

12장 프로그래밍 실습_C 코드

[프로그래밍 실습 1]

```
#include<stdio.h>
#include<conio.h>
#include<time.h>
#include<string.h>
#include<Windows.h>
#pragma warning(disable : 4996)
#define max11 3 // T1에 입력될 문장의 개수를 위한 상수
#define max12 101 // T1의 각 문자열의 길이를 위한 상수
#define max21 3 // T2에 입력될 문장의 개수를 위한 상수
#define max22 101 // T2의 각 문자열의 길이를 위한 상수

void main()
{
    int i, j, k, l;
    char t1[max11][max12 + 1]; // T1에 입력될 문장을 위한 문자열 선언
    char t2[max21][max22 + 1]; // T2에 입력될 문장을 위한 문자열 선언
    char t1t2[max11 * max21][max12 + max22 + 1]; // 접합 T1·T2를 저장하기 위한 문자열 선언
    char t2t1[max11 * max21][max12 + max22 + 1]; // 접합 T2·T1을 저장하기 위한 문자열 선언

    // 3개 문장을 입력받고 T1에 추가
    printf("\n 문자열 T1에 대해 3개의 문자열을 입력하라.\n");
    for (i = 0; i < 3; i++)
    {
        fflush(stdin);
        printf("\u25cf ");
        scanf("%s", t1[i]);
    }
    // 3개 문장을 입력받고 T2에 추가
    printf("\n 문자열 T2에 대해 3개의 문자열을 입력하라.\n");
    for (i = 0; i < 3; i++)
    {
        fflush(stdin);
        printf("\u25cf ");
        scanf("%s", t2[i]);
    }
    // T1에 있는 각 문장과 T2에 있는 각 문장의 접합을 T1·T2에 추가
    printf("\n 접합 T1 \u25cf T2\n");
    k = 0;
    for (j = 0; j < max21; j++)
    {
        for (i = 0; i < max11; i++)
        {
            strcpy(t1t2[k], t1[i]);
            strcat(t1t2[k], t2[j]);
            k++;
        }
    }
    // 접합 T1·T2에 대한 결과를 출력
    for (i = 0; i < max11 * max21; i++)
    {
        printf(" %s\n", t1t2[i]);
    }
    printf("\n 접합 T2 \u25cf T1\n");
```

```

k = 0;
// T2에 있는 각 문장과 T1에 있는 각 문장의 접합을 T2·T1에 추가
for (j = 0; j < max11; j++)
{
    for (i = 0; i < max21; i++)
    {
        strcpy(t2t1[k], t2[i]);
        strcat(t2t1[k], t1[j]);
        k++;
    }
}
// 접합 T2·T1에 대한 결과를 출력
for (i = 0; i < max11 * max21; i++)
{
    printf(" %s\n", t2t1[i]);
}
system("PAUSE");
}

```

[프로그래밍 실습 2]

```

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

void main()
{
    int i, j, k, l, g;
    char t1[5][20001] = { NULL }; // 랜덤 형식으로 생성될 길이가 20000인 문자열 5개를 위한
    문자열 선언
    char t2[3][101] = { NULL }; // T2에 입력될 문장 3개를 위한 문자열 선언
    char t1t2[15][20102]; // 접합 T1·T2를 저장하기 위한 문자열 선언
    char temp = '\0';
    double start, end;
    FILE* fp1, * fp2, * fp3; // 파일 포인터 3개 선언
    fp1 = fopen("T1data.txt", "w+"); // 랜덤 데이터를 저장하기 위한 텍스트 파일 생성 및
    열기
    fp2 = fopen("pointer_result.txt", "w+"); // 첫 번째 알고리즘으로 접합을 저장하기 위한
    텍스트 파일 생성 및 열기
    fp3 = fopen("notpoint_result.txt", "w+"); // 두 번째 알고리즘으로 접합을 저장하기 위한
    텍스트 파일 생성 및 열기

    // 길이가 20000인 랜덤 문자열 5개를 생성하고 텍스트 파일에 저장
    printf("\n문자열 T10| \"T1data.txt\" 파일에서 생성되었다.\n");
    for (i = 0; i < 5; i++)
    {
        for (j = 0; j < 20000; j++)
        {
            temp = rand() % 26 + 65;
            fprintf(fp1, "%c", temp);
            t1[i][j] = temp;
        }
        t1[i][j] = '\0';
        fprintf(fp1, "\n");
    }
}

