

## 1. 문제 정의 기술

## 문제

## 1. 사용자로부터 좌표 및 문자 입력받아 화면에 출력

- 좌표는 행과 열 번호로 입력 받으며, 사전에 아래와 같은 가이드를 미리 출력해 놓는다. (20행 70열)

```
**123456789012345678901234567890123456789012345678901234567890
```

```
*****
```

```
01_____
```

```
02_____
```

```
03_____
```

```
04_____
```

```
...//. 생략
```

```
18_____
```

```
19_____
```

```
20_____
```

- 사용자로부터 행과 열 번호 및 문자하나를 입력받는다.
- 해당위치에 그 문자를 출력한다.
- 화면의 모든 문자들을 지우는 것은 시스템 명령어 실행 함수 'int system(const char \*command)'를 이용하여 'cls' 명령어를 실행하여 화면을 초기화 할 수 있다.
- 2차원 배열을 이용하여 화면 저장.

## 2. 화면 구성 및 입력

아스키 문자 및 2바이트 문자들을 이용하여 게임 화면을 구성한다.

- 사용자가 선택할 수 있는 메뉴를 구성한다.
- 1. 그리기 2. 지우기 3. 저장하기 4. 불러오기 5. 새 화면 6. 종료
- 필요에 따라 그리기 메뉴의 세부 메뉴를 구성해도 된다. 예를 들어 1번행부터 5번행까지 20열에 '\*' 출력을 위한 여러 행 출력(세로줄) 메뉴 또는 15열부터 27열까지 7행에 '\*'를 출력하는 메뉴 (가로줄). 그 외 대각줄, 사각형 등의 메뉴 가능.
- 저장하기는 파일이름을 입력받아 텍스트파일로 저장을 한다. 다만 저장할 때 동일한 파일이 존재한다면 덮어쓸지 말지를 선택할 수 있어야 한다. (파일이 존재하는지 확인할 때 사용하는 함수는 io.h 에 있는 access(경로명, 0) 함수를 활용. 파일이 존재하면 0반환 ) (가산점)
- 불러오기는 이미 저장된 파일을 불러와서 화면에 그려준다. 파일 불러오기가 실패할 때 등을 위한 오류 처리 필요.(가산점)

문제 정의: 사용자로부터 좌표와 문자를 입력받아 해당 위치에 문자를 출력할 수 있게 하고, 화면을 초기화하거나 저장된 파일을 불러올 수 있어야 하고, 파일 저장시 동일한 이름의 파일이 있을 경우 덮어쓰기 여부를 물어봐야 하고, 저장 및 불러오기 과정에서 발생할 수 있는 오류를 처리하여 사용자에게 알려야 한다.

## 2. 문제 해결 방법

### -자료 구조 정의

char board[rows][cols] : 20행 70열의 2차원 배열이며, char의 자료형은 문자형이고, 표현범위는 -128~127까지이다. 텍스트 기반 그림판의 화면 상태를 저장하는 자료 구조이다. 각 셀은 하나의 문자를 저장하며, 초기 상태에서는 '\_'로 설정되어 빈 화면을 나타낸다. 사용자는 특정 좌표에 문자를 추가하거나 삭제할 수 있으며, 배열의 내용은 파일로 저장되거나 불러올 수 있다. board 배열은 화면 관리, 데이터 처리, 입출력 작업에서 역할을 한다.

char filename[100]: 사용자가 저장 또는 불러오기 작업 시 파일 이름을 저장하는 배열이며, char의 자료형은 문자형이고, 표현범위는 -128~127까지이다. 또한 최대 99개의 문자를 저장할 수 있으며, 파일 입출력 작업에서 파일명을 저장하고 fopen 함수와 함께 사용된다.

char overwrite; : 파일이 이미 존재할 때 덮어쓰기 여부를 저장하는 변수이며, char의 자료형은 문자형이고, 표현범위는 -128~127까지이다. 이 프로그램에서는 saveboard 함수에서 파일 덮어쓰기 동작을 제어하는데 사용이 된다.

