**소스 파일**

**Project 1\_Slide Game 12184407 이상운**

**#include <stdio.h>**

**#include <stdlib.h>**

**#include <conio.h>**

**#include <time.h>**

**#include <Windows.h>**

**#define MINP 3**

**#define MAXP 7**

**#define OFFSET 5**

**#define KEY\_ESC 0x1B**

**#define KEY\_8 '8'**

**#define KEY\_2 '2'**

**#define KEY\_4 '4'**

**#define KEY\_6 '6'**

**#define KEY\_W 'w'**

**#define KEY\_X 'x'**

**#define KEY\_A 'a'**

**#define KEY\_D 'd'**

**int rows, cols; // board size 퍼즐 크기**

**int piece[MAXP][MAXP]; // pieces 퍼즐 조각**

**int blank\_i, blank\_j; // empty piece 빈 조각의 위치**

**void init(); // initialize**

**void move(int key); // slide a piece**

**int judge(); // judge the end of the game**

**void draw\_piece(int i, int j); // draw a piece**

**void gotoxy(int x, int y); // move cursor**

**int main(void)**

**{**

**printf("Enter a size number (%d~%d) : ", MINP, MAXP);**

**// 사용자로부터 크기 입력 받는다.**

**scanf("%d", &rows);**

**if (rows < MINP || rows > MAXP) // 범위 (3~7)을 벗어나면 최소 크기 3으로 고정**

**rows = MINP;**

**cols = rows; // 가로줄과 세로줄의 카드 개수는 같다.**

**printf("Press ESC to exit.\n");**

**printf("Left(4, A), Right(6, D), Down(2, X), Up(8, W)\n");**

**init(); // 초기 함수 시작**

**do // 무조건 한번은 움직인다.**

**{**

**move(getch()); // 사용자로부터 키를 입력 받아 move함수로 이동**

**} while (judge()); // judge 함수가 0을 return할 때까지 무한 반복**

**return 0; // 반복 문 종료 시 메인 함수 종료**

**}**

**int judge()**

**{**

**int i, j; // i: 가로줄, j: 세로줄**

**for (i = 0; i < rows; i++)**

**{**

**for (j = 0; j < cols; j++)**

**{**

**if (piece[i][j] != (j + 1) + i\*rows)**

**// piece[0][0]부터 piece[cols-1][rows-1]까지 진행하면서 한조각이라도 틀릴 시 반복 문(do~while) 계속 진행**

**return 1;**

**}**

**} // 모든 카드를 맞출 경우 반복 문 탈출**

**return 0; // 반복 문 종료**

**}**

**void move(int key)**

**{**

**int i, j; // 빈칸의 위치 찾기**

**switch (key)**

**{**

**case KEY\_4: // left**

**case KEY\_A:**

**if (blank\_i < rows - 1) // 카드 범위 밖에 나가지 않게 범위를 지정**

**{**

**piece[blank\_j][blank\_i] = piece[blank\_j][blank\_i + 1];**

**// 오른쪽에 있는 카드를 왼쪽(빈칸의 위치)으로 이동**

**blank\_i++; // 빈칸의 가로줄 1증가**

**}**

**break;**

**case KEY\_6: // right**

**case KEY\_D:**

**if (blank\_i > 0) // 카드 범위 밖에 나가지 않게 범위를 지정**

**{**

**piece[blank\_j][blank\_i] = piece[blank\_j][blank\_i - 1];**

**// 왼쪽에 있는 카드를 오른쪽(빈칸의 위치)으로 이동**

**blank\_i--; // 빈칸의 가로줄 1감소**

**}**

**break;**

**case KEY\_2: // down**

**case KEY\_X:**

**if (blank\_j > 0) // 카드 범위 밖에 나가지 않게 범위를 지정**

**{**

**piece[blank\_j][blank\_i] = piece[blank\_j - 1][blank\_i];**

**// 위쪽에 있는 카드를 