

2장 프로그래밍 실습_C 코드

[프로그래밍 실습 2]

```
#include<stdio.h>
#include<Windows.h>

#pragma warning(disable : 4996)

void main()
{
    int i;
    int input;

    // 사용자로부터 정수를 입력받음
    printf("입력에 대한 소수 여부를 판단한다.\n");
    printf("소수 판별하기 위한 N값을 입력하라.: ");
    scanf("%d", &input);

    // 2부터 input-1까지의 정수들로 input을 나누면서 소수인지 확인
    for (i = 2; i < input; i++)
    {
        // input이 짝수이면 바로 결과 출력
        if (input % i == 0 && input != i)
        {
            printf("\n%5d는 소수가 아니다!!\n", input);
            system("PAUSE");
            return;
        }
    }
    // 마지막 결과를 출력
    if (input > 1)
    {
        printf("\n%5d는 소수이다!!\n", input);
    }
    else
    {
        printf("\n%5d는 소수가 아니다!!\n", input);
    }
    system("PAUSE");
}
```

[프로그래밍 실습 5]

```
#include<stdio.h>
#include<Windows.h>

int main()
{
    int i, j, k;
    int arr_1[4][4]={1,2,3,4},{4,3,2,1},{2,5,7,9},{6,3,2,1} }; // 정방행렬 arr_1
    int arr_2[4][4]={1,5,6,7},{8,3,1,7},{6,2,8,3},{9,2,1,2} }; // 정방행렬 arr_2
    int res[4][4] = { {0} };

    // 정방행렬 arr_1을 출력
    printf("행렬 1 : \n");
    for (i = 0; i < 4; i++)
    {
        printf(" ");
        for (j = 0; j < 4; j++)
        {
            printf("%4d", arr_1[i][j]);
        }
        printf("\n");
    }
    printf("\n\n");

    // 정방행렬 arr_2를 출력
    printf("행렬 2 : \n");
    for (i = 0; i < 4; i++)
    {
        printf(" ");
        for (j = 0; j < 4; j++)
        {
            printf("%4d", arr_2[i][j]);
        }
        printf("\n");
    }

    // 정방행렬 res를 0으로 초기화
    for (i = 0; i < 4; i++)
    {
        for (j = 0; j < 4; j++)
        {
            res[i][j] = 0;
        }
    }

    // 삼중 for 문은 (i, j, k 순으로) 정방행렬 arr_1과 arr_2를 곱함
    printf("행렬 1 * 행렬 2 (i, j, k 순으로) : \n");
    for (i = 0; i < 4; i++)
    {
```

```

        for (j = 0; j < 4; j++)
        {
            for (k = 0; k < 4; k++)
            {
                res[i][j] += arr_1[i][k] * arr_2[k][j];
            }
        }
    }

    // 정방행렬 res를 출력
    for (i = 0; i < 4; i++)
    {
        printf(" ");
        for (j = 0; j < 4; j++)
        {
            printf("%4d", res[i][j]);
        }
        printf("\n");
    }

    // 정방행렬 res를 0으로 재초기화
    for (i = 0; i < 4; i++)
    {
        for (j = 0; j < 4; j++)
        {
            res[i][j] = 0;
        }
    }

    // 삼중 for 문은 (i, k, j 순으로) 정방행렬 arr_1과 arr_2를 곱함
    printf("행렬 1 * 행렬 2 (i, k, j 순으로) : \n");
    for (i = 0; i < 4; i++)
    {
        for (k = 0; k < 4; k++)
        {
            for (j = 0; j < 4; j++)
            {
                res[i][j] += arr_1[i][k] * arr_2[k][j];
            }
        }
    }

    // 정방행렬 res를 출력
    for (i = 0; i < 4; i++)
    {
        printf(" ");
        for (j = 0; j < 4; j++)
        {
            printf("%4d", res[i][j]);

```

```

        }
        printf("\n");
    }
    printf("\n\n결과는 같다. \n");
    system("PAUSE");
    return 0;
}

```

[프로그래밍 실습 6]

```

#include<stdio.h>
#include<conio.h>
#include<Windows.h>
#pragma warning (disable : 4996)

const max = 3; // 행렬의 차수를 위한 상수
// 3x3 행렬을 위한 자료형
typedef int matrix[3][3];
matrix c, d, e;

// 함수 원형
void mat_compute(matrix, matrix);
void print_matrix(matrix);

void main()
{
    int i, j;
    matrix a, b;
    putchar('\n');
    printf("3x3인 부울행렬 A를 입력 \n(0과 1만 사용함)\n");
    printf("예) 0 1 0\n    0 0 0\n    1 0 1\n\n");

    // 사용자가 입력할 부울행렬을 a에 저장
    for (i = 0; i < max; i++)
    {
        for (j = 0; j < max; j++)
        {
            scanf("%d", &a[i][j]);
        }
    }
    printf("\n 3x3인 부울행렬 B를 입력\n");
    // 사용자가 입력할 부울행렬을 b에 저장
    for (i = 0; i < max; i++)
    {
        for (j = 0; j < max; j++)
        {
            scanf("%d", &b[i][j]);
        }
    }
}