int row, col; : 정수형 자료형이며, 이 프로그램에서는 사용자가 입력한 행과 열의 위치를 저장하는 데 사용된다. 이 프로그램에서는 drawcharacter함수, erasecharacter 등에서 좌표 기반 작업을 수행할 때 사용된다.

char ch: 문자형 자료형이며, 사용자가 입력한 문자를 저장하는 변수이다. 이 프로그램에서는 사용자가 입력한 문자를 저장할 때 사용된다.

int choice: 정수형 자료형이며, 메뉴 선택에서 사용자가 입력한 선택 값을 저장하는 변수이다. 이 프로그램에서는 displaymenu 함수에서 사용자 선택에 따라 작업을 수행하는 데 사용이 된다.

## -알고리즘 설명- pseudo-code or flow chart

~~~~~  
프로그램 시작

상수 rows 20으로 정의  
상수 cols 70으로 정의

2차원 문자 배열 board[rows][cols] 정의

함수 resetboard 정의 :  
모든 위치의 값을 '\_'로 설정하여 board 초기화

함수 displayboard 정의 :  
첫 번째 줄에 열 번호 출력  
화면 테두리를 출력  
각 행 번호와 함께 해당 행의 문자 출력

함수 drawcharacter 정의(매개변수 : row, col, ch) :  
row와 col이 유효한 범위이면  
board[row - 1][col - 1] 위치에 ch를 설정  
유효하지 않으면 "틀린 좌표" 출력

함수 erasecharacter 정의(매개변수 : row, col) :  
row와 col이 유효한 범위이면  
board[row - 1][col - 1] 위치에 '\_'로 설정  
유효하지 않으면 "틀린 좌표" 출력

함수 clearscreen 정의 :  
화면을 지우기 위해 system("cls") 호출  
resetboard 호출하여 board 초기화

함수 saveboard 정의 :  
파일 이름 입력받음  
파일이 존재하면 덮어쓰기 여부를 묻고, 선택에 따라 저장  
board의 내용을 파일에 쓰기  
저장 성공 여부 출력

함수 loadboard 정의 :  
파일 이름 입력받음  
파일을 열어 board의 데이터를 읽어옴  
파일 존재하지 않거나 오류 발생 시 "불러오기 실패" 출력  
성공하면 "불러오기 완료" 출력

함수 displaymenu 정의 :  
사용자에게 메뉴를 보여주고 choice 입력받음  
만약 choice가 1인 경우(그리기 선택) :  
row, col, ch 입력받음  
drawcharacter(row, col, ch) 호출  
displayboard 호출하여 업데이트된 화면 출력  
만약 choice가 2인 경우(지우기 선택) :  
row, col 입력받음  
erasecharacter(row, col) 호출  
displayboard 호출하여 업데이트된 화면 출력  
만약 choice가 3인 경우(저장하기 선택) :  
saveboard 호출  
만약 choice가 4인 경우(불러오기 선택) :  
loadboard 호출  
displayboard 호출하여 업데이트된 화면 출력  
만약 choice가 5인 경우(새 화면 선택) :  
clearscreen 호출  
만약 choice가 6인 경우 :  
"프로그램 종료" 출력하고 프로그램 종료  
그 외의 입력인 경우 :  
"유효하지 않은 입력" 출력

함수 main 실행 :  
resetboard 호출하여 board 초기화  
displayboard 호출하여 초기 화면 출력  
displaymenu 호출하여 메뉴 실행

프로그램 종료

