correct answer or time. This assumption is important for random search because according to our TA random walk does not analyze adjacent cells, hence, it can step on an ork
- For random search if the player meets org - player dies
- For random search if he/she tries to go out of map bound - player dies
- Random walk makes maximum 100 steps during a search
- All algorithms search for the 1st valid path. This is done because to find all passes on 10 on 10 map is extremely time consuming. In fact this is what happens on the empty map. Time necessary for such test is available further. This was also allowed by Prof. Brown.
- The code was checked on SWI prolog and with its system on local machine.
- No input files are read. The map conditions should be written in executed file.
- Link to github with code.

Description of the 3rd algorithm:
My third algorithm is an improved backtracking. The idea is that I assume that we know the point of start, which is (0,0). Hence, we can investigate the map only in 2 directions: up and right. This reduces the number of step twice, comparing with backtracking, which should check all possible directions: up, rights, down, left. This is kind of heuristic. However, with such moving scheme there is a tricky case - Figure 1. In case of figure 1-a any of 3 algorithms will perform successfully only one step - up-right diagonal throw. But for case 1-b and 1-c we have opportunities to move up or right but for such moving scheme some part of the field will stay unvisited, hence, if touchdown will be there - it will not find it. *In other languages we could use a global list of visited cells and than solve it like this: add the second part for our algorithm. If during up-right search the touchdown was not found, hence, we start moving down-left and if the cell was not visited during the first period(up and right exploration), it is not in the global list,hence, we start backtracking from it. To understand, why it is important look at picture 2.* Unfortunately, prolog design does not allow us to have global variables, therefore, in case the first up and right diagonal walk has found no paths, we start backtracking for the initial position. The reasons for this solution to be still improving will be discussed in Part One.

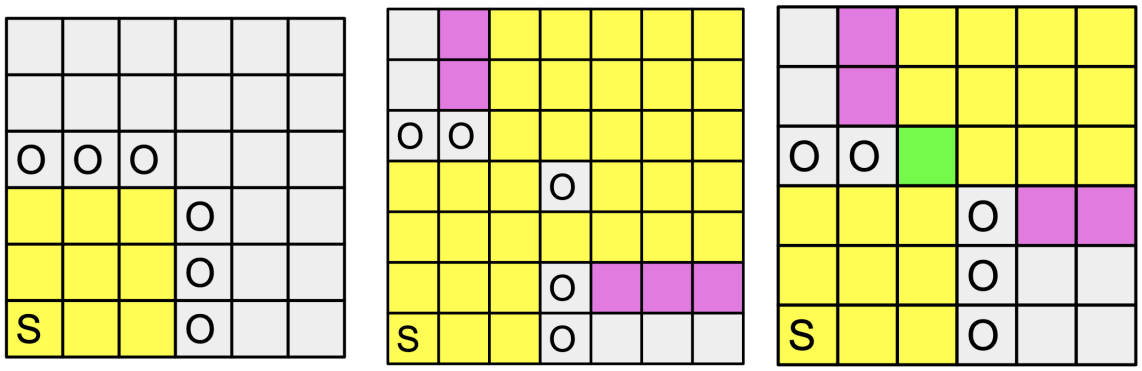


Figure 1: S- start, O -Ork.
Yellow cells - the algorithm will visit them.
Pink cells - the algorithm will visit them only if touchdown is there.
Green cell - the only possibility to go out of orks' angle, this point limits the areas that will be visited.

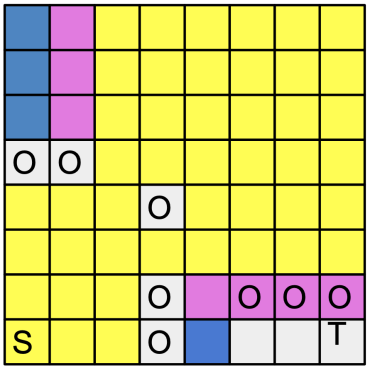


Figure 2: Blue cells - will be additionally visited during left and down iteration. Hence, from such cells we should start backtracking, otherwise, we will not be able to find touchdown in some cases. The example is in the picture.There is no use start backtracking from every cell because backtracking is very time consuming, only from unvisited before cells. If from backtracking we return to the cells from the first(right and up) iteration, we stop this backtracking branch.

Part One First algorithm is on average works much faster than backtracking because it always scans more than half of the map in a very short period of time. The chances that the touchdown point will be found during the up and right search is equal to $\frac{\text{yellow cells} + \text{pink cells} + \text{orks on boundary}}{\text{all cells in the map}}$.
The worst probability to find touchdown during up and right search is 36%. This happens when orks' angle appear is one line along the whole axe - figure 3. This angle has low probability to appear $=2 * 0.25^9 \approx 7.6 * 10^{-6}$ because each cell has 4 options - ork, nothing, human, touchdown, therefore, we can assume that all of them have equal probability to appear in this cell, therefore, probability for ork is 0.25, and there should be 9 such cells simultaneously on the fixed positions, 2 become from one horizontal and one vertical row of 9 orks. Hence, on average, our chances to find touchdown are much higher and the first iteration is so cheap to perform (comparing with backtrackings hard-to-solve cases), that it definitely worth it. The worst case for all three algorithms is when we have absolutely empty map because they should visit each cell. For the third algorithm it is even worse as it will visit all cells more times - during up and right search and during backtracking search. I have tested all three algorithms on absolutely empty 5 on 5, 6 on 6 maps (7 on 7 empty map was too long to execute - after several hours I have interrupted the execution- in order to save time as empty 10 on 10 map is extremely long to execute) - the difference between 3rd approach and backtracking seems to be neglectable- figure 4,5. Moreover, is the touchdown point will be found during up and right search and there are no humans on the map - the result from 3rd algorithm will be the shortest possible as it makes more rational steps than backtracking. Even with humans it will most probably be better than for the backtracking,
The first and second tests were intended to check the time improvement of the 3rd method over backtracking.

```
%Test one
o(0,3).
o(1,3).
o(3,0).
o(3,1).
o(3,2).
t(4,4).
```

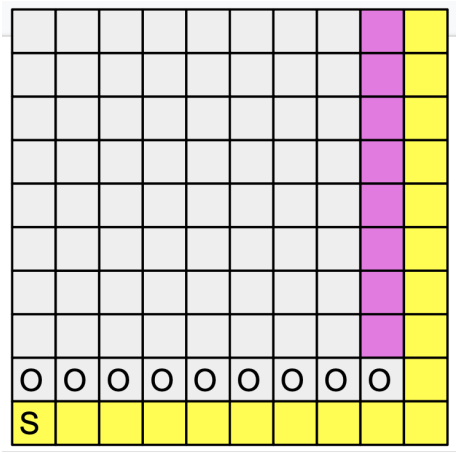


Figure 3: Worst case for 3rd algorithm heuristic.

```
?- main.
Backtracking  No solution
% 85,100,281 inferences, 7.522 CPU in 7.526 seconds (100% CPU, 11314196 Lips)
Random search  No solution
% 119 inferences, 0.001 CPU in 0.001 seconds (83% CPU, 224105 Lips)
Improved backtracking  No solution
% 85,482,854 inferences, 7.530 CPU in 7.533 seconds (100% CPU, 11352801 Lips)
```

Figure 4: Empty 5 on 5 map. Difference between backtracking and third method is 7ms, when execution takes more than 7.5 seconds.

```
Backtracking  No solution
% 20,673,344,525 inferences, 1993.505 CPU in 1999.864 seconds (100% CPU, 10370350 Lips)
Random search  No solution
% 133 inferences, 0.000 CPU in 0.000 seconds (87% CPU, 306452 Lips)
Improved backtracking  No solution
% 20,726,814,104 inferences, 2008.347 CPU in 2014.941 seconds (100% CPU, 10320333 Lips)
true.
```

Figure 5: Empty 6 on 6 map. Difference between backtracking and third method is about 7s, when execution takes more than 2000 seconds.

