lab06

```
$ gcc -DN=11 lab06.c

$ a.out < mat11.in
Matrix A is
  11  10  9  8  7  6  5  4  3  2  1
  10  11  10  9  8  7  6  5  4  3  2
  9  10  11  10  9  8  7  6  5  4  3
  8  9  10  11  10  9  8  7  6  5  4
  7  8  9  10  11  10  9  8  7  6  5
  6  7  8  9  10  11  10  9  8  7  6
  5  6  7  8  9  10  11  10  9  8  7
  4  5  6  7  8  9  10  11  10  9  8
  3  4  5  6  7  8  9  10  11  10  9
  2  3  4  5  6  7  8  9  10  11  10
  1  2  3  4  5  6  7  8  9  10  11
det(A) = 6144
CPU time: 1.81072 sec
```

---

score: 96.0

o. [Output] Program output is correct, good.

o. [Format] Program format can be improved.

o. [Comments] should explain the functions more clearly.

lab06.c

```c
// EE231002 Lab06. Matrix Determinant
// 110060007, 黃俊穎
// 2021/11/18

#include <stdio.h>
#if !defined(N)
#define N 3
#endif

double det(double A[N][N], int dim);    // determinant function declaration

int main(void)                          // start main function
{
    int row, column;                    // variables of array
    double A[N][N];

    // print out the result
    printf("Matrix A is\n");
    for (row = 0; row < N; row++) {
        for (column = 0; column < N; column++) {
            scanf("%lg", &A[row][column]);
            printf("  %lg", A[row][column]);
        }
        printf("\n");
    }
    printf("det(A) = %lg\n", det(A, N)); // print out determinant

    return 0;                            // finish main function
}

double det(double A[N][N], int dim)
Comments?
{
    int i, j, k;                         // counters in following loops
    double sub[N][N];                    // save recursive arrays
    double sum = 0;                      // initialize of determinant
    int sign = 1;                        // change '+','-' in sum
    int row;                             // row for input array

    if (dim == 1) {
```

```
40          sum = A[0][0];                    // way of 1*1 determinant
41      }
42      else if (dim == 2) {
43                                            // way of 2*2 determinant
44          sum = A[0][0] * A[1][1] - A[0][1] * A[1][0];
45      } else {
46          for (i = 0; i < dim; i++) {       // decide title number
47              row = 0;                      // prevent title to be other column
48              for (j = 0; j < dim; j++) {
49                  if (j != i) {             // subarray's row downgrade
50                      for (k = 1; k < dim; k++) {
51                          sub[row][k - 1] = A[j][k];
52                                            // subarray's column downgrade
53                      }
54                      row++;                // store value for next subarray
55                  }
56              }
57              // formula to calculate determinant
58              sum += sign * A[i][0]* det(sub, dim - 1);
                sum += sign * A[i][0] * det(sub, dim - 1);
59              // subarray determinant will be positive and negative
60              sign *= -1;
61          }
62      }
63      return sum;                          // return sum value to main function
64  }
```