## -정의한 함수에 대한 api 기술

void resetboard() : void는 원래 함수가 아니라 자료형의 종류 중 하나이지만, 사용자가 함수로 정의하면 함수로 사용이 가능하다. 사용자가 직접 개발한 함수를 사용하기 위해서는 함수선언, 함수호출, 함수정의가 필요하다. 이 함수는 프로그램 시작시와 사용자가 새 화면을 선택할 때 호출이 된다. 이중 반복문을 사용해 각 행과 열의 값을 탐색하며 배열을 초기화한다. 사용자가 기존 화면을 완전히 지우고 새 작업을 시작할 수 있도록 지원한다.

void displayboard() : void는 원래 함수가 아니라 자료형의 종류 중 하나이지만, 사용자가 함수로 정의하면 함수로 사용이 가능하다. 사용자가 직접 개발한 함수를 사용하기 위해서는 함수선언, 함수호출, 함수정의가 필요하다. 이 함수는 board 배열의 현재 상태를 출력하며, 출력은 열 번호와 행 번호를 포함하며, 각

셀의 값을 보여주는 방식으로 구성된다. 이 함수는 화면의 현재 상태를 사용자에게 명확히 전달하기 위해 호출이 된다.

`void drawcharacter` : `void`는 원래 함수가 아니라 자료형의 종류 중 하나이지만, 사용자가 함수로 정의하면 함수로 사용이 가능하다. 사용자가 직접 개발한 함수를 사용하기 위해서는 함수선언, 함수호출, 함수정의가 필요하다. 이 함수는 사용자가 입력한 좌표와 문자를 기반으로 해당 위치에 문자를 저장하며, 입력된 좌표가 범위 안에 있으면 `board[row-1][col-1]`에 문자를 저장하며, 그렇지 않은 경우 오류메시지를 출력하며, 화면에 문자를 추가하거나 변경할 때 호출이 된다.

`void erasecharacter` : `void`는 원래 함수가 아니라 자료형의 종류 중 하나이지만, 사용자가 함수로 정의하면 함수로 사용이 가능하다. 사용자가 직접 개발한 함수를 사용하기 위해서는 함수선언, 함수호출, 함수정의가 필요하다. 이 함수는 특정 좌표의 문자를 삭제하여 해당 위치를 빈 공간('\_')으로 초기화 하고, 좌표가 유효한 범위 내에 있어야 하고, 그렇지 않을 경우 오류메시지를 출력하며, 사용자가 특정 위치의 문자를 제거하거나 화면을 수정할 때 사용이 된다.

`void clearscreen` : `void`는 원래 함수가 아니라 자료형의 종류 중 하나이지만, 사용자가 함수로 정의하면 함수로 사용이 가능하다. 사용자가 직접 개발한 함수를 사용하기 위해서는 함수선언, 함수호출, 함수정의가 필요하다. 이 함수는 화면을 초기화하고 깨끗하게 지우는 역할을 하며, `restboard` 함수를 호출하여 배열을 초기화한 후 `system("cls")`를 실행하여 화면을 지운다. 또한 새로운 작업을 시작하기 위해 사용자가 요청할 때 호출이 되며, 초기화 된 상태를 출력하기 위해 `displayboard` 함수를 사용한다.

`void saveboard`: `void`는 원래 함수가 아니라 자료형의 종류 중 하나이지만, 사용자가 함수로 정의하면 함수로 사용이 가능하다. 사용자가 직접 개발한 함수를 사용하기 위해서는 함수선언, 함수호출, 함수정의가 필요하다. 이 함수는 현재 `board` 배열의 상태를 파일로 저장하며, 저장할 파일 이름은 사용자로부터 입력을 받으며, 파일이 이미 존재하면 `_access` 함수를 통해 확인하고, 덮어쓰기 여부를 묻는다. 그리고나면 사용자의 선택에 따라 데이터를 저장하며, 저장 성공 또는 실패 메시지를 출력한다.

`void loadboard`: `void`는 원래 함수가 아니라 자료형의 종류 중 하나이지만, 사용자가 함수로 정의하면 함수로 사용이 가능하다. 사용자가 직접 개발한 함수를 사용하기 위해서는 함수선언, 함수호출, 함수정의가 필요하다. 이 함수는 사용자로부터 입력받은 파일에서 데이터를 읽어와 `board` 배열에 복원한다. 파일이 존재하지 않거나 읽기 실패 시 오류 메시지를 출력한다. 데이터는 각 행 단위로 읽어오며, 파일이 유효한 형식의 경우 화면에 적용이 된다. 만약에 성공한다면 "불러오기 완료"라는 메시지를 출력한다.