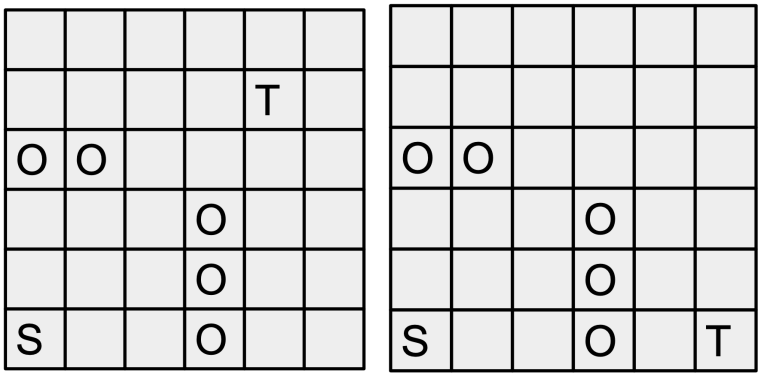
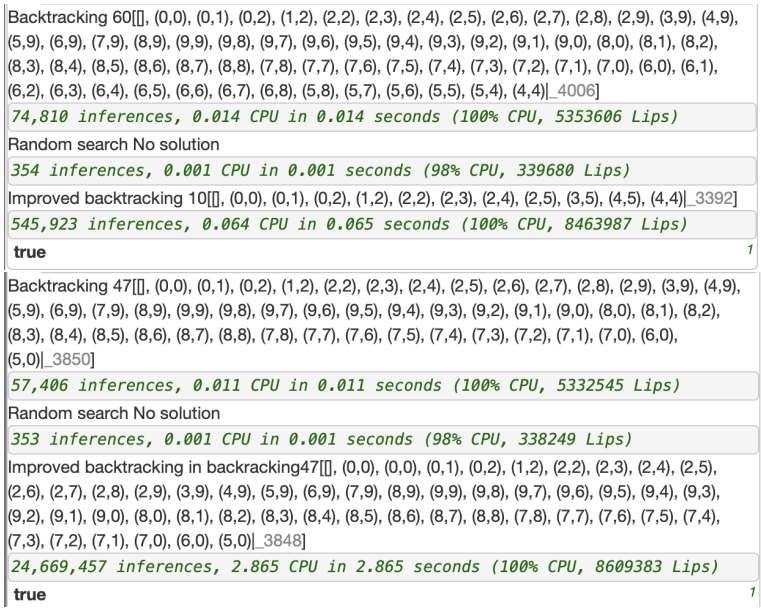


Figure 6: In the first test the touchdown should be found during up and right search, in the second tests it will be found only during backtracking stage.



Test 1

Test 2

Figure 7: Results for test 1(up) and test 2 (down).In test one algorithm 3 reaches the touchdown during the up and right stage, hence it provides much more rational path. However, as algorithm is also kind of 2-directional backtracking it starts recursion which is rather time consuming. During the 2nd test, when the 3rd algorithm will go to backtracking stage this is even more obvious. Note that the path output looks like [[], *actual path* | *_something*] , the empty list in the beginning and everything after | can be neglected, it is shown just because I have used difference list for my inner implementation.

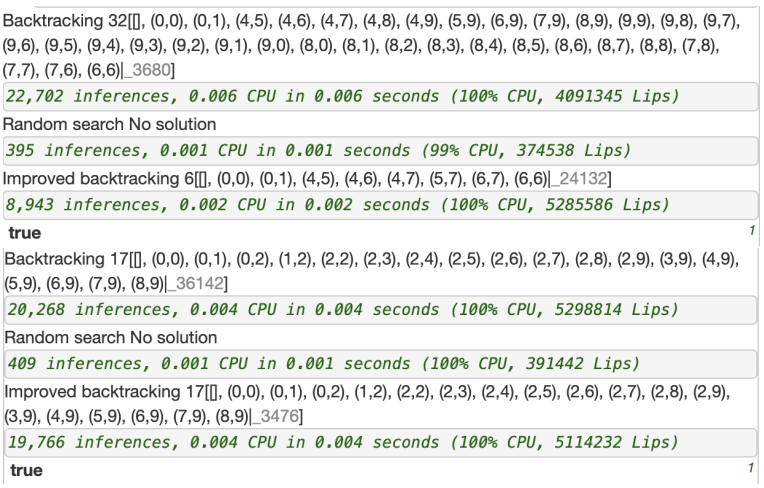


Figure 8: Tests 3 and 4 have same orks positions as for tests 1 and 2, not other touchdown positions and also in test 3 a human was added. Results for test 3(up) and test 4(down). Time difference is noticeable, it is 3 times less for the 3rd algorithm.For the 4th test, when 3rd algorithm when on the backtracking stage the time difference is not noticeable. Both tests were run on 10 on 10 map.

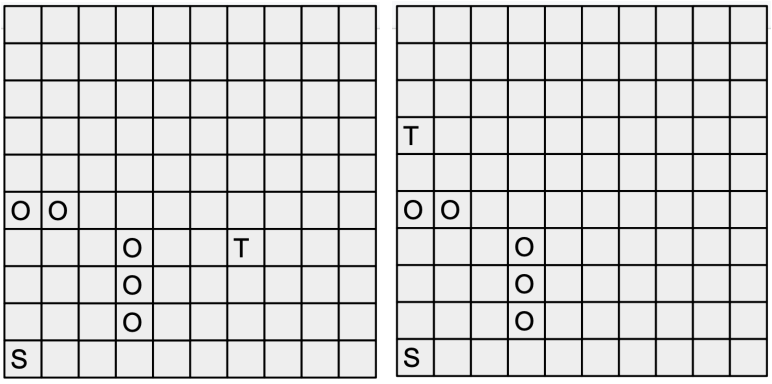
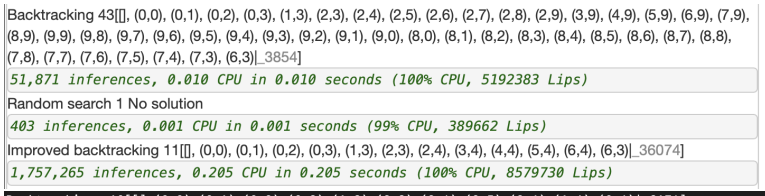
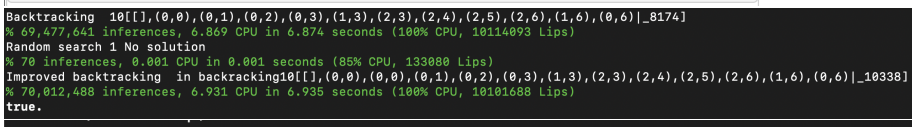


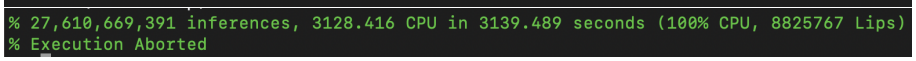
Figure 9: Maps for Test 6 and Test 7.



Test6



Test7-1



Test7-2

Figure 10: Test 6 was run on 10 on 10 map.Test 7 was run on 7 on 7 map. Because of time reasons.Test 7 on 10 on 10 map has not terminated in an hour for backracking on my machine.This is also reasoning for not looking for the best path.

```
%Test two
o(0,3).
o(1,3).
o(3,0).
o(3,1).
o(3,2).
t(5,0).
%Test three
o(0,3).
o(1,3).
o(3,0).
o(3,1).
o(3,2).
h(4,5).
t(6,6).
%Test four
o(0,3).
o(1,3).
o(3,0).
o(3,1).
o(3,2).
t(8,9).
%test five
h(3,3).
```

```
t(4,4).
o(0,-1).
o(0,3).
o(1,3).
o(2,3).
o(3,2).
o(3,1).
o(3,0).
%test six
o(0,4).
o(1,4).
o(3,3).
o(3,1).
o(3,2).
t(6,3).
%test seven
o(0,4).
o(1,4).
o(3,3).
o(3,1).
o(3,2).
t(6,3).
%Test eight
t(0,2).
%Test nine
t(5,5).
```

One can notice that random search has not succeeded ever. I have run random walk on the empty 10 on 10 map and analyzed the paths it has chosen in order to try to find the type of maps that can be solved with it. In the output A is value of current randomly generated number. For A - 0 is up throw, 1 is right throw, 2 is down throw, 3 is left throw, 4 is up-left throw, 5 is up-right throw, 6 is down-right throw, 7 is down-left throw, 8 is up step, 9 is right step, 10 is down step, 11 is left step.
Paths from random walk on the empty map:

1. X=0 Y=0 A=5 No solution
2. X=0 Y=0 A=6 No solution
3. X=0 Y=0 A=5 No solution
4. X=0 Y=0 A=6 No solution

5. X=0 Y=0 A=11 X=1 Y=0 A=9 X=2 Y=0 A=1 No solution
6. X=0 Y=0 A=4 No solution
7. X=0 Y=0 A=0 No solution
8. X=0 Y=0 A=3 No solution
9. X=0 Y=0 A=0 No solution
10. X=0 Y=0 A=8 X=0 Y=1 A=6 No solution
11. X=0 Y=0 A=0 No solution
12. X=0 Y=0 A=8 X=0 Y=1 A=0 No solution
13. X=0 Y=0 A=0 No solution
14. X=0 Y=0 A=1 No solution
15. X=0 Y=0 A=0 No solution
16. X=0 Y=0 A=9 X=1 Y=0 A=8 X=1 Y=1 A=7 No solution
17. X=0 Y=0 A=1 No solution
18. X=0 Y=0 A=11 X=1 Y=0 A=10 X=1 Y=-1 No solution
19. X=0 Y=0 A=3 No solution
20. X=0 Y=0 A=3 No solution
21. X=0 Y=0 A=1 No solution
22. X=0 Y=0 A=0 No solution
23. X=0 Y=0 A=8 X=0 Y=1 A=3 No solution
24. X=0 Y=0 A=8 X=0 Y=1 A=11 X=2 Y=1 A=8 X=2 Y=2 A=6 No solution
25. X=0 Y=0 A=8 X=0 Y=1 A=7 No solution
26. X=0 Y=0 A=9 X=1 Y=0 A=3 No solution
27. X=0 Y=0 A=10 X=0 Y=-1 No solution
28. X=0 Y=0 A=6 No solution
29. X=0 Y=0 A=5 No solution
30. X=0 Y=0 A=10 X=0 Y=-1 No solution
31. X=0 Y=0 A=9 X=1 Y=0 A=1 No solution
32. X=0 Y=0 A=3 No solution
33. X=0 Y=0 A=1 No solution
34. X=0 Y=0 A=8 X=0 Y=1 A=0 No solution
35. X=0 Y=0 A=6 No solution
36. X=0 Y=0 A=5 No solution
37. X=0 Y=0 A=6 No solution
38. X=0 Y=0 A=7 No solution

