

8장 프로그래밍 실습_C 코드

[프로그래밍 실습 1]

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

#define domain 5          // 집합의 크기

void readfile(char fn[13], char adjancy[domain + 1][domain + 1]);

void main()
{
    // 인접행렬을 위한 2차원 배열 정의 및 초기화
    char adjancymx[domain + 1][domain + 1] = { 0 };
    char fn[13] = "pp4-1.dat";      // 파일 이름 초기화
    int i, j, degree;              // 차수를 위한 변수

    // 파일을 읽고 데이터를 adjancymx에 대입하여 관계를 출력
    readfile(fn, adjancymx);
    printf("\b\b}\n\n");

    // 각 정점에 대한 차수를 계산해서 출력
    printf("%5c정점의 차수\n\n", ' ');
    for (i = 1; i <= domain; i++)
    {
        degree = 0;                // 차수 변수를 재초기화
        // 정점 i에 대한 차수 계산
        for (j = 1; j <= domain; j++)
        {
            if (adjancymx[i][j] == 1)
            {
                degree++;
            }
        }
        printf("%5c정점 %d의 차수는 %2d \n", ' ', i, degree);
    }

    // 인접행렬 출력
    printf("\n\n%7c인접행렬\n%5c\u250c          \u2510\n", ' ', ' ');
    for (i = 1; i <= domain; i++)
    {
        printf("%5c| ", ' ');
        for (j = 1; j <= domain; j++)
        {
            printf("%d ", adjancymx[i][j]);
        }
        printf("| \n");
    }
    printf("%5c\u2514          \u2518\n", ' ');
    system("PAUSE");
}

// readfile 함수가 텍스트 파일에서 관계를 읽고 adjacency에 대입
void readfile(char fn[13], char adjancy[domain + 1][domain + 1])
{
```

```

FILE* fp;
int x, y;
fp = fopen(fn, "r"); // 파일 열기

printf("입력받은 관계 R의 출력이다.\n");
printf("R = {");
while (!feof(fp))
{
    fscanf(fp, "%d %d", &x, &y);
    adjancy[x][y] = 1;
    printf("(%d,%1d), ", x, y);
}
fclose(fp);
}

```

[프로그래밍 실습 3]

```

#include<stdio.h>
#include<conio.h>
#include<Windows.h>
#pragma warning (disable : 4996)
#define max 5 // 정점의 수
#define n 6 // 입력할 간선의 수
void main()
{
    int b[n + 1][3] = { 0 }; // 입력할 간선들을 저장하기 위한 (nx3) 이차원 배열 선언 및 초기화
    // 입력할 간선들을 저장하기 위한 (nxn) 인접행렬 선언 및 초기화
    int a[max + 1][max + 1];
    int ord[max + 1] = { 0 }; // 차수 기준으로 정점들을 내림차순으로 저장할 배열 선언 및 초기화
    int deg[max + 1] = { 0 }; // 각 정점의 차수에 대한 배열 선언 및 초기화
    int col[max + 1] = { 0 }; // 각 정점의 색에 대한 배열 선언 및 초기화
    int temp, c, cnt, i, j, k;

    // 5개 정점으로 구성된 그래프의 간선들을 입력받고 a와 b에 대입
    printf("정점이 5개인 그래프이다.(정점의 라벨은 1~5이다.)\n");
    printf("간선을 1 2와 같은 순서쌍으로 6개 입력하라.\n");
    for (i = 1; i <= n; i++)
    {
        scanf("%d %d", &b[i][1], &b[i][2]);
        a[b[i][1]][b[i][2]] = 1;
        a[b[i][2]][b[i][1]] = 1;
    }

    // 각 정점에 대한 차수를 계산해서 배열 deg에 대입
    for (i = 1; i <= max; i++)
    {
        for (j = 1; j <= max; j++)
        {
            if (a[i][j] == 1)
            {
                deg[i]++;
            }
        }
    }

    // 계산된 차수의 값들을 출력

```

```

printf("\n\n%5c각 정점에 대한 차수는 다음과 같다.\n", ' ');
for (i = 1; i <= max; i++)
{
    printf("%5c정점 %d의 차수는 %2d\n", ' ', i, deg[i]);
}

// ord 배열을 초기화
for (i = 1; i <= max; i++)
{
    ord[i] = i;
}

// 정점들을 차수에 따라 내림차순으로 정렬
for (i = 1; i <= max - 1; i++)
{
    for (j = i + 1; j <= max; j++)
    {
        if (deg[ord[i]] < deg[ord[j]])
        {
            temp = ord[i];
            ord[i] = ord[j];
            ord[j] = temp;
        }
    }
}

// 각 정점을 착색하는 반복문
for (int i = 1; i <= max; i++) {
    int v = ord[i];

    // 현재 정점에서 착색되지 않은 최소의 색을 찾음
    int usedColors[max + 1] = { 0 };
    for (int adjVertex = 1; adjVertex <= max; adjVertex++) {
        if (a[v][adjVertex] == 1 && col[adjVertex] != 0)
            usedColors[col[adjVertex]] = 1;
    }

    // 사용하지 않은 최소의 색을 현재 정점에 대입
    for (int c = 1; c <= max; c++) {
        if (!usedColors[c]) {
            col[v] = c;
            break;
        }
    }
}

// 색의 수를 세기
int col_num = -1;
for (i = 1; i <= max - 1; i++)
{
    if (col[i] >= col_num)
        col_num = col[i];
}

// 착색된 그래프에 대한 정보를 출력
printf("\n\n%5c각 정점에 대한 차수는 다음과 같다.\n\n", ' ');
for (int colors = 1; colors <= col_num; colors++)
{
    printf("%5c정점 ", ' ');
    for (int j = 1; j <= max; j++) {
        if (col[j] == colors)

```

```
        printf("%d, ", j);
    }
        printf("\b\b (들)이");
        printf(" %d번째 색으로 착색됨\n", colors);
    }
    printf("\n%5c이 그래프는 %d색 가능하다.\n\n", ' ', col[i]);
    system("PAUSE");
}
```