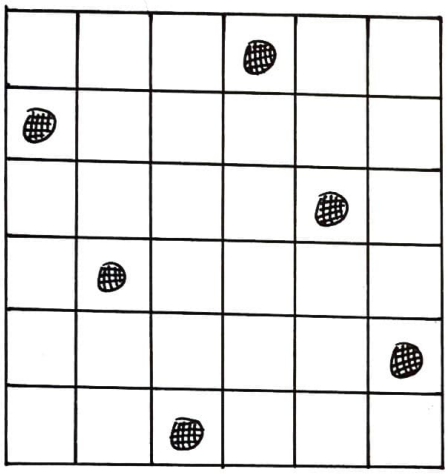
Textbook 5.1

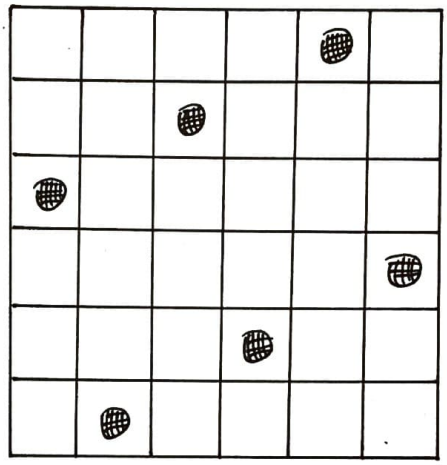
41084316
詹禎文

for $n = 6 \Rightarrow 6 \times 6$ matrix

SOL1 :



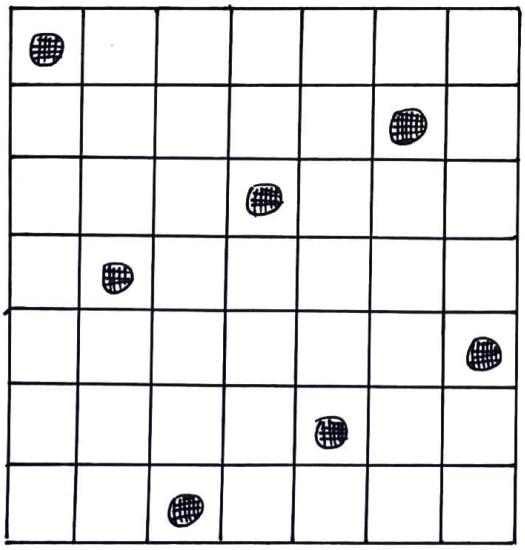
SOL2 :



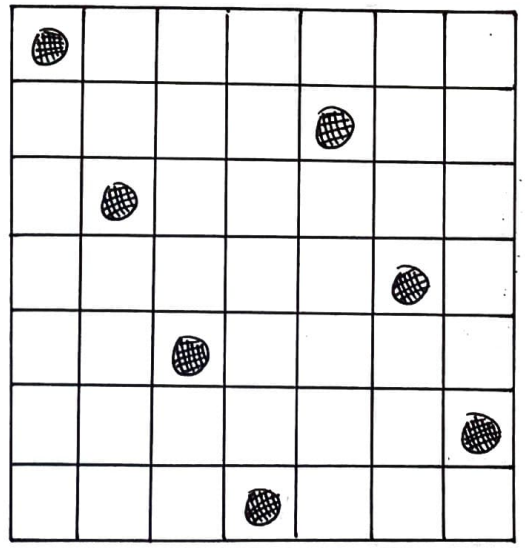
※

for $n = 7 \Rightarrow 7 \times 7$ matrix

SOL1 :



SOL2 :



※

Textbook 5.7

```
void Solution ( b[n][n] )
    for ( i = 0 ; i < n ; i++ )
        for ( j = 0 ; j < n ; j++ )
            cout << b[i][j] ;
```

⇒ Output solution

```
bool CheckPlace ( b[n][n] , row , col )
    for ( i = 0 ; i < col ; i++ )
        if ( b[row][i] ) return false ;
    for ( i = row , j = col ; i >= 0 and j >= 0 ; i-- , j-- )
        if ( b[i][j] ) return false ;
    for ( i = row , j = col ; j >= 0 and i < n ; i++ , j-- )
        if ( b[i][j] ) return false ;
    return true ;
```

⇒ Check if the queen can be placed at specific position

```
bool Queen ( b[n][n] , col )
    if ( col >= n ) return true ;
    for ( i = 0 ; i < n ; i++ )
        if ( CheckPlace ( b , i , col ))
            b[i][col] = 1 ;
            if ( Backtracking ( b , col+1 ) == true ) return true ;
            b[i][col] = 0 ;
    return false ;
```

⇒ Solve the n-queen problem (Called by "Backtracking" function)