```

```

}
// 3개 문장을 입력받고 T2에 추가
printf("\n 문자열 T2에 대해 3개의 문자열을 입력하라.\n");
for (i = 0; i < 3; i++)
{
    fflush(stdin);
    printf("\u25cf ");
    scanf("%s", t2[i]);
}
// T2에 대한 Image를 출력
printf("\nImage of T2\n");
for (i = 0; i < 3; i++)
{
    printf("< %s>\n", t2[i]);
}
printf("\n 페인트공 알고리즘 : T1\u25cfT2");
printf("\n \"pointer_result.txt\" 파일이 생성되었다.");
start = clock();
for (g = 0; g < 100; g++)
{
    k = 0;
    for (j = 0; j < 3; j++)
    {
        for (i = 0; i < 5; i++)
        {
            strcpy(t1t2[k], t1[i]);
            strcat(t1t2[k], t2[j]);
            k++;
        }
    }
}
end = clock();
for (i = 0; i < 15; i++)
{
    fprintf(fp2, "%s\n", t1t2[i]);
}
printf(" 실행 시간 : %.3lf(ms)\n", (end - start) / 100.0);

// 집합 T1·T2를 계산하는 ‘반환값을 주는 포인터 없는 알고리즘’의 실행 시간으로 출력하고
// 집합을 텍스트 파일에 저장
printf("\n 반환값을 주는 포인터 없는 알고리즘 : T1\u25cfT2");
printf("\n \"notpointer_result.txt\" 파일이 생성되었다.\n");
temp = '\0';
start = clock();
for (g = 0; g < 100; g++)
{
    k = 0;
    for (j = 0; j < 3; j++)
    {
        for (i = 0; i < 5; i++)
        {
            for (l = 0; l < 20001; l++)
            {
                if (t1[i][l] == '\0')
                {
                    strcpy(t1t2[k], t1[i]);
                }
            }
            strcat(t1t2[k], t2[j]);
        }
    }
}

```

```

        k++;
    }
}
end = clock();
for (i = 0; i < 15; i++)
{
    fprintf(fp3, "%s\n", t1t2[i]);
}
printf(" 실행 시간 : %.3lf(ms)\n", (end - start) / 100.0);

// 생성된 파일들을 닫음
fclose(fp1);
fclose(fp2);
fclose(fp3);
system("PAUSE");
}

```

[프로그래밍 실습 3]

```

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

// 상태를 위한 구조체
typedef struct
{
    int m;
    int c;
    char p;
}state;

// 상태를 초기화하기 위한 함수
void state_init(state* s)
{
    s->c = 2;
    s->m = 2;
    s->p = 'r';
}

// 상태 전이를 진행하는 함수
int state_moving(state* s, int m, int c)
{
    if (s->p == 'r')        // 배가 오른쪽에 있을 때
    {
        // 현재 상태가 (2, 2, R)일 때
        if (s->m == 2 && s->c == 2)
        {
            // 상태 (2, 2, R)로부터 입력 (0, 1)에 따른 상태 (2, 1, L)로의 전이
            if (m == 0 && c == 1)
            {
                s->m = 2;
                s->c = 1;
                s->p = 'l';
            }
            // 상태 (2, 2, R)로부터 입력 (0, 2)에 따른 상태 (2, 0, L)로의 전이
            else if (m == 0 && c == 2)
            {