39. X=0 Y=0 A=6 No solution

40. X=0 Y=0 A=2 No solution

41. X=0 Y=0 A=4 No solution

42. X=0 Y=0 A=2 No solution

43. X=0 Y=0 A=2 No solution

44. X=0 Y=0 A=8 X=0 Y=1 A=11 X=2 Y=1 A=5 No solution

45. X=0 Y=0 A=0 No solution

46. X=0 Y=0 A=9 X=1 Y=0 A=7 No solution

47. X=0 Y=0 A=5 No solution

48. X=0 Y=0 A=7 No solution

49. X=0 Y=0 A=2 No solution

50. X=0 Y=0 A=3 No solution

51. X=0 Y=0 A=10 X=0 Y=-1 No solution

52. X=0 Y=0 A=3 No solution

53. X=0 Y=0 A=9 X=1 Y=0 A=3 No solution

54. X=0 Y=0 A=9 X=1 Y=0 A=2 No solution

55. X=0 Y=0 A=2 No solution

56. X=0 Y=0 A=3 No solution

57. X=0 Y=0 A=3 No solution

58. X=0 Y=0 A=4 No solution

59. X=0 Y=0 A=5 No solution

60. X=0 Y=0 A=9 X=1 Y=0 A=2 No solution

61. X=0 Y=0 A=1 No solution

62. X=0 Y=0 A=2 No solution

63. X=0 Y=0 A=4 No solution

64. X=0 Y=0 A=11 X=1 Y=0 A=2 No solution

65. X=0 Y=0 A=5 No solution

66. X=0 Y=0 A=8 X=0 Y=1 A=2 No solution

67. X=0 Y=0 A=6 No solution

68. X=0 Y=0 A=2 No solution

69. X=0 Y=0 A=9 X=1 Y=0 A=0 No solution

70. X=0 Y=0 A=0 No solution

71. X=0 Y=0 A=4 No solution

72. X=0 Y=0 A=0 No solution

73. X=0 Y=0 A=4 No solution

74. X=0 Y=0 A=9 X=1 Y=0 A=3 No solution

75. X=0 Y=0 A=2 No solution

76. X=0 Y=0 A=0 No solution

77. X=0 Y=0 A=0 No solution

78. X=0 Y=0 A=10 X=0 Y=-1 No solution

79. X=0 Y=0 A=11 X=1 Y=0 A=6 No solution

80. X=0 Y=0 A=2 No solution

81. X=0 Y=0 A=1 No solution

82. X=0 Y=0 A=8 X=0 Y=1 A=0 No solution

83. X=0 Y=0 A=9 X=1 Y=0 A=1 No solution

84. X=0 Y=0 A=7 No solution

85. X=0 Y=0 A=5 No solution

86. X=0 Y=0 A=4 No solution

87. X=0 Y=0 A=2 No solution

88. X=0 Y=0 A=9 X=1 Y=0 A=6 No solution

89. X=0 Y=0 A=3 No solution

90. X=0 Y=0 A=1 No solution

91. X=0 Y=0 A=3 No solution

92. X=0 Y=0 A=4 No solution

93. X=0 Y=0 A=6 No solution

94. X=0 Y=0 A=8 X=0 Y=1 A=2 No solution

95. X=0 Y=0 A=6 No solution

96. X=0 Y=0 A=3 No solution

97. X=0 Y=0 A=5 No solution

98. X=0 Y=0 A=5 No solution

99. X=0 Y=0 A=4 No solution

100. X=0 Y=0 A=5 No solution

21 of the paths had length 2, 3 had length 3 and one had length 4. At total here were 141 choosing of random number, here is statistics about it:

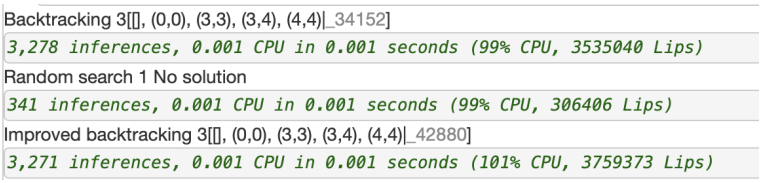


Figure 11: Results of test 5 for all 3 algorithms.Proves that random search is extremely inefficient.

Number	Direction	Times	Percentage
0	Up Throw	15	10.6
1	Right Throw	21	14.9
2	Down Throw	15	10.6
3	Left Throw	15	10.6
4	Up-Right Throw	9	6.4
5	Up-Left Throw	12	8.5
6	Down-Right Throw	13	9.2
7	Down-Left Throw	6	4.3
8	Up Move	12	8.5
9	Right Move	12	8.5
10	Down Move	5	3.5
11	Left Move	6	4.3

From here we see that the choose of the move is not purely uniformly distributed (because random is not pure random), however, each choice has competitive chances. The most popular values are throws, hence, maps with people catching ball from right or up sides may be successful for the random search. So lets try. The result of 400 runs of the test 5 are written down. The average execution time is less 0.001s. However, *only 3% of runs succeeded to find the touchdown and only 1.5 of them found the most optimal path.* Hence, we conclude that random search is useless. It may be crucially improved if it would analyze its surroundings but in the assumptions it is written that we were asked not to do it. *Scroll for page 25 to see the results.*

Note that for some outputs X or Y are equal to -1, this happened because print was put before checking if the cell is in bound.

1. X=0 Y=0 A=3 No solution
2. X=0 Y=0 A=10 No solution
3. X=0 Y=0 A=7 No solution
4. X=0 Y=0 A=6 No solution
5. X=0 Y=0 A=10 No solution

6. X=0 Y=0 A=6 No solution

7. X=0 Y=0 A=4 No solution

8. X=0 Y=0 A=6 No solution

9. X=0 Y=0 A=11 X=1 Y=0 A=9 X=2 Y=0 A=11 X=1 Y=0 A=10 No solution

10. X=0 Y=0 A=11 X=1 Y=0 A=11 X=1 Y=0 A=5 No solution

11. X=0 Y=0 A=10 No solution

12. X=0 Y=0 A=11 X=1 Y=0 A=10 No solution

13. X=0 Y=0 A=10 No solution

14. X=0 Y=0 A=10 No solution

15. X=0 Y=0 A=10 No solution

16. X=0 Y=0 A=8 X=0 Y=1 A=8 X=0 Y=2 A=11 X=3 Y=2 No solution

17. X=0 Y=0 A=8 X=0 Y=1 A=9 X=1 Y=1 A=2 No solution

18. X=0 Y=0 A=2 No solution

19. X=0 Y=0 A=0 No solution

20. X=0 Y=0 A=11 X=1 Y=0 A=0 No solution

21. X=0 Y=0 A=7 No solution

22. X=0 Y=0 A=0 No solution

23. X=0 Y=0 A=7 No solution

24. X=0 Y=0 A=5 X=3 Y=3 A=10 X=3 Y=2 No solution

25. X=0 Y=0 A=0 No solution

26. X=0 Y=0 A=1 No solution

27. X=0 Y=0 A=5 X=3 Y=3 A=9 X=4 Y=3 A=11 X=4 Y=3 A=10 X=4 Y=2 A=9 X=5 Y=2 A=8 X=5 Y=3 A=10 X=5 Y=2 A=8 X=5 Y=3 A=11 X=4 Y=3 A=8 X=4 Y=4 cost =10 Path=[[], (0,0), (3,3), (4,3), (4,3), (4,2), (5,2), (5,3), (5,2), (5,3), (4,3), (4,4)]