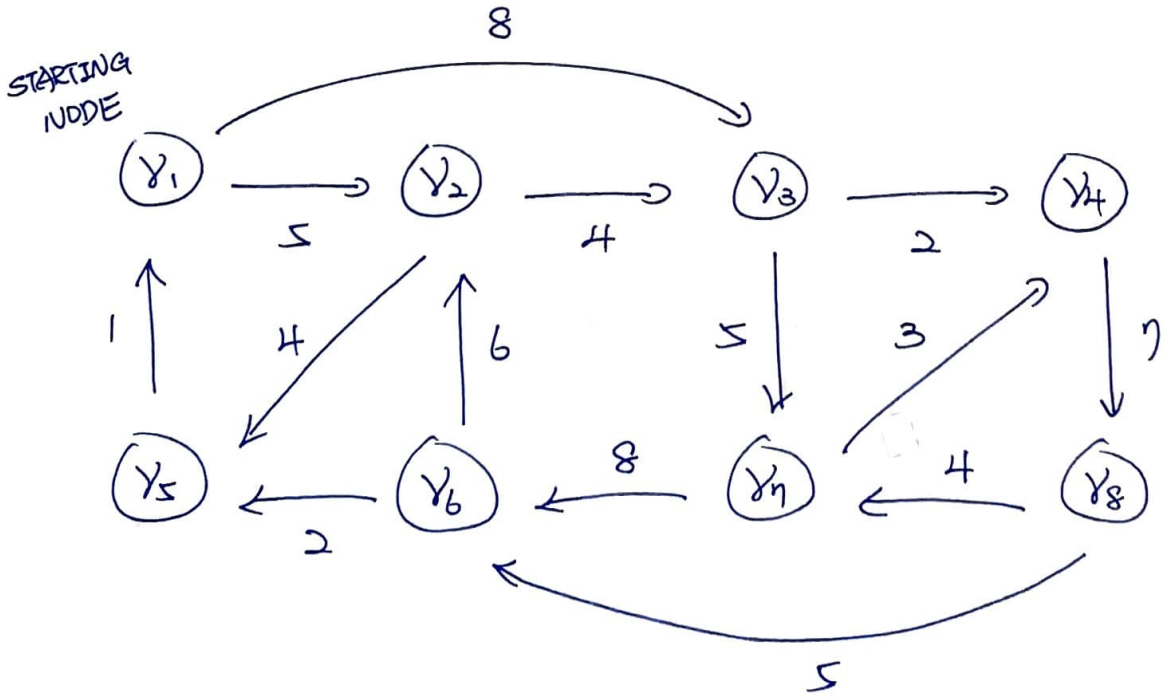
```
bool  Backtracking (n)
    int  b[n][n] = {} ;
    if ( Queen ( b , 0 ) == false )
        cout << " No Solution " << endl ;
        return false ;
    Solution (b) ;
    return true ;
```

⇒ Backtracking method for solving n-queen (Called by " Queen " function)

```
int  main ()
    Backtracking (n) ;
    return 0 ;
```

⇒ Main function

Textbook 6.7



STEP 1 :

$v_1 \rightarrow v_2 = 5$

$v_1 \rightarrow v_3 = 8$

$\Rightarrow$ $v_1 \xrightarrow{5} v_2$

STEP 2 :

$v_2 \rightarrow v_3 = 4$

$v_2 \rightarrow v_5 = 4$ (No possible to cover all vertices)

$\Rightarrow$ $v_1 \xrightarrow{5} v_2 \xrightarrow{4} v_3$

STEP 3 :

$v_3 \rightarrow v_4 = 2$

$v_3 \rightarrow v_7 = 5$

$\Rightarrow$ $v_1 \xrightarrow{5} v_2 \xrightarrow{4} v_3 \xrightarrow{2} v_4$

STEP 4 :

$v_4 \rightarrow v_8 = 7$ (Only possible route)

$\Rightarrow$ $v_1 \xrightarrow[5]{} v_2 \xrightarrow[4]{} v_3 \xrightarrow[2]{} v_4 \downarrow 7$ v_8

STEP 5 :

$v_8 \rightarrow v_7 = 4$

$v_8 \rightarrow v_6 = 5$

$\Rightarrow$ $v_1 \xrightarrow{5} v_2 \xrightarrow{4} v_3 \xrightarrow{2} v_4 \downarrow 7$, $v_8 \xleftarrow[4]{} v_7$

STEP 6 :

$v_7 \rightarrow v_4 = 3$ (No repeated vertex)

$v_7 \rightarrow v_6 = 8$

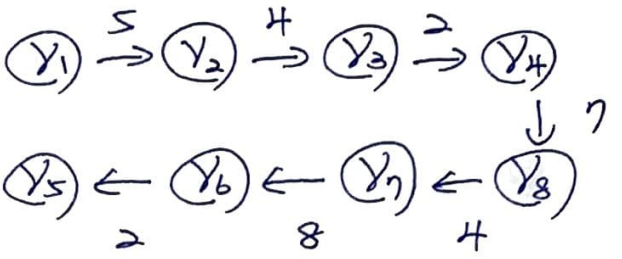
$\Rightarrow$ $v_1 \xrightarrow{5} v_2 \xrightarrow{4} v_3 \xrightarrow{2} v_4 \downarrow 7$, $v_8 \xleftarrow{4} v_7 \xleftarrow{8} v_6$

STEP 7 :

$v_6 \to v_2 = 6$

$v_6 \to v_5 = 2$

$\Rightarrow$

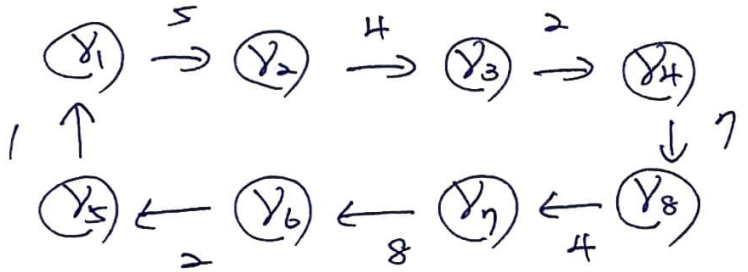


STEP 8 :

$v_5 \to v_1 = 1$ (Go back to starting node)

(Check all vertices are covered)

$\Rightarrow$



$\therefore$ Length of the optimal tour

$= 5 + 4 + 2 + 7 + 4 + 8 + 2 + 1$

$= 33$ ✱