```

```

        s->m = 2;
        s->c = 0;
        s->p = 'l';
    }
    // 상태 (2, 2, R)로부터 입력 (1, 0)에 따른 상태 종결이 불가능한 경우
    else if (m == 1 && c == 0)
    {
        printf("식인종들이 선교사(들)을 잡아먹음!");
        return -1;
    }
    // 상태 (2, 2, R)로부터 입력 (1, 1)에 따른 상태 (1, 1, L)로의 전이
    else if (m == 1 && c == 1)
    {
        s->m = 1;
        s->c = 1;
        s->p = 'l';
    }
    // 상태 (2, 2, R)로부터 입력 (2, 0)에 따른 상태 (0, 2, L)로의 전이
    else if (m == 2 && c == 0)
    {
        s->m = 0;
        s->c = 2;
        s->p = 'l';
    }
}
// 현재 상태가 (2, 1, R)일 때
else if (s->m == 2 && s->c == 1)
{
    // 상태 (2, 1, R)로부터 입력 (0, 1)에 따른 상태 (2, 0, L)로의 전이
    if (m == 0 && c == 1)
    {
        s->m = 2;
        s->c = 0;
        s->p = 'l';
    }
    // 상태 (2, 1, R)로부터 입력 (0, 2)에 따른 상태 종결이 불가능한 경우
    else if (m == 0 && c == 2)
    {
        printf("식인종들이 선교사(들)을 잡아먹음!");
        return -1;
    }
    // 상태 (2, 1, R)로부터 입력 (1, 0)에 따른 상태 (1, 1, L)로의 전이
    else if (m == 1 && c == 0)
    {
        s->m = 1;
        s->c = 1;
        s->p = 'l';
    }
    // 상태 (2, 1, R)로부터 입력 (1, 1)에 따른 상태 (1, 0, L)로의 전이
    else if (m == 1 && c == 1)
    {
        s->m = 1;
        s->c = 0;
        s->p = 'l';
    }
    // 상태 (2, 1, R)로부터 입력 (2, 0)에 따른 상태 (0, 1, L)로의 전이
    else if (m == 2 && c == 0)
    {
        s->m = 0;

```

```

        s->c = 1;
        s->p = 'l';
    }
}
// 현재 상태가 (2, 0, R)일 때
else if (s->m == 2 && s->c == 0)
{
    // 상태 (2, 0, R)로부터 입력 (0, 1)에 따른 상태 종결이 불가능한 경우
    if (m == 0 && c == 1)
    {
        printf("식인종들이 선교사(들)을 잡아먹음!");
        return -1;
    }
    // 상태 (2, 0, R)로부터 입력 (0, 2)에 따른 상태 종결이 불가능한 경우
    else if (m == 0 && c == 2)
    {
        printf("식인종들이 선교사(들)을 잡아먹음!");
        return -1;
    }
    // 상태 (2, 0, R)로부터 입력 (1, 0)에 따른 상태 (1, 0, L)로의 전이
    else if (m == 1 && c == 0)
    {
        s->m = 1;
        s->c = 0;
        s->p = 'l';
    }
    // 상태 (2, 0, R)로부터 입력 (1, 1)에 따른 상태 종결이 불가능한 경우
    else if (m == 1 && c == 1)
    {
        printf("식인종들이 선교사(들)을 잡아먹음!");
        return -1;
    }
    // 상태 (2, 0, R)로부터 입력 (2, 0)에 따라 상태 (0, 0, L)로 전이하고 종결 상태에
    도달
    else if (m == 2 && c == 0)
    {
        s->m = 0;
        s->c = 0;
        s->p = 'l';
        printf("무사히 왼쪽 강가에 도달함.: \t축하합니다!!!");
        return 1;
    }
}
// 현재 상태가 (1, 1, R)일 때
else if (s->m == 1 && s->c == 1)
{
    // 상태 (1, 1, R)로부터 입력 (0, 1)에 따른 상태 (1, 0, L)로의 전이
    if (m == 0 && c == 1)
    {
        s->m = 1;
        s->c = 0;
        s->p = 'l';
    }
    // 상태 (1, 1, R)로부터 입력 (0, 2)에 따른 상태 종결이 불가능한 경우
    else if (m == 0 && c == 2)
    {
        printf("식인종들이 선교사(들)을 잡아먹음!");
        return -1;
    }
}