28. X=0 Y=0 A=8 X=0 Y=1 A=11 X=2 Y=1 A=8 X=2 Y=2 A=0 No solution

29. X=0 Y=0 A=4 No solution

30. X=0 Y=0 A=9 X=1 Y=0 A=10 No solution

31. X=0 Y=0 A=5 X=3 Y=3 A=10 X=3 Y=2 No solution

32. X=0 Y=0 A=10 No solution

33. X=0 Y=0 A=3 No solution

34. X=0 Y=0 A=4 No solution

35. X=0 Y=0 A=3 No solution

36. X=0 Y=0 A=1 No solution

37. X=0 Y=0 A=11 X=1 Y=0 A=9 X=2 Y=0 A=2 No solution

38. X=0 Y=0 A=10 No solution

39. X=0 Y=0 A=2 No solution

40. X=0 Y=0 A=2 No solution

41. X=0 Y=0 A=11 X=1 Y=0 A=10 No solution

42. X=0 Y=0 A=2 No solution

43. X=0 Y=0 A=7 No solution

44. X=0 Y=0 A=9 X=1 Y=0 A=11 X=1 Y=0 A=0 No solution

45. X=0 Y=0 A=3 No solution

46. X=0 Y=0 A=0 No solution

47. X=0 Y=0 A=7 No solution

48. X=0 Y=0 A=0 No solution

49. X=0 Y=0 A=7 No solution

50. X=0 Y=0 A=3 No solution

51. X=0 Y=0 A=0 No solution

52. X=0 Y=0 A=7 No solution

53. X=0 Y=0 A=3 No solution

54. X=0 Y=0 A=1 No solution

55. X=0 Y=0 A=11 X=1 Y=0 A=6 No solution

56. X=0 Y=0 A=10 No solution

57. X=0 Y=0 A=11 X=1 Y=0 A=4 No solution

58. X=0 Y=0 A=10 No solution

59. X=0 Y=0 A=2 No solution

60. X=0 Y=0 A=4 No solution

61. X=0 Y=0 A=5 X=3 Y=3 A=11 X=4 Y=3 A=10 X=4 Y=2 A=8 X=4 Y=3 A=8 X=4 Y=4 cost =5 Path=[[], (0,0), (3,3), (4,3), (4,2), (4,3), (4,4)]

62. X=0 Y=0 A=11 X=1 Y=0 A=11 X=1 Y=0 A=5 No solution

63. X=0 Y=0 A=8 X=0 Y=1 A=10 X=0 Y=0 A=3 No solution

64. X=0 Y=0 A=9 X=1 Y=0 A=4 No solution

65. X=0 Y=0 A=6 No solution

66. X=0 Y=0 A=8 X=0 Y=1 A=10 X=0 Y=0 A=7 No solution

67. X=0 Y=0 A=10 No solution

68. X=0 Y=0 A=1 No solution

69. X=0 Y=0 A=2 No solution

70. X=0 Y=0 A=5 X=3 Y=3 A=9 X=4 Y=3 A=9 X=5 Y=3 A=11 X=4 Y=3 A=11 X=4 Y=3 A=9 X=5 Y=3 A=8 X=5 Y=4 A=9 X=6 Y=4 A=11 X=5 Y=4 A=8 X=5 Y=5 A=9 X=6 Y=5 A=10 X=6 Y=4 A=10 X=6 Y=3 A=10 X=6 Y=2 A=8 X=6 Y=3 A=10 X=6 Y=2 A=11 X=3 Y=2 No solution

71. X=0 Y=0 A=2 No solution

72. X=0 Y=0 A=5 X=3 Y=3 A=11 X=4 Y=3 A=10 X=4 Y=2 A=10 X=4 Y=1 A=9 X=5 Y=1 A=11 X=2 Y=1 A=10 X=2 Y=0 A=11 X=1 Y=0 A=8 X=1 Y=1 A=9 X=2 Y=1 A=11 X=2 Y=1 A=11 X=2 Y=1 A=8 X=2 Y=2 A=8 X=2 Y=3 No solution

73. X=0 Y=0 A=4 No solution

74. X=0 Y=0 A=4 No solution

75. X=0 Y=0 A=7 No solution

76. X=0 Y=0 A=0 No solution

77. X=0 Y=0 A=6 No solution

78. X=0 Y=0 A=8 X=0 Y=1 A=11 X=2 Y=1 A=9 X=3 Y=1 No solution

79. X=0 Y=0 A=7 No solution

80. X=0 Y=0 A=4 No solution

81. X=0 Y=0 A=8 X=0 Y=1 A=7 No solution

82. X=0 Y=0 A=11 X=1 Y=0 A=1 No solution

83. X=0 Y=0 A=6 No solution

84. X=0 Y=0 A=1 No solution

85. X=0 Y=0 A=1 No solution

86. X=0 Y=0 A=10 No solution

87. X=0 Y=0 A=10 No solution

88. X=0 Y=0 A=10 No solution

89. X=0 Y=0 A=8 X=0 Y=1 A=11 X=2 Y=1 A=5 No solution

90. X=0 Y=0 A=11 X=1 Y=0 A=11 X=1 Y=0 A=1 No solution

91. X=0 Y=0 A=6 No solution

92. X=0 Y=0 A=7 No solution

93. X=0 Y=0 A=5 X=3 Y=3 A=9 X=4 Y=3 A=9 X=5 Y=3 A=9 X=6 Y=3 A=10 X=6 Y=2 A=11 X=3 Y=2 No solution

94. X=0 Y=0 A=2 No solution

95. X=0 Y=0 A=1 No solution

96. X=0 Y=0 A=6 No solution

97. X=0 Y=0 A=0 No solution

98. X=0 Y=0 A=5 X=3 Y=3 A=8 X=3 Y=4 A=8 X=3 Y=5 A=11 X=6 Y=5 A=8 X=6 Y=6 A=11 X=7 Y=6 A=9 X=8 Y=6 A=10 X=8 Y=5 A=9 X=9 Y=5 A=11 X=6 Y=5 A=10 X=6 Y=4 A=11 X=5 Y=4 A=8 X=5 Y=5 A=8 X=5 Y=6 A=9 X=6 Y=6 A=9 X=7 Y=6 A=8 X=7 Y=7 A=10 X=7 Y=6 A=8 X=7 Y=7 A=8 X=7 Y=8 A=11 X=9 Y=8 A=10 X=9 Y=7 A=8 X=9 Y=8 A=9 No solution

99. X=0 Y=0 A=1 No solution

100. X=0 Y=0 A=6 No solution

101. X=0 Y=0 A=2 No solution

102. X=0 Y=0 A=8 X=0 Y=1 A=3 No solution

103. X=0 Y=0 A=7 No solution

104. X=0 Y=0 A=6 No solution

105. X=0 Y=0 A=6 No solution

106. X=0 Y=0 A=5 X=3 Y=3 A=8 X=3 Y=4 A=11 X=5 Y=4 A=8 X=5 Y=5 A=9 X=6 Y=5 A=10 X=6 Y=4 A=9 X=7 Y=4 A=9 X=8 Y=4 A=9 X=9 Y=4 A=8 X=9 Y=5 A=11 X=6 Y=5 A=11 X=6 Y=5 A=10 X=6 Y=4 A=10 X=6 Y=3 A=8 X=6 Y=4 A=9 X=7 Y=4 A=10 X=7 Y=3 A=11 X=4 Y=3 A=9 X=5 Y=3 A=10 X=5 Y=2 A=10 X=5 Y=1 A=8 X=5 Y=2 A=10 X=5 Y=1 A=10 X=5 Y=0 A=9 X=6 Y=0 A=9 X=7 Y=0 A=11 X=1 Y=0 A=9 X=2 Y=0 A=8 X=2 Y=1 A=8 X=2 Y=2 A=10 X=2 Y=1 A=9 X=3 Y=1 No solution

107. X=0 Y=0 A=11 X=1 Y=0 A=5 No solution

108. X=0 Y=0 A=2 No solution

109. X=0 Y=0 A=10 No solution

110. X=0 Y=0 A=0 No solution

111. X=0 Y=0 A=0 No solution

112. X=0 Y=0 A=1 No solution

113. X=0 Y=0 A=0 No solution

114. X=0 Y=0 A=3 No solution

115. X=0 Y=0 A=11 X=1 Y=0 A=10 No solution

116. X=0 Y=0 A=9 X=1 Y=0 A=1 No solution

117. X=0 Y=0 A=11 X=1 Y=0 A=1 No solution

118. X=0 Y=0 A=10 No solution

119. X=0 Y=0 A=8 X=0 Y=1 A=9 X=1 Y=1 A=2 No solution

120. X=0 Y=0 A=2 No solution

121. X=0 Y=0 A=9 X=1 Y=0 A=0 No solution

122. X=0 Y=0 A=11 X=1 Y=0 A=4 No solution

123. X=0 Y=0 A=10 No solution

124. X=0 Y=0 A=0 No solution

125. X=0 Y=0 A=9 X=1 Y=0 A=7 No solution

126. X=0 Y=0 A=3 No solution

127. X=0 Y=0 A=1 No solution

128. X=0 Y=0 A=3 No solution

129. X=0 Y=0 A=2 No solution

130. X=0 Y=0 A=10 No solution

131. X=0 Y=0 A=0 No solution

132. X=0 Y=0 A=6 No solution

133. X=0 Y=0 A=11 X=1 Y=0 A=7 No solution

134. X=0 Y=0 A=10 No solution

135. X=0 Y=0 A=11 X=1 Y=0 A=0 No solution

136. X=0 Y=0 A=7 No solution

137. X=0 Y=0 A=8 X=0 Y=1 A=6 No solution

138. X=0 Y=0 A=8 X=0 Y=1 A=0 No solution

139. X=0 Y=0 A=11 X=1 Y=0 A=1 No solution

140. X=0 Y=0 A=9 X=1 Y=0 A=1 No solution

141. X=0 Y=0 A=7 No solution

142. X=0 Y=0 A=11 X=1 Y=0 A=1 No solution

143. X=0 Y=0 A=9 X=1 Y=0 A=11 X=1 Y=0 A=11 X=1 Y=0 A=0 No solution

144. X=0 Y=0 A=10 No solution

145. X=0 Y=0 A=4 No solution

146. X=0 Y=0 A=1 No solution

147. X=0 Y=0 A=8 X=0 Y=1 A=1 No solution

148. X=0 Y=0 A=7 No solution

149. X=0 Y=0 A=0 No solution

150. X=0 Y=0 A=9 X=1 Y=0 A=7 No solution

151. X=0 Y=0 A=5 X=3 Y=3 A=10 X=3 Y=2 No solution

152. X=0 Y=0 A=5 X=3 Y=3 A=9 X=4 Y=3 A=9 X=5 Y=3 A=9 X=6 Y=3 A=11 X=4 Y=3 A=9 X=5 Y=3 A=9 X=6 Y=3 A=11 X=4 Y=3 A=9 X=5 Y=3 A=10 X=5 Y=2 A=8 X=5 Y=3 A=11 X=4 Y=3 A=8 X=4 Y=4 cost =13 Path=[[], (0,0), (3,3), (4,3), (5,3), (6,3), (4,3), (5,3), (6,3), (4,3), (5,3), (5,2), (5,3), (4,3), (4,4)]