`void displaymenu`: `void`는 원래 함수가 아니라 자료형의 종류 중 하나이지만, 사용자가 함수로 정의하면 함수로 사용이 가능하다. 사용자가 직접 개발한 함수를 사용하기 위해서는 함수선언, 함수호출, 함수정의가 필요하다. 이 함수는 사용자에게 메뉴를 출력하고, 선택된 작업을 수행한다. 사용자가 메뉴에서 입력한 값에 따라 적절한 함수를 호출하여, 종료 요청시 프로그램을 종료한다. 만약에 메뉴번호가 유효하지 않으면, 오류메시지를 출력한다.

`int main()` : 프로그램의 시작점으로, 먼저 `restboard` 함수를 호출하여 `board` 배열을 초기화하고, `displayboard`를 통해 초기 상태의 화면을 출력한다. 그 다음에 `displaymenu()`를 호출하여 사용자 입력을

받아 메뉴 선택에 따라 작업을 수행한다. 프로그램 종료 시에는 return 0을 반환하여 종료되었음을 알린다. 다른 모든 함수들을 조합하여 프로그램의 전반적인 실행을 관리한다.

#### -호출한 함수에 대한 간략한 설명

system("cls"): 화면을 깨끗하게 지우는 명령어 중 하나이며, clearscreen 함수에서 호출되어 화면을 초기화할 때 사용된다. 이 프로그램에서는 이전 출력 내용을 지우고 새 화면을 출력할 준비를 할 때 사용한다.

\_access: 이 함수는 파일이 존재하는지 확인하는 함수이다. 이 프로그램에서는 파일이 존재하면 0을 반환하고, 존재하지 않으면 -1을 반환한다. saveboard 함수에서 파일이 이미 있는지 확인해 덮어쓰기 여부를 사용자에게 묻기 전에 호출이 된다. 특히 저장 작업에서 중요한 사전 점검 역할을 하며, 파일 존재 여부를 확인하는 데 매우 유용하다.

fopen: 파일을 열거나 새로 생성하는 데 사용되는 함수이며, 쓰기 모드("w")와 읽기 모드("r")로 열 수 있었다. saveboard에서는 파일에 데이터를 저장하기 위해 사용되며, loadboard에서는 파일 데이터를 읽어오기 위해 호출된다. 파일 열기에 실패하면 NULL을 반환하여 오류 처리가 가능하다.

fwrite: 메모리에 있는 데이터를 파일에 쓰는 함수이며, saveboard에서 board배열의 데이터를 파일에 저장할 때 사용이 된다. 각 행의 데이터를 쓰고 줄바꿈 문자를 추가하여 저장하며, 저장 작업이 성공했는지 확인할 수 있는 반환값도 제공한다.

fread: 파일에서 데이터를 읽어와 메모리에 저장하는 함수이며, loadboard에서 파일 데이터를 board배열에 복원할 때 사용이 된다. 저장된 파일에서 각 행의 데이터를 읽어와 화면을 복원할 때 사용되며, 읽기 실패 시 반환값으로 오류를 확인할 수도 있다.

fgetc: 파일에서 한 문자를 읽어오는 함수이며, loadboard에서 각 줄의 데이터를 읽은 후 줄바꿈 문자(wn)을 0처리하는데 사용된다. 파일 포인터를 한 문자씩 이동시키며 파일 끝에 도달하면 -1을 반환한다.

scanf(): 대표적인 입력함수이며, 메뉴 선택, 파일명 입력, 좌표와 문자 입력 등에서 사용된다. 사용자가 원하는 작업을 프로그램에 전달할 수 있는 기본적인 입력 도구이며, 입력값의 형식을 지정하여 다양한 데이터를 받을 수 있다.

printf(): 대표적인 출력함수이며, 이 프로그램에서는 프로그램 내에서 메뉴 출력, 상태 알림, 오류 메시지 등을 출력할 때 사용이 된다.

### 3. 소스 코드(주석 포함)