```

```

// 부울행렬 a와 b에 대해 접합, 교합, 부울곱을 계산하는 함수 호출
mat_compute(a, b);
putchar('\n');

// 접합 결과를 출력
printf("A MEET B\n");
print_matrix(c);
putchar('\n');
// 교합 결과를 출력
printf("A JOIN B\n");
print_matrix(d);
putchar('\n');
// 부울곱 결과를 출력
printf("A Boolean product B\n");
print_matrix(e);
system("PAUSE");
}

// mat_compute 함수가 a와 b에 대해 접합, 교합, 부울곱 계산
void mat_compute(matrix a, matrix b)
{
    int i, j, k;
    // 접합, 교합, 부울곱 계산을 위해 삼중 for 문 사용
    for (i = 0; i < max; i++)
    {
        for (j = 0; j < max; j++)
        {
            for (k = 0; k < max; k++)
            {
                c[i][j] = a[i][j] & b[i][j]; // 접합
                d[i][j] = a[i][j] | b[i][j]; // 교합
                // 부울곱

                if (k == 0)
                {
                    e[i][j] = a[i][k] * b[k][j];
                }
                else
                {
                    e[i][j] = e[i][j] | (a[i][k] & b[k][j]);
                }
            }
        }
    }
}

// print_matrix 함수가 부울행렬 x를 출력
void print_matrix(matrix x)
{

```

```

int i, j;
for (i = 0; i < max; i++)
{
    putchar('\n');
    for (j = 0; j < max; j++)
    {
        printf("%2d", x[i][j]);
    }
    putchar('\n');
}

```

[프로그래밍 실습 7]

```

#include <stdio.h>
#include <process.h>
#pragma warning (disable : 4996)

#define MAX 3
typedef float mxtype[MAX][MAX];
void matrixout(mxtype);
void main() {
    mxtype mxa, inversemx;
    int i, j, l, n;
    float sumx, sumy;
    float determins;
    printf("3x3인 정방행렬 A를 입력\n");
    printf("ex) 1 2 3\n    4 5 6\n    7 8 9\n\n");

    // 사용자가 입력할 정방행렬을 읽음
    for (i = 0; i < MAX; i++)
        for (j = 0; j < MAX; j++) {
            scanf("%f", &mx[i][j]);
        }

    // 행렬 A를 출력
    printf("%5cmatrix A\n", ' ');
    matrixout(mxa);

    sumx = 0;
    sumy = 0;
    determins = 0;
    // 행렬 A의 행렬식 계산
    for (j = 0; j < MAX; j++) {
        l = (j + 1) % MAX;
        n = (j + 2) % MAX;
        sumx += mxa[0][j] * mxa[1][l] * mxa[2][n];
        sumy += mxa[0][j] * mxa[1][n] * mxa[2][l];
    }
}

```

```

    }

    determinins = sumx - sumy;
    putchar('\n');
    printf("%5cdeterminant%7c:%15E\n", ' ', ' ', determinins);

    // 행렬식이 0 이면 역행렬을 계산하지 못함
    if (determinins == 0) {
        printf("%5cinverse matrix does not exist\n", ' ');
        exit(1);
    }

    // 정방행렬 A 의 역행렬 계산
    for (i = 0; i < MAX; i++)
        for (j = 0; j < MAX; j++) {
            if (i == j) {
                switch (i) {
                    case 0:
                        inversemx[0][0]=mxa[1][1]*mxa[2][2]-mxa[1][2]*mxa[2][1];
                        break;
                    case 1:
                        inversemx[1][1]=mxa[0][0]*mxa[2][2]-mxa[0][2]*mxa[2][0];
                        break;
                    case 2:
                        inversemx[2][2]=mxa[0][0]*mxa[1][1]-mxa[0][1]*mxa[1][0];
                        break;
                }
            }
            else {
                l = i + j;
                switch (l) {
                    case 1: l = 2; break;
                    case 2: l = 1; break;
                    case 3: l = 0; break;
                }
                inversemx[i][j]=(mxa[i][l]*mxa[l][j]-mxa[i][j]*mxa[l][l])*1.0;
            }
        }

    // 계산된 역행렬을 출력
    for (i = 0; i < MAX; i++)
        for (j = 0; j < MAX; j++)
            inversemx[i][j] /= determinins;
    printf("\n%5cinverse matrix\n", ' ');
    matrixout(inversemx);
}

// matrixout 함수가 정방행렬 mx를 출력
void matrixout(mxtype mx) {
    int i, j;
    putchar('\n');

```

```
    for (i = 0; i < MAX; i++) {  
        for (j = 0; j < MAX; j++)  
            printf("%15E ", mx[i][j]);  
        printf("\n");  
    }  
}
```