153. X=0 Y=0 A=10 No solution

154. X=0 Y=0 A=7 No solution

155. X=0 Y=0 A=5 X=3 Y=3 A=9 X=4 Y=3 A=11 X=4 Y=3 A=9 X=5 Y=3 A=9 X=6 Y=3 A=8 X=6 Y=4 A=9 X=7 Y=4 A=8 X=7 Y=5 A=9 X=8 Y=5 A=8 X=8 Y=6 A=10 X=8 Y=5 A=8 X=8 Y=6 A=9 X=9 Y=6 A=10 X=9 Y=5 A=9 No solution

156. X=0 Y=0 A=3 No solution

157. X=0 Y=0 A=5 X=3 Y=3 A=10 X=3 Y=2 No solution

158. X=0 Y=0 A=10 No solution

159. X=0 Y=0 A=3 No solution

160. X=0 Y=0 A=11 X=1 Y=0 A=5 No solution

161. X=0 Y=0 A=2 No solution

162. X=0 Y=0 A=0 No solution

163. X=0 Y=0 A=10 No solution

164. X=0 Y=0 A=8 X=0 Y=1 A=4 No solution

165. X=0 Y=0 A=6 No solution

166. X=0 Y=0 A=10 No solution

167. X=0 Y=0 A=7 No solution

168. X=0 Y=0 A=11 X=1 Y=0 A=10 No solution

169. X=0 Y=0 A=5 X=3 Y=3 A=8 X=3 Y=4 A=8 X=3 Y=5 A=10 X=3 Y=4 A=8 X=3 Y=5 A=10 X=3 Y=4 A=11 X=5 Y=4 A=9 X=6 Y=4 A=11 X=5 Y=4 A=11 X=5 Y=4 A=9 X=6 Y=4 A=8 X=6 Y=5 A=8 X=6 Y=6 A=9 X=7 Y=6 A=11 X=7 Y=6 A=9 X=8 Y=6 A=8 X=8 Y=7 A=8 X=8 Y=8 A=9 X=9 Y=8 A=8 X=9 Y=9 A=9 No solution

170. X=0 Y=0 A=3 No solution

171. X=0 Y=0 A=11 X=1 Y=0 A=8 X=1 Y=1 A=1 No solution

172. X=0 Y=0 A=4 No solution

173. X=0 Y=0 A=0 No solution

174. X=0 Y=0 A=11 X=1 Y=0 A=1 No solution

175. X=0 Y=0 A=1 No solution

176. X=0 Y=0 A=11 X=1 Y=0 A=4 No solution

177. X=0 Y=0 A=4 No solution

178. X=0 Y=0 A=5 X=3 Y=3 A=10 X=3 Y=2 No solution

179. X=0 Y=0 A=2 No solution

180. X=0 Y=0 A=11 X=1 Y=0 A=7 No solution

181. X=0 Y=0 A=0 No solution

182. X=0 Y=0 A=4 No solution

183. X=0 Y=0 A=7 No solution

184. X=0 Y=0 A=1 No solution

185. X=0 Y=0 A=6 No solution

186. X=0 Y=0 A=8 X=0 Y=1 A=10 X=0 Y=0 A=7 No solution

187. X=0 Y=0 A=8 X=0 Y=1 A=3 No solution

188. X=0 Y=0 A=6 No solution

189. X=0 Y=0 A=1 No solution

190. X=0 Y=0 A=5 X=3 Y=3 A=11 X=4 Y=3 A=10 X=4 Y=2 A=9 X=5 Y=2 A=10 X=5 Y=1 A=10 X=5 Y=0 A=10 No solution

191. X=0 Y=0 A=2 No solution

192. X=0 Y=0 A=6 No solution

193. X=0 Y=0 A=8 X=0 Y=1 A=0 No solution

194. X=0 Y=0 A=0 No solution

195. X=0 Y=0 A=9 X=1 Y=0 A=5 No solution

196. X=0 Y=0 A=5 X=3 Y=3 A=8 X=3 Y=4 A=11 X=5 Y=4 A=11 X=5 Y=4 A=10 X=5 Y=3 A=10 X=5 Y=2 A=11 X=3 Y=2 No solution

197. X=0 Y=0 A=8 X=0 Y=1 A=4 No solution

198. X=0 Y=0 A=9 X=1 Y=0 A=0 No solution

199. X=0 Y=0 A=3 No solution

200. X=0 Y=0 A=7 No solution

201. X=0 Y=0 A=8 X=0 Y=1 A=2 No solution

202. X=0 Y=0 A=9 X=1 Y=0 A=2 No solution

203. X=0 Y=0 A=3 No solution

204. X=0 Y=0 A=1 No solution

205. X=0 Y=0 A=2 No solution

206. X=0 Y=0 A=5 X=3 Y=3 A=9 X=4 Y=3 A=11 X=4 Y=3 A=8 X=4 Y=4 cost =4 Path=[[], (0,0), (3,3), (4,3), (4,3), (4,4)] X=0 Y=0 A=0 No solution

207. X=0 Y=0 A=9 X=1 Y=0 A=10 No solution

208. X=0 Y=0 A=1 No solution

209. X=0 Y=0 A=5 X=3 Y=3 A=11 X=4 Y=3 A=8 X=4 Y=4 cost =3 Path=[[], (0,0), (3,3), (4,3), (4,4)]

210. X=0 Y=0 A=10 No solution

211. X=0 Y=0 A=9 X=1 Y=0 A=1 No solution

212. X=0 Y=0 A=1 No solution

213. X=0 Y=0 A=3 No solution

214. X=0 Y=0 A=8 X=0 Y=1 A=9 X=1 Y=1 A=6 No solution

215. X=0 Y=0 A=3 No solution

216. X=0 Y=0 A=4 No solution

217. X=0 Y=0 A=6 No solution

218. X=0 Y=0 A=2 No solution

219. X=0 Y=0 A=8 X=0 Y=1 A=5 No solution

220. X=0 Y=0 A=7 No solution

221. X=0 Y=0 A=5 X=3 Y=3 A=10 X=3 Y=2 No solution

222. X=0 Y=0 A=6 No solution

223. X=0 Y=0 A=0 No solution

224. X=0 Y=0 A=10 No solution

225. X=0 Y=0 A=2 No solution

226. X=0 Y=0 A=2 No solution

227. X=0 Y=0 A=0 No solution

228. X=0 Y=0 A=6 No solution

229. X=0 Y=0 A=10 No solution

230. X=0 Y=0 A=0 No solution

231. X=0 Y=0 A=8 X=0 Y=1 A=9 X=1 Y=1 A=11 X=2 Y=1 A=3 No solution

232. X=0 Y=0 A=4 No solution

233. X=0 Y=0 A=11 X=1 Y=0 A=11 X=1 Y=0 A=3 No solution

234. X=0 Y=0 A=7 No solution

235. X=0 Y=0 A=7 No solution

236. X=0 Y=0 A=1 No solution

237. X=0 Y=0 A=0 No solution

238. X=0 Y=0 A=8 X=0 Y=1 A=7 No solution

239. X=0 Y=0 A=7 No solution

240. X=0 Y=0 A=3 No solution

241. X=0 Y=0 A=5 X=3 Y=3 A=11 X=4 Y=3 A=9 X=5 Y=3 A=11 X=4 Y=3 A=8 X=4 Y=4 cost =5 Path=[[], (0,0), (3,3), (4,3), (5,3), (4,3), (4,4)]