```

도달

```
// 상태 (1, 1, R)로부터 입력 (1, 0)에 따른 상태 (0, 1, L)로의 전이
else if (m == 1 && c == 0)
{
    s->m = 0;
    s->c = 1;
    s->p = 'l';
}
// 상태 (1, 1, R)로부터 입력 (1, 1)에 따라 상태 (0, 0, L)로 전이하고 종결 상태에
else if (m == 1 && c == 1)
{
    s->m = 0;
    s->c = 0;
    s->p = 'l';
    printf("무사히 왼쪽 강가에 도달함.: \t축하합니다!!!");
    return 1;
}
// 상태 (1, 1, R)로부터 입력 (2, 0)에 따른 상태 종결이 불가능한 경우
else if (m == 2 && c == 0)
{
    printf("식인종들이 선교사(들)을 잡아먹음!");
    return -1;
}
}
// 현재 상태가 (0, 2, R)일 때
else if (s->m == 0 && s->c == 2)
{
    // 상태 (0, 2, R)로부터 입력 (0, 1)에 따른 상태 (0, 1, L)로의 전이
    if (m == 0 && c == 1)
    {
        s->m = 0;
        s->c = 1;
        s->p = 'l';
    }
    // 상태 (0, 2, R)로부터 입력 (0, 2)에 따라 상태 (0, 0, L)로 전이하고 종결 상태에
    else if (m == 0 && c == 2)
    {
        s->m = 0;
        s->c = 0;
        s->p = 'l';
        printf("무사히 왼쪽 강가에 도달함. : \t축하합니다!!!");
        return 1;
    }
    // 상태 (0, 2, R)로부터 입력 (1, 0)에 따른 상태 종결이 불가능한 경우
    else if (m == 1 && c == 0)
    {
        printf("식인종들이 선교사(들)을 잡아먹음!");
        return -1;
    }
    // 상태 (0, 2, R)로부터 입력 (1, 1)에 따른 상태 종결이 불가능한 경우
    else if (m == 1 && c == 1)
    {
        printf("식인종들이 선교사(들)을 잡아먹음!");
        return -1;
    }
    // 상태 (0, 2, R)로부터 입력 (0, 0)에 따른 상태 종결이 불가능한 경우
    else if (m == 2 && c == 0)
    {

```

도달

```

        printf("식인종들이 선교사(들)을 잡아먹음!");
        return -1;
    }
}
// 현재 상태가 (0, 1, R)일 때
else if (s->m == 0 && s->c == 1)
{
    // 상태 (0, 1, R)로부터 입력 (0, 1)에 따라 상태 (0, 0, L)로 전이하고 종결 상태에
도달
    if (m == 0 && c == 1)
    {
        s->m = 0;
        s->c = 0;
        s->p = 'l';
        printf("무사히 왼쪽 강가에 도달함. : \t축하합니다!!!");
        return 1;
    }
    // 상태 (0, 1, R)로부터 입력 (0, 2)에 따른 상태 종결이 불가능한 경우
    else if (m == 0 && c == 2)
    {
        printf("식인종들이 선교사(들)을 잡아먹음!");
        return -1;
    }
    // 상태 (0, 1, R)로부터 입력 (1, 0)에 따른 상태 종결이 불가능한 경우
    else if (m == 1 && c == 0)
    {
        printf("식인종들이 선교사(들)을 잡아먹음!");
        return -1;
    }
    // 상태 (0, 1, R)로부터 입력 (1, 1)에 따른 상태 종결이 불가능한 경우
    else if (m == 1 && c == 1)
    {
        printf("식인종들이 선교사(들)을 잡아먹음!");
        return -1;
    }
    // 상태 (0, 1, R)로부터 입력 (2, 0)에 따른 상태 종결이 불가능한 경우
    else if (m == 2 && c == 0)
    {
        printf("식인종들이 선교사(들)을 잡아먹음!");
        return -1;
    }
}
}
else if (s->p == 'l')    // 배가 왼쪽에 있을 때
{
    // 현재 상태가 (2, 1, L)일 때
    if (s->m == 2 && s->c == 1)
    {
        // 상태 (2, 1, L)로부터 입력 (0, 1)에 따른 상태 (2, 2, R)로의 전이
        if (m == 0 && c == 1)
        {
            s->m = 2;
            s->c = 2;
            s->p = 'r';
        }
        // 상태 (2, 1, L)로부터 입력 (0, 2)에 따른 상태 종결이 불가능한 경우
        else if (m == 0 && c == 2)
        {
            printf("식인종들이 선교사(들)을 잡아먹음!");

