lab06

```
$ gcc lab06.c
$ ./a.out
Solution 1:
1 2 3 4 5
  1 2 3 4 5
2 1 4 5 3
  2 1 4 5 3
3 4 5 1 2
  3 4 5 1 2
4 5 2 3 1
  4 5 2 3 1
5 3 1 2 4
  5 3 1 2 4
...
...
Solution 161280:
5 4 3 2 1
  5 4 3 2 1
4 5 2 1 3
  4 5 2 1 3
3 2 1 5 4
  3 2 1 5 4
2 1 4 3 5
  2 1 4 3 5
1 3 5 4 2
  1 3 5 4 2
Total number solutions found: 161280

CPU time: 0.657178 sec
score: 86
o. [Output] Program output is incorrect
o. [Format] Program format can be improved
o. [Efficiency] can still be improved.
o. [Comments] can be more helpful to explain your codes.
```

lab06.c

```
// EE2310 lab06 Latin Squares
// 109061217 林峻霆
// Date: 2020/11/9

#include <stdio.h>

#define N 5

int A[N][N];                                           // a latin square array
int Nsol = 0;                                          // amount of solution

void latin_square(int row, int col);                   // declare function

int main(void)
{
    latin_square(0, 0);                                // run the function
    printf("Total number solutions found: %d\n", Nsol);// print the result
    printf("Total number solutions found: %d\n", Nsol);  // print the result

    return 0;                                          // end the program
}

void latin_square (int row, int col) {
void latin_square(int row, int col)
Comments?
{
    int k;                                             // parameter for loop
    int i = 0;                                         // parameter for loop
    int j = 0;                                         // parameter for loop
Need a blank line here.
    for (k = 1; k <= N; k++) {
        for (i = 0; i < row && k != A[i][col]; i++);   // check for top blocks
        for (j = 0; j < col && k != A[row][j]; j++);   // check for left blocks

        if (i == row && j == col) {
            A[i][j] = k;
                                                       // if we can fill in the
            if (row == N - 1 && col == N - 1) {        // last block print resul
t
This line has more than 80 characters
```

```c
                Nsol += 1;
                printf("Solution %d:\n", Nsol);
                for (i = 0; i < N; i++) {
                    for (j = 0; j < N; j++) {
                        if (j == N - 1)
                            printf("%d\n", A[i][j]);
                        else
                            printf("%d ", A[i][j]);
                    }
                }
            }
            else if (col == N - 1) {                   // if we fill in last
                latin_square(row + 1, 0);              // block in row, go to
            }                                          // next row
            else {
                latin_square(row, col + 1);            // fill in next block
            }
        }
    }
}
```