CS162 – Programming Techniques

Lab 08
Advanced Recursion

Cảm ơn thầy Trần Duy Quang đã cung cấp template cho môn học

epartment of Software Engineering-FIT-VNU-HCMUS

1 Notes

Create a single solution/folder to store your source code in a week.

Then, create a project/sub-folder to store your source code of each assignment.

The source code in an assignment should have at least 3 files:

- A header file (.h): struct definition, function prototypes/definition.
- A source file (.cpp): function implementation.
- Another source file (.cpp): named YourID_Ex01.cpp, main function. Replace 01 by id of an assignment.

Make sure your source code was built correctly. Use many test cases to check your code before submitting to Moodle.

Name of your submission, for example: **18125001_W01_07.zip**

2 Content

In this lab, we will review the following topics:

- Solve popular recursion problems.

3 Assignments

A: 01 problems / assignments.

H: 07 problems / assignments.

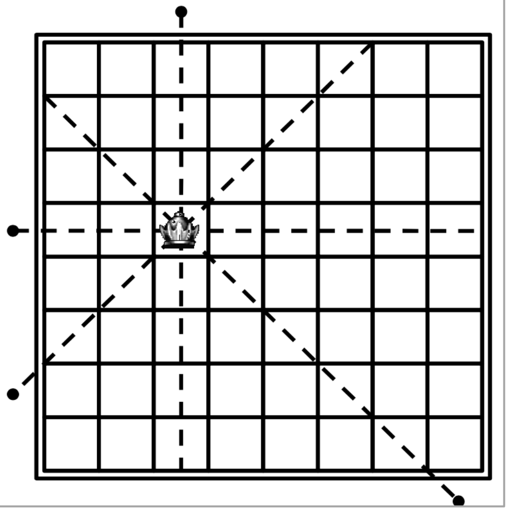
Implement these problems in the **recursive style**.

3.1 N Queens

Input: N: a chessboard with size N * N.

Output: location (i, j) of N queens in that chessboard. Only 1 solution is required.

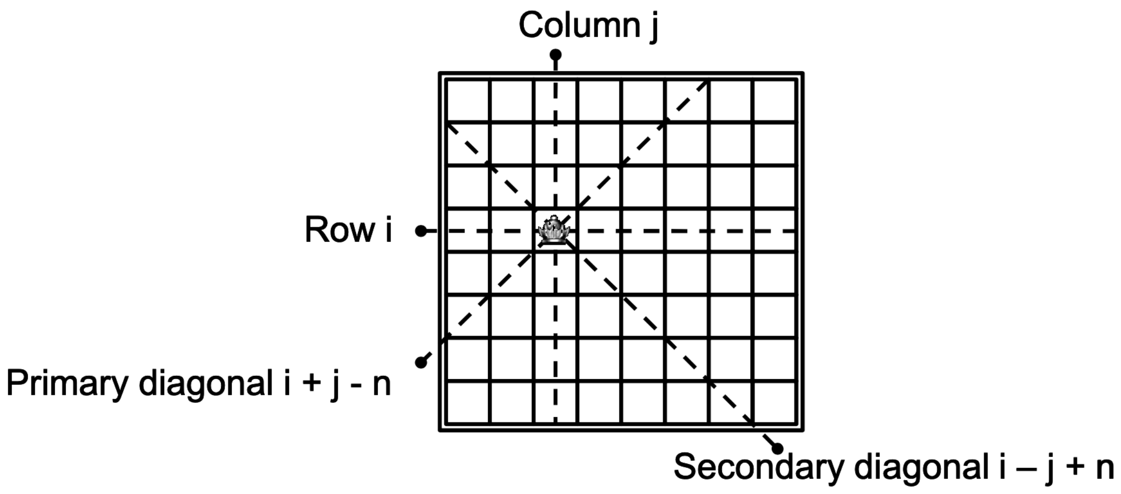
- Eight Queens:
  - Problem:
    - Chessboard 8 x 8 cells.
    - Try to put 8 queens on board.
    - The queens do not capture each other:
      - Not in same row.
      - Not in same column.
      - Not in same primary diagonal.
      - Not in same secondary diagonal.



Eight Queens:

Analysis:

- Can only put a queen on un-captured cells.
- If queen is put at (i, j), which cells are captured?