아래(빈칸의 위치)로 이동**

**blank\_j--; // 빈칸의 세로줄 1감소**

**}**

**break;**

**case KEY\_8: // up**

**case KEY\_W:**

**if (blank\_j < rows - 1) // 카드 범위 밖에 나가지 않게 범위를 지정**

**{**

**piece[blank\_j][blank\_i] = piece[blank\_j + 1][blank\_i];**

**// 아래쪽에 있는 카드를 위(빈칸의 위치)로 이동**

**blank\_j++; // 빈칸의 세로줄 1증가**

**}**

**break;**

**case KEY\_ESC: // ECS**

**gotoxy(0, rows \* 3 + 1 + OFFSET);**

**printf("Exit!\n\n");**

**exit(0); // 프로그램 종료의 라이브러리 함수.**

**break;**

**}**

**piece[blank\_j][blank\_i] = rows\*cols;**

**/\* 빈칸을 rows X cols 값으로 지정, Judge 함수의 편의를 위해 기존**

**0~rowsXcols-1의 범위를 1~rowsXcols로 지정 \*/**

**draw\_piece(blank\_i, blank\_j);**

**// 빈칸의 위치에 퍼즐 조각을 없앰 & 빈칸을 제외한 숫자에 퍼즐 그림 printf**

**for (i = 0; i < rows; i++)**

**{**

**for (j = 0; j < cols; j++)**

**{**

**gotoxy(5 + j \* 6, 5 + i \* 3);**

**if (piece[i][j] == rows\*cols)**

**// 반복 문으로 빈칸의 숫자(rowsXcols)를 배열 piece[rows][cols]에서 위치 찾기**

**printf(" ");**

**else**

**printf("%d ", piece[i][j]);**

**// 빈칸이 아닌 숫자는 원래 숫자 출력**

**}**

**printf("\n");**

**}**

**}**

**void init()**

**{**

**int i, j; // 반복 문을 위한 변수**

**int num; // 난수 생성을 위한 변수**

**blank\_i = 0; // 초기 빈칸 위치 선정**

**blank\_j = 0;**

**for (i = 0; i < rows; i++)**

**{**

**for (j = 0; j < cols; j++)**

**piece[i][j] = j + i\*rows;**

**// 배열의 초기값 설정, 0~rowsXcols-1까지 설정**

**}**

**srand((int)time(NULL)); // time으로 인한 시간에 따라 다르게 나타나는 난수 설정**

**for (i = 0; i < 10; i++)**

**// 카드를 섞을 땐, 빈칸이 움직이면서 섞여야 하기 때문에 난수의 경우의 수로 move함수로 이동하여 섞음.**

**{ // 10번 섞어준다.**

**num = rand() % 4; // 난수의 범위를 0~3의 숫자로 지정**

**if (num == 1) // 1일 때 위로 이동**

**move('w');**

**else if (num == 2) // 2일 때 좌로 이동**

**move("a");**

**else if (num == 3) // 3일 때 우로 이동**

**move("d");**

**else if (num == 0) // 0일 때 아래로 이동**

**move("x");**

**}**

**draw\_piece(blank\_i, blank\_j);**

**// 퍼즐 조각 그림 printf(빈칸의 위치엔 그림을 넣지 않는다.)**

**for (i = 0; i < rows; i++) // 퍼즐 조각을 섞은 후, 초기 퍼즐 숫자 출력**

**{**

**for (j = 0; j < cols; j++)**

**{**

**gotoxy(5 + j \* 6, 5 + i \* 3); // (5,5)부터 채워져 +6, +3씩 증가**

**if (piece[i][j] == rows\*cols)**

**// piece[i][j] 가 빈칸의 위치일 때, 숫자 지우기**

**printf(" ");**

**else**

**printf("%d ", piece[i][j]);**

**// 빈칸의 위치가 아닌 곳은 숫자 출력**

**}**

**printf("\n");**

**}**

**}**

**void draw\_piece(int i, int j) // i=blank\_i, j=blank\_j 빈칸의 위치 변수**

**{**

**int a, b, c; // gotoxy로 인한 위치 지정을 위한 변수**

**char piece\_of\_piece[3][7] = { // 퍼즐 조각 그림**

**{ "┌─┐" },**

**{ "│ │" },**

**{ "└─┘" }**

**};**

**char blank[3][7] = { // 빈칸의 위치엔 조각 그림을 지운다.