```
#define _CRT_SECURE_NO_WARNINGS
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <iio.h> // _access 함수 사용
#define rows 20
#define cols 70

char board[rows][cols];

// 화면 초기화 함수
void resetboard() {
    for (int i = 0; i < rows; i++) {
        for (int j = 0; j < cols; j++) {
            board[i][j] = '_'; // 초기 화면에 '_' 문자를 사용하여 비워두기
        }
    }
}

// 화면 출력 함수
void displayboard() {
    printf("**123456789012345678901234567890123456789012345678901234567890\n");
    printf("*****\n");
    for (int i = 0; i < rows; i++) {
        printf("%02d", i + 1);
        for (int j = 0; j < cols; j++) {
            printf("%c", board[i][j]);
        }
        printf("\n");
    }
}

// 좌표에 문자 그리기 함수
void drawcharacter(int row, int col, char ch) {
    if (row >= 1 && row <= rows && col >= 1 && col <= cols) {
        board[row - 1][col - 1] = ch;
    }
    else {
        printf("틀린 좌표입니다.\n");
    }
}

// 좌표 지우기 함수
void erasecharacter(int row, int col) {
    if (row >= 1 && row <= rows && col >= 1 && col <= cols) {
        board[row - 1][col - 1] = '_';
    }
    else {
        printf("틀린 좌표입니다.\n");
    }
}

// 화면 초기화
void clearscreen() {
    resetboard();
    system("cls");
    displayboard();
}
```

```

// 화면 저장 함수
void saveboard() {
    char filename[100];
    printf("저장할 파일 이름 입력: ");
    scanf("%s", filename);

    if (_access(filename, 0) == 0) {
        printf("파일이 이미 존재합니다. 덮어쓸까요? (y/n): ");
        char overwrite;
        scanf("%c", &overwrite);
        if (overwrite != 'y' && overwrite != 'Y') {
            printf("저장 취소\n");
            return;
        }
        else {
            printf("새 파일을 생성하여 저장합니다.\n");
        }
    }

    FILE* file = fopen(filename, "w");
    if (!file) {
        printf("파일 저장 실패\n");
        return;
    }

    for (int i = 0; i < rows; i++) {
        fwrite(board[i], sizeof(char), cols, file);
        fputc('\n', file);
    }
    fclose(file);
    printf("저장 완료\n");
}

```

```

// 화면 불러오기 함수
void loadboard() {
    char filename[100];
    printf("불러올 파일 이름 입력: ");
    scanf("%s", filename);

    FILE* file = fopen(filename, "r");
    if (!file) {
        printf("파일 불러오기 실패\n");
        return;
    }

    for (int i = 0; i < rows; i++) {
        if (fread(board[i], sizeof(char), cols, file) < cols) {
            printf("파일 데이터가 올바르지 않습니다.\n");
            fclose(file);
            return;
        }
        fgetc(file); // 개행 문자 읽기
    }
    fclose(file);
    printf("불러오기 완료\n");
}

```