242. X=0 Y=0 A=9 X=1 Y=0 A=0 No solution

243. X=0 Y=0 A=5 X=3 Y=3 A=9 X=4 Y=3 A=8 X=4 Y=4 cost =3 Path=[[], (0,0), (3,3), (4,3), (4,4)]

244. X=0 Y=0 A=1 No solution

245. X=0 Y=0 A=10 No solution

246. X=0 Y=0 A=4 No solution

247. X=0 Y=0 A=7 No solution

248. X=0 Y=0 A=11 X=1 Y=0 A=7 No solution

249. X=0 Y=0 A=1 No solution

250. X=0 Y=0 A=1 No solution

251. X=0 Y=0 A=4 No solution

252. X=0 Y=0 A=7 No solution

253. X=0 Y=0 A=3 No solution

254. X=0 Y=0 A=4 No solution

255. X=0 Y=0 A=10 No solution

256. X=0 Y=0 A=11 X=1 Y=0 A=0 No solution

257. X=0 Y=0 A=5 X=3 Y=3 A=10 X=3 Y=2 No solution

258. X=0 Y=0 A=2 No solution

259. X=0 Y=0 A=5 X=3 Y=3 A=11 X=4 Y=3 A=11 X=4 Y=3 A=11 X=4 Y=3 A=10 X=4 Y=2 A=11 X=3 Y=2 No solution

260. X=0 Y=0 A=1 No solution

261. X=0 Y=0 A=0 No solution

262. X=0 Y=0 A=4 No solution

263. X=0 Y=0 A=6 No solution

264. X=0 Y=0 A=0 No solution

265. X=0 Y=0 A=10 No solution

266. X=0 Y=0 A=1 No solution

267. X=0 Y=0 A=7 No solution

268. X=0 Y=0 A=9 X=1 Y=0 A=1 No solution

269. X=0 Y=0 A=0 No solution

270. X=0 Y=0 A=6 No solution

271. X=0 Y=0 A=11 X=1 Y=0 A=9 X=2 Y=0 A=5 No solution

272. X=0 Y=0 A=4 No solution

273. X=0 Y=0 A=2 No solution

274. X=0 Y=0 A=3 No solution

275. X=0 Y=0 A=9 X=1 Y=0 A=3 No solution

276. X=0 Y=0 A=1 No solution

277. X=0 Y=0 A=5 X=3 Y=3 A=10 X=3 Y=2 No solution

278. X=0 Y=0 A=0 No solution

279. X=0 Y=0 A=3 No solution

280. X=0 Y=0 A=11 X=1 Y=0 A=2 No solution

281. X=0 Y=0 A=1 No solution

282. X=0 Y=0 A=6 No solution

283. X=0 Y=0 A=1 No solution

284. X=0 Y=0 A=3 No solution

285. X=0 Y=0 A=0 No solution

286. X=0 Y=0 A=3 No solution

287. X=0 Y=0 A=1 No solution

288. X=0 Y=0 A=10 No solution

289. X=0 Y=0 A=4 No solution

290. X=0 Y=0 A=5 X=3 Y=3 A=11 X=4 Y=3 A=8 X=4 Y=4 cost =3 Path=[[], (0,0), (3,3), (4,3), (4,4)]

291. X=0 Y=0 A=3 No solution

292. X=0 Y=0 A=0 No solution

293. X=0 Y=0 A=9 X=1 Y=0 A=6 No solution

294. X=0 Y=0 A=7 No solution

295. X=0 Y=0 A=3 No solution

296. X=0 Y=0 A=4 No solution

297. X=0 Y=0 A=1 No solution

298. X=0 Y=0 A=6 No solution

299. X=0 Y=0 A=0 No solution

300. X=0 Y=0 A=9 X=1 Y=0 A=6 No solution

301. X=0 Y=0 A=3 No solution

302. X=0 Y=0 A=4 No solution

303. X=0 Y=0 A=1 No solution

304. X=0 Y=0 A=6 No solution

305. X=0 Y=0 A=1 No solution

306. X=0 Y=0 A=1 No solution

307. X=0 Y=0 A=10 No solution

308. X=0 Y=0 A=2 No solution

309. X=0 Y=0 A=5 X=3 Y=3 A=9 X=4 Y=3 A=11 X=4 Y=3 A=8 X=4 Y=4 cost =4 Path=[[], (0,0), (3,3), (4,3), (4,3), (4,4)]

310. X=0 Y=0 A=0 No solution

311. X=0 Y=0 A=4 No solution

312. X=0 Y=0 A=11 X=1 Y=0 A=0 No solution

313. X=0 Y=0 A=2 No solution

314. X=0 Y=0 A=3 No solution

315. X=0 Y=0 A=6 No solution

316. X=0 Y=0 A=6 No solution

317. X=0 Y=0 A=4 No solution

318. X=0 Y=0 A=3 No solution

319. X=0 Y=0 A=3 No solution

320. X=0 Y=0 A=7 No solution

321. X=0 Y=0 A=9 X=1 Y=0 A=1 No solution

322. X=0 Y=0 A=0 No solution

323. X=0 Y=0 A=0 No solution

324. X=0 Y=0 A=11 X=1 Y=0 A=0 No solution

325. X=0 Y=0 A=7 No solution

326. X=0 Y=0 A=2 No solution

327. X=0 Y=0 A=10 No solution

328. X=0 Y=0 A=10 No solution

329. X=0 Y=0 A=10 No solution

330. X=0 Y=0 A=10 No solution

331. X=0 Y=0 A=1 No solution

332. X=0 Y=0 A=0 No solution

333. X=0 Y=0 A=6 No solution

334. X=0 Y=0 A=7 No solution

335. X=0 Y=0 A=3 No solution

336. X=0 Y=0 A=9 X=1 Y=0 A=4 No solution

337. X=0 Y=0 A=11 X=1 Y=0 A=5 No solution

338. X=0 Y=0 A=0 No solution

339. X=0 Y=0 A=8 X=0 Y=1 A=8 X=0 Y=2 A=10 X=0 Y=1 A=8 X=0 Y=2 A=4 No solution

340. X=0 Y=0 A=5 X=3 Y=3 A=11 X=4 Y=3 A=11 X=4 Y=3 A=10 X=4 Y=2 A=10 X=4 Y=1 A=9 X=5 Y=1 A=10 X=5 Y=0 A=9 X=6 Y=0 A=10 No solution

341. X=0 Y=0 A=8 X=0 Y=1 A=10 X=0 Y=0 A=7 No solution

342. X=0 Y=0 A=7 No solution

343. X=0 Y=0 A=11 X=1 Y=0 A=0 No solution

344. X=0 Y=0 A=8 X=0 Y=1 A=11 X=2 Y=1 A=11 X=2 Y=1 A=3 No solution

345. X=0 Y=0 A=3 No solution

346. X=0 Y=0 A=7 No solution

347. X=0 Y=0 A=3 No solution

348. X=0 Y=0 A=9 X=1 Y=0 A=2 No solution

349. X=0 Y=0 A=6 No solution

350. X=0 Y=0 A=6 No solution

351. X=0 Y=0 A=4 No solution

352. X=0 Y=0 A=11 X=1 Y=0 A=0 No solution

353. X=0 Y=0 A=2 No solution

354. X=0 Y=0 A=11 X=1 Y=0 A=9 X=2 Y=0 A=10 No solution

355. X=0 Y=0 A=7 No solution

356. X=0 Y=0 A=3 No solution

357. X=0 Y=0 A=2 No solution

358. X=0 Y=0 A=3 No solution

359. X=0 Y=0 A=10 No solution

360. X=0 Y=0 A=4 No solution

361. X=0 Y=0 A=5 X=3 Y=3 A=8 X=3 Y=4 A=11 X=5 Y=4 A=11 X=5 Y=4 A=10 X=5 Y=3 A=8 X=5 Y=4 A=9 X=6 Y=4 A=8 X=6 Y=5 A=9 X=7 Y=5 A=11 X=6 Y=5 A=9 X=7 Y=5 A=8 X=7 Y=6 A=9 X=8 Y=6 A=10 X=8 Y=5 A=8 X=8 Y=6 A=11 X=7 Y=6 A=10 X=7 Y=5 A=10 X=7 Y=4 A=10 X=7 Y=3 A=11 X=4 Y=3 A=8 X=4 Y=4 cost =30 Path=[[], (0,0), (3,3), (3,4), (5,4), (5,4), (5,3), (5,4), (6,4), (6,5), (7,5), (6,5), (7,5), (7,6), (8,6), (8,5), (8,6), (7,6), (7,5), (7,4), (7,3), (4,3), (4,4)]

362. X=0 Y=0 A=6 No solution

363. X=0 Y=0 A=8 X=0 Y=1 A=3 No solution

364. X=0 Y=0 A=3 No solution

365. X=0 Y=0 A=10 No solution

366. X=0 Y=0 A=1 No solution

367. X=0 Y=0 A=8 X=0 Y=1 A=4 No solution

368. X=0 Y=0 A=8 X=0 Y=1 A=1 No solution

369. X=0 Y=0 A=7 No solution

370. X=0 Y=0 A=5 X=3 Y=3 A=8 X=3 Y=4 A=11 X=5 Y=4 A=9 X=6 Y=4 A=8 X=6 Y=5 A=9 X=7 Y=5 A=10 X=7 Y=4 A=9 X=8 Y=4 A=11 X=5 Y=4 A=9 X=6 Y=4 A=8 X=6 Y=5 A=8 X=6 Y=6 A=11 X=7 Y=6 A=8 X=7 Y=7 A=10 X=7 Y=6 A=10 X=7 Y=5 A=10 X=7 Y=4 A=8 X=7 Y=5 A=11 X=6 Y=5 A=8 X=6 Y=6 A=10 X=6 Y=5 A=10 X=6 Y=4 A=11 X=5 Y=4 A=8 X=5 Y=5 A=11 X=6 Y=5 A=10 X=6 Y=4 A=8 X=6 Y=5 A=9 X=7 Y=5 A=9 X=8 Y=5 A=8 X=8 Y=6 A=8 X=8 Y=7 A=9 X=9 Y=7 A=11 X=8 Y=7 A=11 X=8 Y=7 A=8 X=8 Y=8 A=11 X=9 Y=8 A=10 X=9 Y=7 A=10 X=9 Y=6 A=9 No solution

