Using any programming language implement a state space search algorithm to solve a problem or a puzzle according to your variant.

[Problems and puzzles](https://docs.google.com/document/d/1O13QDkPO1ooG1kFBE7SBQUftQB_-4IcMYpjkuUTA1h8/edit?usp=sharing)

Variants:

1. Solve the 15 puzzle using recursive best first search.

2. Solve the cannibals and missionaries problem using breadth-first search.

3. Solve the tower of Hanoi puzzle using greedy best-first search.

4. Solve the n queens puzzle using iterative deepening strategy.

5. Solve the water pouring puzzle using bidirectional best-first search.

6. Solve a knight's tour puzzle using A\* search.

7. Solve the traveling salesman problem using uniform-cost search.

8. Solve the water pouring puzzle using greedy best-first search.

9. Solve the cannibals and missionaries problem using uniform-cost search.

10. Solve a knight's tour puzzle using iterative deepening strategy.

11. Solve the 15 puzzle using using bidirectional best-first search.

12. Solve the traveling salesman problem using using breadth-first search.

13. Solve the tower of Hanoi puzzle using recursive best first search.

14. Solve the n queens puzzle using A\* search.

15. Solve the 15 puzzle using uniform-cost search.

16. Solve the cannibals and missionaries problem using iterative deepening strategy.

17. Solve the tower of Hanoi puzzle using A\* search.

18. Solve the n queens puzzle using recursive best first search.

19. Solve the water pouring puzzle using breadth-first search.

20. Solve a knight's tour puzzle using greedy best-first search.

21. Solve the traveling salesman problem using bidirectional best-first search.

22. Solve the cannibals and missionaries problem using greedy best-first search.

23. Solve the tower of Hanoi puzzle using uniform-cost search.

24. Solve the n queens puzzle using breadth-first search.

25. Solve the water pouring puzzle using iterative deepening strategy.

26. Solve a knight's tour puzzle using recursive best first search.