Eight Queens:

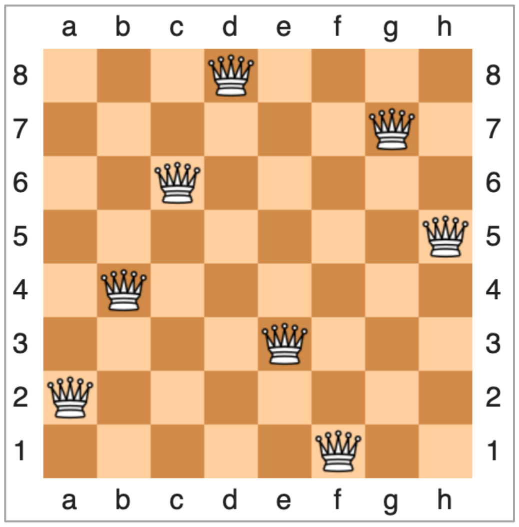
Backtracking:

```
TryQueen ( cell (i, j), rowFlag, colFlag, pDiaFlag, sDiaFlag  )
{
    if ( cell (i, j) is captured )
        return;

    Update captures at the cell;

    if ( i is last row )
        Print result;
    else
        for (int k = 0; k < 7; k++)
            TryQueen(cell (i+1, k),rowFlag,colFlag,pDiaFlag, sDiaFlag);

    Roll back captures at the cell;
}
```


a b c d e f g h
8 7 6 5 4 3 2 1

3.2 N Queens

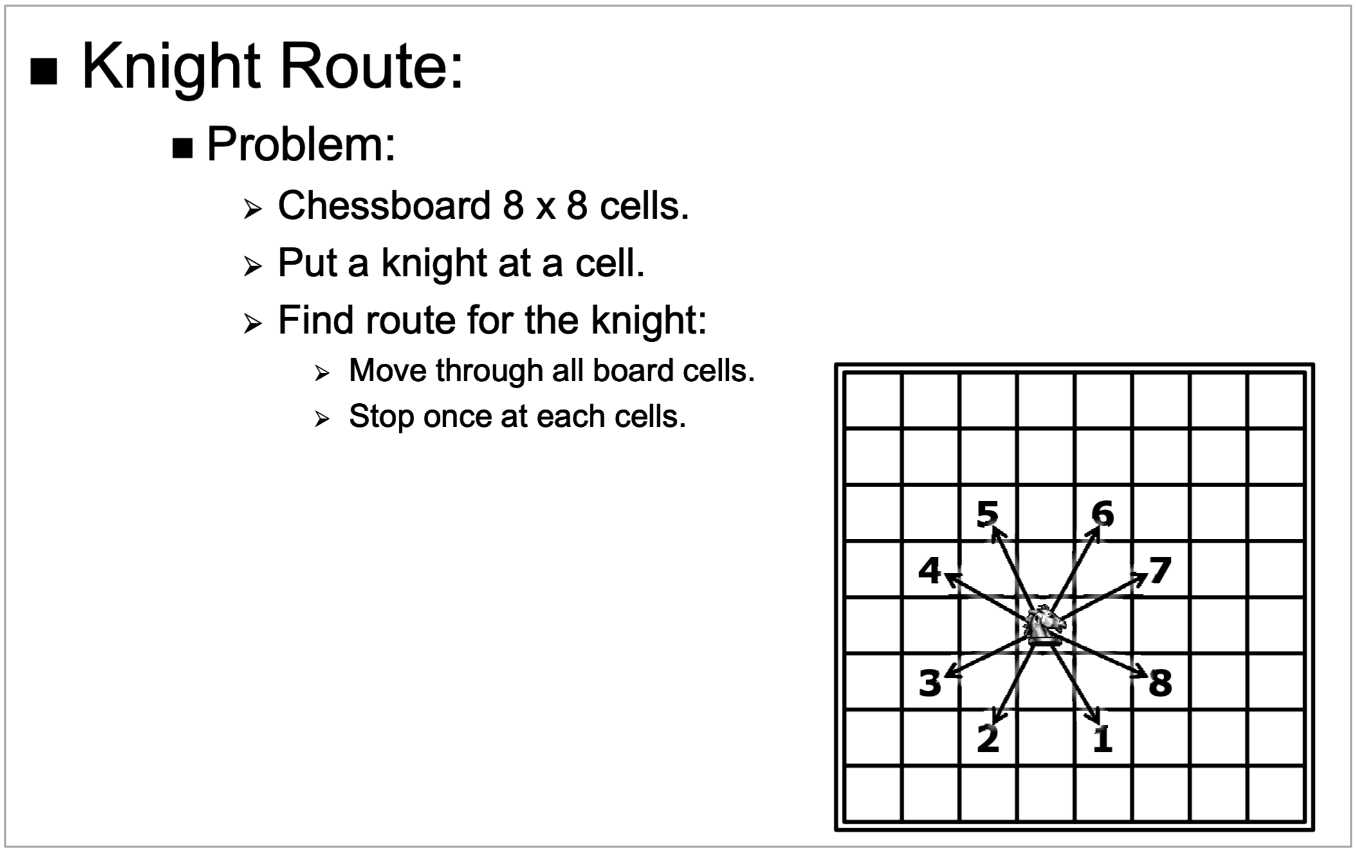
Input: N: a chessboard with size N * N.