371. X=0 Y=0 A=4 No solution

372. X=0 Y=0 A=8 X=0 Y=1 A=11 X=2 Y=1 A=1 No solution

373. X=0 Y=0 A=11 X=1 Y=0 A=6 No solution

374. X=0 Y=0 A=11 X=1 Y=0 A=7 No solution

375. X=0 Y=0 A=8 X=0 Y=1 A=0 No solution

376. X=0 Y=0 A=5 X=3 Y=3 A=9 X=4 Y=3 A=10 X=4 Y=2 A=9 X=5 Y=2 A=8 X=5 Y=3 A=9 X=6 Y=3 A=8 X=6 Y=4 A=10 X=6 Y=3 A=10 X=6 Y=2 A=11 X=3 Y=2 No solution

377. X=0 Y=0 A=3 No solution

378. X=0 Y=0 A=5 X=3 Y=3 A=11 X=4 Y=3 A=8 X=4 Y=4 cost =3 Path=[[], (0,0), (3,3), (4,3), (4,4)]

379. X=0 Y=0 A=6 No solution

380. X=0 Y=0 A=4 No solution

381. X=0 Y=0 A=2 No solution

382. X=0 Y=0 A=0 No solution

383. X=0 Y=0 A=5 X=3 Y=3 A=11 X=4 Y=3 A=11 X=4 Y=3 A=10 X=4 Y=2 A=11 X=3 Y=2 No solution

384. X=0 Y=0 A=10 No solution

385. X=0 Y=0 A=8 X=0 Y=1 A=11 X=2 Y=1 A=3 No solution

386. X=0 Y=0 A=7 No solution

387. X=0 Y=0 A=5 X=3 Y=3 A=11 X=4 Y=3 A=10 X=4 Y=2 A=8 X=4 Y=3 A=10 X=4 Y=2 A=10 X=4 Y=1 A=8 X=4 Y=2 A=8 X=4 Y=3 A=9 X=5 Y=3 A=8 X=5 Y=4 A=11 X=5 Y=4 A=8 X=5 Y=5 A=11 X=6 Y=5 A=11 X=6 Y=5 A=11 X=6 Y=5 A=8 X=6 Y=6 A=9 X=7 Y=6 A=8 X=7 Y=7 A=11 X=8 Y=7 A=8 X=8 Y=8 A=11 X=9 Y=8 A=11 X=9 Y=8 A=9 No solution

388. X=0 Y=0 A=2 No solution

389. X=0 Y=0 A=2No solution

390. X=0 Y=0 A=9 X=1 Y=0 A=10 No solution

391. X=0 Y=0 A=1 No solution

392. X=0 Y=0 A=9 X=1 Y=0 A=5 No solution

393. X=0 Y=0 A=1 No solution

394. X=0 Y=0 A=11 X=1 Y=0 A=5 No solution

395. X=0 Y=0 A=1 No solution

396. X=0 Y=0 A=11 X=1 Y=0 A=6 No solution

397. X=0 Y=0 A=2 No solution

398. X=0 Y=0 A=9 X=1 Y=0 A=7 No solution

399. X=0 Y=0 A=3 No solution

400. X=0 Y=0 A=5 X=3 Y=3 A=8 X=3 Y=4 A=9 X=4 Y=4 cost =3 Path=[[], (0,0), (3,3), (3,4), (4,4)]

Conclusion: Third algorithm improves backtracking from optimal path search perspective if the point is found during up and right stage. For the worst case the time require3d for the 1st iteration is neglactable comparing with backtracking stage especially for the hard-to-solve-maps (see Part Three). Random search is useless in this variation.

Part Two No significant changes will appear to random search because it does not use info about adjacent cells. For the 3rd algorithm and backtracking it is a nice chance to make a right move, the increment of the seen area can reduce time number of steps for crucially. See examples and explanations under them.

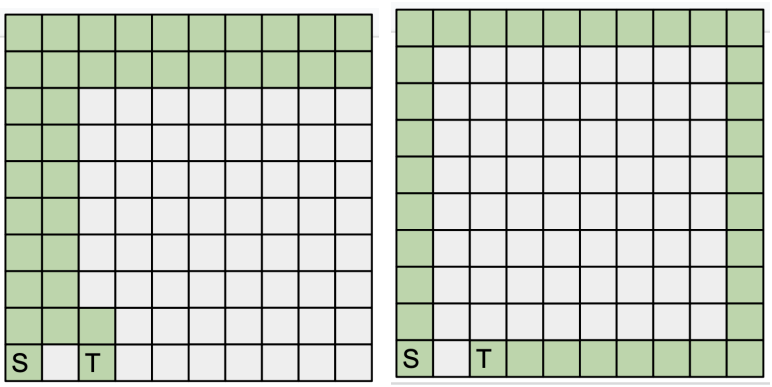


Figure 12: Green cells - the ones visited before the touchdown in case of radius of vision =1. In case of r=2 it will take 2 steps in both algorithms. In case of r=1 findind touchdown will take 37 cells for the 3rd algorithm, right picture, and 35 cells for backtracking,left picture.

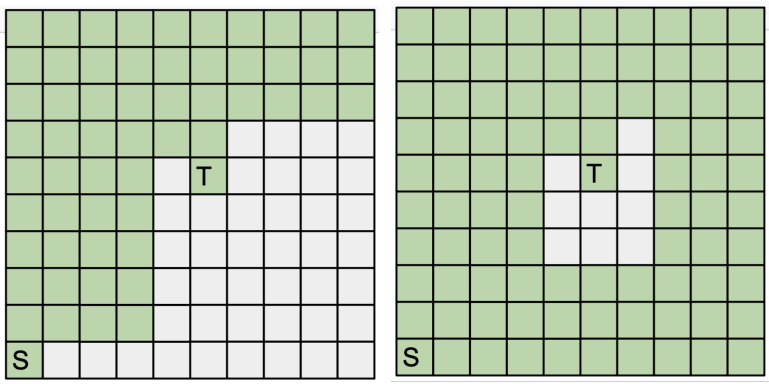


Figure 13: Green cells - the ones visited before the touchdown in case of radius of vision =1. In case of r=1 findind touchdown will take 58 cells for the 3rd algorithm, right picture, and 91 cells for backtracking, left picture.