```



```

        return -1;
    }
    // 상태 (2, 1, L)로부터 입력 (1, 0)에 따른 상태 종결이 불가능한 경우
    else if (m == 1 && c == 0)
    {
        printf("식인종들이 선교사(들)을 잡아먹음!");
        return -1;
    }
    // 상태 (2, 1, L)로부터 입력 (1, 1)에 따른 상태 종결이 불가능한 경우
    else if (m == 1 && c == 1)
    {
        printf("식인종들이 선교사(들)을 잡아먹음!");
        return -1;
    }
    // 상태 (2, 1, L)로부터 입력 (2, 0)에 따른 상태 종결이 불가능한 경우
    else if (m == 2 && c == 0)
    {
        printf("식인종들이 선교사(들)을 잡아먹음!");
        return -1;
    }
}
// 현재 상태가 (2, 0, L)일 때
else if (s->m == 2 && s->c == 0)
{
    // 상태 (2, 0, L)로부터 입력 (0, 1)에 따른 상태 (2, 1, R)로의 전이
    if (m == 0 && c == 1)
    {
        s->m = 2;
        s->c = 1;
        s->p = 'r';
    }
    // 상태 (2, 0, L)로부터 입력 (0, 2)에 따른 상태 (2, 2, R)로의 전이
    else if (m == 0 && c == 2)
    {
        s->m = 2;
        s->c = 2;
        s->p = 'r';
    }
    // 상태 (2, 0, L)로부터 입력 (1, 0)에 따른 상태 종결이 불가능한 경우
    else if (m == 1 && c == 0)
    {
        printf("식인종들이 선교사(들)을 잡아먹음!");
        return -1;
    }
    // 상태 (2, 0, L)로부터 입력 (1, 1)에 따른 상태 종결이 불가능한 경우
    else if (m == 1 && c == 1)
    {
        printf("식인종들이 선교사(들)을 잡아먹음!");
        return -1;
    }
    // 상태 (2, 0, L)로부터 입력 (2, 0)에 따른 상태 종결이 불가능한 경우
    else if (m == 2 && c == 0)
    {
        printf("식인종들이 선교사(들)을 잡아먹음!");
        return -1;
    }
}
// 현재 상태가 (1, 1, L)일 때
else if (s->m == 1 && s->c == 1)

```

```

{
    // 상태 (1, 1, L)로부터 입력 (0, 1)에 따른 상태 종결이 불가능한 경우
    if (m == 0 && c == 1)
    {
        printf("식인종들이 선교사(들)을 잡아먹음!");
        return -1;
    }
    // 상태 (1, 1, L)로부터 입력 (0, 2)에 따른 상태 종결이 불가능한 경우
    else if (m == 0 && c == 2)
    {
        printf("식인종들이 선교사(들)을 잡아먹음!");
        return -1;
    }
    // 상태 (1, 1, L)로부터 입력 (1, 0)에 따른 상태 (2, 1, R)로의 전이
    else if (m == 1 && c == 0)
    {
        s->m = 2;
        s->c = 1;
        s->p = 'r';
    }
    // 상태 (1, 1, L)로부터 입력 (1, 1)에 따른 상태 (2, 2, R)로의 전이
    else if (m == 1 && c == 1)
    {
        s->m = 2;
        s->c = 2;
        s->p = 'r';
    }
    // 상태 (1, 1, L)로부터 입력 (2, 0)에 따른 상태 종결이 불가능한 경우
    else if (m == 2 && c == 0)
    {
        printf("식인종들이 선교사(들)을 잡아먹음!");
        return -1;
    }
}
// 현재 상태가 (0, 2, L)일 때
else if (s->m == 0 && s->c == 2)
{
    // 상태 (0, 2, L)로부터 입력 (0, 1)에 따른 상태 종결이 불가능한 경우
    if (m == 0 && c == 1)
    {
        printf("식인종들이 선교사(들)을 잡아먹음!");
        return -1;
    }
    // 상태 (0, 2, L)로부터 입력 (0, 2)에 따른 상태 (2, 2, R)로의 전이
    else if (m == 0 && c == 2)
    {
        s->m = 2;
        s->c = 2;
        s->p = 'r';
    }
    // 상태 (0, 2, L)로부터 입력 (1, 0)에 따른 상태 종결이 불가능한 경우
    else if (m == 1 && c == 0)
    {
        printf("식인종들이 선교사(들)을 잡아먹음!");
        return -1;
    }
    // 상태 (0, 2, L)로부터 입력 (1, 1)에 따른 상태 종결이 불가능한 경우
    else if (m == 1 && c == 1)
    {