```

void displaymenu() {
    int choice;
    while (1) {
        printf("\n1. 그리기\n2. 지우기\n3. 저장하기\n4. 불러오기\n5. 새 화면\n6. 종료\n");
        printf("선택: ");
        if (scanf("%d", &choice) != 1) {
            printf("잘못된 입력입니다. 프로그램 종료\n");
            exit(1);
        }

        if (choice == 1) { // 그리기
            int row, col;
            char ch;
            printf("행, 열 번호 및 문자를 입력 (예: 5 10 *): ");
            if (scanf("%d %d %c", &row, &col, &ch) != 3) {
                printf("잘못된 입력입니다. 프로그램 종료\n");
                exit(1);
            }
            drawcharacter(row, col, ch);
            displayboard();
        }
        else if (choice == 2) { // 지우기
            int row, col;
            printf("지울 행과 열 번호 입력 (예: 5 10): ");
            if (scanf("%d %d", &row, &col) != 2) {
                printf("잘못된 입력입니다. 프로그램 종료\n");
                exit(1);
            }
            erasecharacter(row, col);
            displayboard();
        }
        else if (choice == 3) { // 저장하기
            saveboard();
        }
        else if (choice == 4) { // 불러오기
            loadboard();
            displayboard();
        }
        else if (choice == 5) { // 새 화면
            clearscreen();
        }
        else if (choice == 6) { // 종료
            printf("프로그램 종료\n");
            break;
        }
        else {
            printf("유효하지 않은 선택입니다.\n");
        }
    }
}

```

```

int main() {
    resetboard();
    displayboard();
    displaymenu();
    return 0;
}

```



#### 4. 실행 결과 캡처

그리기 실행할 때

```
**123456789012345678901234567890123456789012345678901234567890
*****
01_-----
02_-----
03_-----
04_-----
05_-----
06_-----
07_-----
08_-----
09_-----
10_-----
11_-----
12_-----
13_-----
14_-----
15_-----
16_-----
17_-----
18_-----
19_-----
20_-----

1. 그리기
2. 지우기
3. 저장하기
4. 불러오기
5. 새 화면
6. 종료
선택: 1
행, 열 번호 및 문자를 입력 (예: 5 10 *): 5 10 *
```

```
행, 열 번호 및 문자를 입력 (예: 5 10 *): 5 10 *
**123456789012345678901234567890123456789012345678901234567890
*****
01_-----
02_-----
03_-----
04_-----
05_-----*-----
06_-----
07_-----
08_-----
09_-----
10_-----
11_-----
12_-----
13_-----
14_-----
15_-----
16_-----
17_-----
18_-----
19_-----
20_-----

1. 그리기
2. 지우기
3. 저장하기
4. 불러오기
5. 새 화면
6. 종료
선택: 1
행, 열 번호 및 문자를 입력 (예: 5 10 *): 6 10 +
```

```
**123456789012345678901234567890123456789012345678901234567890
*****
01_____
02_____
03_____
04_____
05_____ *
06_____ +
07_____
08_____
09_____
10_____
11_____
12_____
13_____
14_____
15_____
16_____
17_____
18_____
19_____
20_____

1. 그리기
2. 지우기
3. 저장하기
4. 불러오기
5. 새 화면
6. 종료
선택: 1
행, 열 번호 및 문자를 입력 (예: 5 10 *): 10 20 /
```

## 지우기 진행할 때

```
**123456789012345678901234567890123456789012345678901234567890
*****
01_____
02_____
03_____
04_____
05_____ *
06_____ +
07_____
08_____
09_____
10_____ /
11_____
12_____
13_____
14_____
15_____
16_____
17_____
18_____
19_____
20_____

1. 그리기
2. 지우기
3. 저장하기
4. 불러오기
5. 새 화면
6. 종료
선택: 2
지울 행과 열 번호 입력 (예: 5 10): 5 10
```

## 파일 저장하기 할 때

```
1. 그리기
2. 지우기
3. 저장하기
4. 불러오기
5. 새 화면
6. 종료
선택: 3
저장할 파일 이름 입력: file3
저장 완료

**123456789012345678901234567890123456789012345678901234567890
*****
01_-----
02_-----
03_-----
04_-----
05_-----
06_-----+-----
07_-----
08_-----
09_-----
10_-----/-----
11_-----
12_-----
13_-----
14_-----
15_-----
16_-----
17_-----
18_-----
19_-----
20_-----