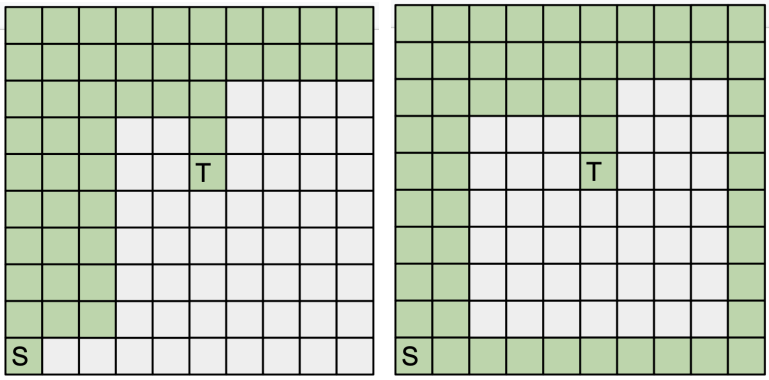


Figure 14: Green cells - the ones visited before the touchdown in case of radius of vision =2. In case of r=2 findind touchdown will take 47 cells for the 3rd algorithm, right picture, and 57 cells for backtracking, left picture. For 3rd algorithm improvement is 11 cells, for backtracking it is 34 cells.

```
Backtracking  82[[],(0,0),(0,1),(0,2),(0,3),(0,4),(0,5),(0,6),(0,7),(0,8),(0,9),(1,9),(2,9),(3,9),(4,9),(5,9),(6,9),(7,9),
(8,9),(9,9),(9,8),(9,7),(9,6),(9,5),(9,4),(9,3),(9,2),(9,1),(9,0),(8,0),(8,1),(8,2),(8,3),(8,4),(8,5),(8,6),(8,7),(8,8),(7
,8),(7,7),(7,6),(7,5),(7,4),(7,3),(7,2),(7,1),(7,0),(6,0),(6,1),(6,2),(6,3),(6,4),(6,5),(6,6),(6,7),(6,8),(5,8),(5,7),(5,6
),(5,5),(5,4),(5,3),(5,2),(5,1),(5,0),(4,0),(4,1),(4,2),(4,3),(4,4),(4,5),(4,6),(4,7),(4,8),(3,8),(3,7),(3,6),(3,5),(3,4),
(3,3),(3,2),(3,1),(3,0),(2,0)|_50510]
% 66,855 inferences, 0.008 CPU in 0.008 seconds (99% CPU, 8299814 Lips)
Random search 1 No solution
% 194 inferences, 0.001 CPU in 0.001 seconds (84% CPU, 335060 Lips)
Improved backtracking  4[[],(0,0),(0,1),(1,1),(2,1),(2,0)|_51742]
% 44,280,416 inferences, 3.099 CPU in 3.102 seconds (100% CPU, 14290634 Lips)
```

```
Backtracking  2[[],(0,0),(1,0),(2,0)|_7310]
% 148 inferences, 0.000 CPU in 0.000 seconds (96% CPU, 1494949 Lips)
Random search 1 No solution
% 243 inferences, 0.001 CPU in 0.001 seconds (88% CPU, 404326 Lips)
Improved backtracking  2[[],(0,0),(1,0),(2,0)|_8494]
% 80 inferences, 0.000 CPU in 0.000 seconds (92% CPU, 2352941 Lips)
true.
```

```
Backtracking  52[[],(0,0),(0,1),(0,2),(0,3),(0,4),(0,5),(0,6),(0,7),(0,8),(0,9),(1,9),(2,9),(3,9),(4,9),(5,9),(6,9),(7,9),
(8,9),(9,9),(9,8),(9,7),(9,6),(9,5),(9,4),(9,3),(9,2),(9,1),(9,0),(8,0),(8,1),(8,2),(8,3),(8,4),(8,5),(8,6),(8,7),(8,8),(7
,8),(7,7),(7,6),(7,5),(7,4),(7,3),(7,2),(7,1),(7,0),(6,0),(6,1),(6,2),(6,3),(6,4),(6,5),(5,5)|_25310]
% 38,087 inferences, 0.004 CPU in 0.004 seconds (100% CPU, 8733547 Lips)
Random search 1 No solution
% 52 inferences, 0.001 CPU in 0.001 seconds (87% CPU, 90121 Lips)
Improved backtracking  12[[],(0,0),(0,1),(0,2),(0,3),(0,4),(0,5),(0,6),(1,6),(2,6),(3,6),(4,6),(5,6),(5,5)|_27094]
% 589,440 inferences, 0.050 CPU in 0.050 seconds (100% CPU, 11696631 Lips)
```

```
Backtracking  42[[],(0,0),(0,1),(0,2),(0,3),(0,4),(0,5),(0,6),(0,7),(0,8),(0,9),(1,9),(2,9),(3,9),(4,9),(5,9),(6,9),(7,9),
(8,9),(9,9),(9,8),(9,7),(9,6),(9,5),(9,4),(9,3),(9,2),(9,1),(9,0),(8,0),(8,1),(8,2),(8,3),(8,4),(8,5),(8,6),(8,7),(8,8),(7
,8),(7,7),(7,6),(7,5),(6,5),(5,5)|_19310]
% 30,362 inferences, 0.004 CPU in 0.004 seconds (97% CPU, 8277535 Lips)
Random search 1 No solution
% 52 inferences, 0.001 CPU in 0.001 seconds (87% CPU, 89655 Lips)
Improved backtracking  14[[],(0,0),(0,1),(0,2),(0,3),(0,4),(0,5),(0,6),(0,7),(1,7),(2,7),(3,7),(4,7),(5,7),(5,6),(5,5)|_21
322]
% 165,298 inferences, 0.014 CPU in 0.014 seconds (100% CPU, 11766657 Lips)
```

Figure 15: 1st picture- test for figure 12, with r=1, test 8. 2nd picture- test for figure 12, with r=2, test 8. 3rd picture- test for figure 13, with r=1, test 9. 4th picture- test for figure 13, with r=2, test 9. For backtracking we have both time and path improvement, for 3rd algorithm we have only significant time improvement.

Part Three

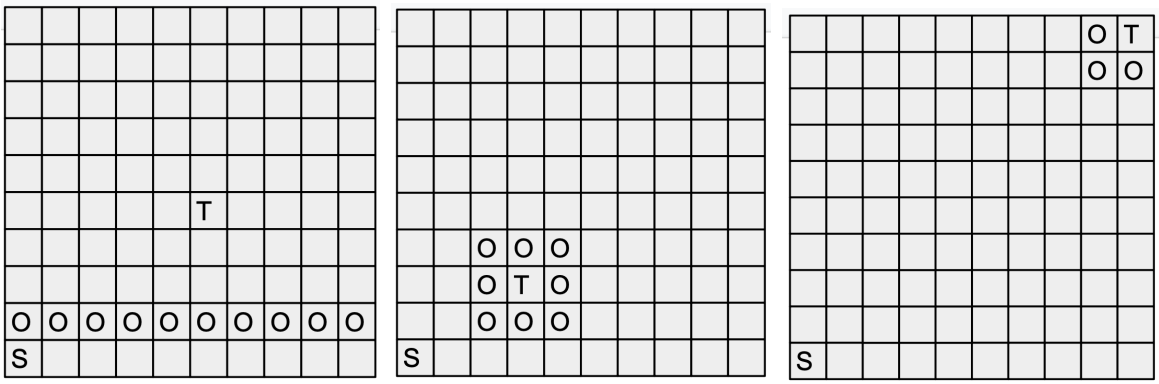


Figure 16: Obvious impossible to solve maps - when the touchdown is not reachable.Isolated player,surrounded touchdown, and partially surrounded on an edge touchdown

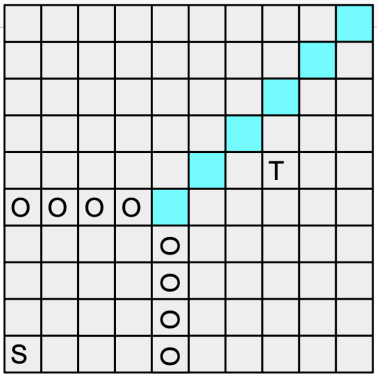


Figure 17: The map is impossible to solve if and only if there is no human staying on the blue line.Otherwise, up-right diagonal throw solves the map.

Impossible to solve maps maps: If a map is a variation (e.g. there is a line of empty cells between touchdown and isolation orks circle) of those or all touchdowns in a map are a combination of those ⇒ the map is insolvable. The other type of impossible to solve maps are when there is no touchdown at all. The common thing between all of them is that the player never gets to the part of the map surrounded by orks, where the touchdown is.
Note: by impossible to solve maps I mean maps, where there is no path for touchdown. If we assume that "no path exists" is an also a solution (if it is correct answer from the logical side as well) than backtracking and third algorithm always return a solution, therefore, all maps are solvable

for them.

Hard to solve maps:

For the 3rd algorithm hard-to-solve maps are those, for which we have to go to the second stage with backtracking. Example of such a map was provided in Part 1, figure 2.

For the backtracking, hard to solve maps are ones without touchdowns because we have to check all the cells. The most difficult ones are absolutely empty or only with people. They are the most difficult because each cell generates 10 options of acting (6 throws and 4 steps). Moreover,we get to the same cell during several distinct backtracking branches. For point with coordinates (x,y), counting from (0,0), we get $C^{x+1}_{(x+1)+(y+1)}$ times - this is a well-known fact from combinatorics, look here for explanations. Hence, each cell with an ork reduces the number of choices for $10C^{x}_{(x+1)+(y+1)}$ cases, which is on average -for a cell (4,4) - is 1260 cases.

The other case of difficult to solve maps also strongly connected with the number of paths that should be checked before. We should understand that in prolog the order of function call really matters. During backtracking we call 10 operations, but lets focus on the 4 of them, which are steps. The order you call them is the order in which this branches will be executed. As far as I choose the 1st satisfactory path, the worst case is when it is met during the last branch execution. In my implementation the order is : up, right, down, left. Therefore, empty maps with a touchdown located at the most down sides of the map with half-surrounded touchdown are difficult to solve as during backtracking other branches because of the orks surrounding the backtracking will not be able to see the touchdown and it will find it only during checking the last branches.It it also important that from the right edge of the map till exit surrounding there should be orks, otherwise it will be solved much faster. Example - figure 18. However, this case is much easier to solve than empty map as coming to the appropriate cells ones will finish the execution immediately And many other cells will no be visited at all.

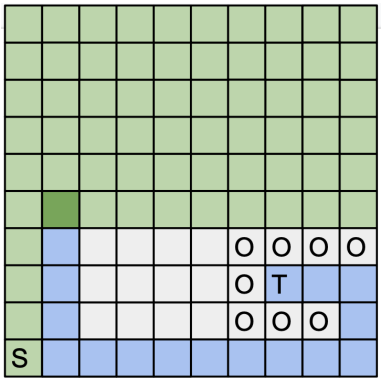


Figure 18: Before the touchdown will be found - backtracking will visit all light green cells, than coming up in the recursion tree from the dark green cell we will step on the correct blue path to the exit.