Output: location (i, j) of N queens in that chessboard. Print all solutions

3.3 Knight Tour

Input: N: a chessboard with size N * N.

Output: the movement orders of the knight (1 to N*N) in that chessboard. Only 1 solution is required.



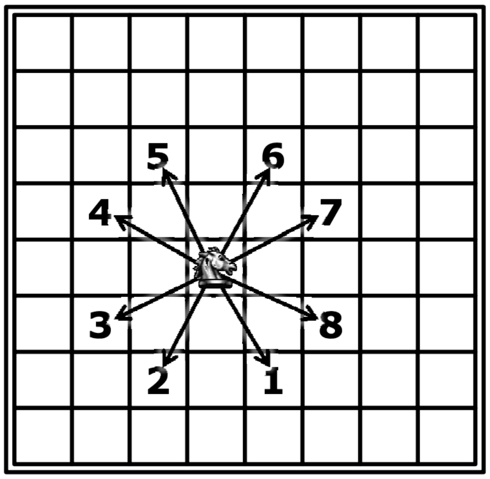
- Knight Route:
  - Problem:
    - Chessboard 8 x 8 cells.
    - Put a knight at a cell.
    - Find route for the knight:
      - Move through all board cells.
      - Stop once at each cells.

Knight Route:

Analysis:

- Can only move to unoccupied cells.
- If knight at (i, j), which cells can move next.

1: (i + 2, j + 1)
2: (i + 2, j – 1)
3: (i + 1, j – 2)
4: (i – 1, j – 2)
5: (i – 2, j – 1)
6: (i – 2, j + 1)
7: (i – 1, j + 2)
8: (i + 1, j + 2)



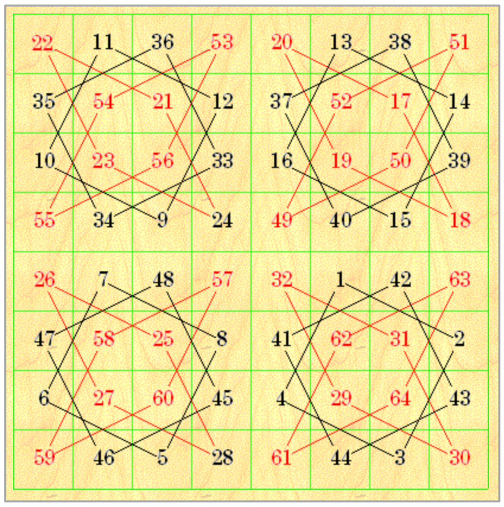
Knight Route:

Backtracking:

```
TryKnight( cell (i, j), board state, step )
{
    if ( cell (i, j) is occupied )
        return;

    Update board state;

    if ( is last step )
        Print result;
    else
        TryKnight( cell (i + 2, j + 1), board state, step + 1 );
        TryKnight( cell (i + 2, j – 2), board state, step + 1 );
        …
    Roll back board state;
}
```



22	11	36	53	20	13	38	51
35	54	21	12	37	52	17	14
10	23	56	33	16	19	50	39
55	34	9	24	49	40	15	18
26	7	48	57	32	1	42	63
47	58	25	8	41	62	31	2
6	27	60	45	4	29	64	43
59	46	5	28	61	44	3	30

3.4 Knight Tour

Input: N: a chessboard with size N * N, a start cell (istart, jstart) and an end cell (iend, jend).

Output: the movement orders of the knight (1 to N*N) in that chessboard. Only 1 solution is required.

For Codeforces/Hackerrank/Leetcode… assignments, you need to submit 2 files: a source code cpp file, and an image/screenshot to prove that your code is ACCEPTED on Codeforces/Hackerrank/Leetcode.

3.5 Tribonacci

You must write a recursive solution and your source code must be accepted.

https://leetcode.com/problems/n-th-tribonacci-number/

3.6 Permutation of N

Given N, print all permutation from 1 to N.

Example:

+ Input: N = 3

+ Output: [[1,2,3],[1,3,2],[2,1,3],[2,3,1],[3,1,2],[3,2,1]]

3.7 Permutation of an array

Given an array, the task is to print or display all the permutations of this array.

https://leetcode.com/problems/permutations/

3.8 Permutation of an array without duplicates

https://leetcode.com/problems/permutations-ii/

3.9 Next permutation

https://leetcode.com/problems/next-permutation/

3.10 Permutation of a string

Write a program to print all permutations of a given string.

3.11 Check permutation

https://leetcode.com/problems/permutation-in-string/

3.12 Letter Case Permutation

https://leetcode.com/problems/letter-case-permutation/