```

```

        printf("식인종들이 선교사(들)을 잡아먹음!");
        return -1;
    }
    // 상태 (0, 2, L)로부터 입력 (2, 0)에 따른 상태 (2, 2, R)로의 전이
    else if (m == 2 && c == 0)
    {
        s->m = 2;
        s->c = 2;
        s->p = 'r';
    }
}
// 현재 상태가 (0, 1, L)일 때
else if (s->m == 0 && s->c == 1)
{
    // 상태 (0, 1, L)로부터 입력 (0, 1)에 따른 상태 (0, 2, R)로의 전이
    if (m == 0 && c == 1)
    {
        s->m = 0;
        s->c = 2;
        s->p = 'r';
    }
    // 상태 (0, 1, L)로부터 입력 (0, 2)에 따른 상태 종결이 불가능한 경우
    else if (m == 0 && c == 2)
    {
        printf("식인종들이 선교사(들)을 잡아먹음!");
        return -1;
    }
    // 상태 (0, 1, L)로부터 입력 (1, 0)에 따른 상태 (0, 2, R)로의 전이
    else if (m == 1 && c == 0)
    {
        s->m = 1;
        s->c = 1;
        s->p = 'r';
    }
    // 상태 (0, 1, L)로부터 입력 (1, 1)에 따른 상태 종결이 불가능한 경우
    else if (m == 1 && c == 1)
    {
        printf("식인종들이 선교사(들)을 잡아먹음!");
        return -1;
    }
    // 상태 (0, 1, L)로부터 입력 (2, 0)에 따른 상태 (1, 1, R)로의 전이

    else if (m == 2 && c == 0)
    {
        s->m = 1;
        s->c = 1;
        s->p = 'r';
    }
}
// 현재 상태가 (0, 0, L)일 때
else if (s->m == 0 && s->c == 0)
{
    // 상태 (0, 0, L)로부터 입력 (0, 1)에 따른 상태 (0, 1, R)로의 전이
    if (m == 0 && c == 1)
    {
        s->m = 0;
        s->c = 1;
        s->p = 'r';
    }
}

```

```

// 상태 (0, 0, L)로부터 입력 (0, 2)에 따른 상태 (0, 2, R)로의 전이
else if (m == 0 && c == 2)
{
    s->m = 0;
    s->c = 2;
    s->p = 'r';
}
// 상태 (0, 0, L)로부터 입력 (1, 0)에 따른 상태 (1, 0, R)로의 전이
else if (m == 1 && c == 0)
{
    s->m = 1;
    s->c = 0;
    s->p = 'r';
}
// 상태 (0, 0, L)로부터 입력 (1, 1)에 따른 상태 (1, 1, R)로의 전이
else if (m == 1 && c == 1)
{
    s->m = 1;
    s->c = 1;
    s->p = 'r';
}
// 상태 (0, 0, L)로부터 입력 (2, 0)에 따른 상태 (2, 0, R)로의 전이
else if (m == 2 && c == 0)
{
    s->m = 2;
    s->c = 0;
    s->p = 'r';
}
}
}
return 0;
}

int main()
{
    int res = 0;
    int input_m, input_c;    // 선교사와 식인종의 총 인원수를 위한 변수(사용자 입력)
    state s;                // 구조체로 선언된 상태 객체를 선언
    state_init(&s);         // 시작 상태를 초기화
    // 인식 상태 (0, 0, L)에 도달하거나 종결이 불가능한 상태에 도달할 때까지 반복
    while (res == 0)
    {
        // 사용자로부터 알파벳을 입력받음
        printf("현재 상태 : (%d,%d,%c)\n", s.m, s.c, s.p);
        printf("승선 인원수를 입력하라. : ");
        scanf("%d %d", &input_m, &input_c);
        printf("입력 : (%d,%d)\n", input_m, input_c);
        // 입력된 알파벳을 바탕으로 전이하고 결과를 반환
        res = state_moving(&s, input_m, input_c);
    }
    // 마지막 상태를 출력
    printf("\n\n상태 : (%d,%d,%c)\n", s.m, s.c, s.p);
    system("PAUSE");
}

```