1. 그리기
2. 지우기
3. 저장하기
4. 불러오기
5. 새 화면
6. 종료
선택: 3
저장할 파일 이름 입력: file3
저장 완료
```

## 파일 불러올 때

```
1. 그리기
2. 지우기
3. 저장하기
4. 불러오기
5. 새 화면
6. 종료
선택: 4
불러올 파일 이름 입력: file3
불러오기 완료

**123456789012345678901234567890123456789012345678901234567890
*****
01_-----
02_-----
03_-----
04_-----
05_-----
06_-----+-----
07_-----
08_-----
09_-----
10_-----/-----
11_-----
12_-----
13_-----
14_-----
15_-----
16_-----
17_-----
18_-----
19_-----
20_-----
```

같은 이름의 파일 덮어쓰기 할 때

```
1. 그리기
2. 지우기
3. 저장하기
4. 불러오기
5. 새 화면
6. 종료
선택: 3
저장할 파일 이름 입력: file3
파일이 이미 존재합니다. 덮어쓰까요? (y/n): n
저장 취소
```

```
1. 그리기
2. 지우기
3. 저장하기
4. 불러오기
5. 새 화면
6. 종료
선택: 3
저장할 파일 이름 입력: file3
파일이 이미 존재합니다. 덮어쓰까요? (y/n): y
새 파일을 생성하여 저장합니다.
저장 완료
```

파일 불러오기 실패할 때

```
**123456789012345678901234567890123456789012345678901234567890
*****
01_-----
02_-----
03_-----
04_-----
05_-----
06_-----
07_-----
08_-----
09_-----
10_-----
11_-----
12_-----
13_-----
14_-----
15_-----
16_-----
17_-----
18_-----
19_-----
20_-----

1. 그리기
2. 지우기
3. 저장하기
4. 불러오기
5. 새 화면
6. 종료
선택: 4
불러올 파일 이름 입력: file555
파일 불러오기 실패
```

## 새 화면 명령어 선택시

```
**123456789012345678901234567890123456789012345678901234567890
*****
01_-----
02_-----
03_-----
04_-----
05_-----
06_-----
07_-----
08_-----
09_-----
10_-----
11_-----
12_-----
13_-----
14_-----
15_-----
16_-----
17_-----
18_-----
19_-----
20_-----

1. 그리기
2. 지우기
3. 저장하기
4. 불러오기
5. 새 화면
6. 종료
선택: |
```

## 제대로 된 입력을 하지 않았을 때

```
**123456789012345678901234567890123456789012345678901234567890
*****
01_-----
02_-----
03_-----
04_-----
05_-----
06_-----
07_-----
08_-----
09_-----
10_-----
11_-----
12_-----
13_-----
14_-----
15_-----
16_-----
17_-----
18_-----
19_-----
20_-----

1. 그리기
2. 지우기
3. 저장하기
4. 불러오기
5. 새 화면
6. 종료
선택: 34534
유효하지 않은 선택입니다.

1. 그리기
2. 지우기
3. 저장하기
4. 불러오기
5. 새 화면
6. 종료
선택: dcs
잘못된 입력입니다. 프로그램 종료
```

## 프로그램 종료 시

```
**123456789012345678901234567890123456789012345678901234567890
*****
01_
02_
03_
04_
05_
06_
07_
08_
09_
10_
11_
12_
13_
14_
15_
16_
17_
18_
19_
20_

1. 그리기
2. 지우기
3. 저장하기
4. 불러오기
5. 새 화면
6. 종료
선택: 6
프로그램 종료
```

## 파일 불러오기 실패할 때

```
**123456789012345678901234567890123456789012345678901234567890
*****
01_
02_
03_
04_
05_
06_
07_
08_
09_
10_
11_
12_
13_
14_
15_
16_
17_
18_
19_
20_

1. 그리기
2. 지우기
3. 저장하기
4. 불러오기
5. 새 화면
6. 종료
선택: 4
불러올 파일 이름 입력: file555
파일 불러오기 실패
```