**

**{ " " },**

**{ " " },**

**{ " " }**

**};**

**for (c = 0; c < rows; c++)**

**{**

**for (b = 0; b < cols; b++)**

**{**

**for (a = 0; a < 3; a++) // piece\_of\_piece[0]~[2]까지 출력**

**{**

**gotoxy(3 + b \* 6, a + c \* 3 + 4); // 처음에 (3,4)로 시작**

**printf("%s", piece\_of\_piece[a]);**

**}**

**}**

**}// 처음엔 빈칸의 위치에도 카드 그림을 출력함**

**for (a = 0; a < 3; a++)**

**{**

**gotoxy(i \* 6 + 3, j \* 3 + 4 + a); // 빈칸의 위치**

**puts(blank[a]); // 카드 조각 그림을 지운다.**

**}**

**}**

**void gotoxy(int x, int y)**

**{**

**COORD Pos = { x, y };**

**SetConsoleCursorPosition(GetStdHandle(STD\_OUTPUT\_HANDLE), Pos);**

**}**

**Extra 소스 파일**

**#include <stdio.h>**

**#include <stdlib.h>**

**#include <conio.h>**

**#include <time.h>**

**#include <Windows.h>**

**#define MINP 3**

**#define MAXP 7**

**#define OFFSET 5**

**#define KEY\_ESC 0x1B**

**#define KEY\_8 '8'**

**#define KEY\_2 '2'**

**#define KEY\_4 '4'**

**#define KEY\_6 '6'**

**#define KEY\_W 'w'**

**#define KEY\_X 'x'**

**#define KEY\_A 'a'**

**#define KEY\_D 'd'**

**#define UP 72 // 방향키 설정**

**#define DOWN 80**

**#define LEFT 75**

**#define RIGHT 77**

**int rows, cols; // board size 퍼즐 크기**

**int piece[MAXP][MAXP]; // pieces 퍼즐 조각**

**int blank\_i, blank\_j; // empty piece 빈 조각의 위치**

**int move\_count; // 이동 횟수**

**int Difficulty; // 난이도**

**int score; // 점수**

**void init(); // initialize**

**void move(int key); // slide a piece**

**int judge(); // judge the end of the game**

**void draw\_piece(int i, int j); // draw a piece**

**void gotoxy(int x, int y); // move cursor**

**int black\_word(void); // 초기시작화면 글자**

**void white\_board(void); // 글자 지우기**

**void start\_page(void); // 테두리**

**void infor(void); // 시작 설정**

**void Cursor(void); // 커서 지우기**

**void Score(void); // 점수 매기기**

**int main(void)**

**{**

**start\_page(); // 테두리 그림 함수**

**black\_word(); // 초기 시작 화면 글자 함수**

**white\_board(); // black\_word함수의 글자 지우기 함수**

**infor(); // 시작 시 설정 값 입력 및 조작 방법 printf 함수**

**init();**

**Cursor(); // 커서 깜빡임을 지우는 함수**

**Score(); // 점수 매기기 함수**

**gotoxy(60, 23);**

**printf("Done!");**

**gotoxy(0, 28);**

**return 0;**

**}**

**int judge() // Extra : 가로줄을 판단하여 첫 번째 가로줄을 정렬 했을 경우 알림을 표시한다.**

**{**

**int i, j;**

**int temp = 0;**

**int n = 0;**

**for (i = 0; i < rows; i++)**

**{**

**for (j = 0; j < cols; j++)**

**{**

**temp++; // temp == 1부터 시작**

**if ((temp - 1) % rows == 0)**

**// piece[j][i]가 제 위치에 자리 했을 때마다 temp 값이 1씩 증가됨**

**{**

**if (temp <= rows) // 0번째 줄 printf방지**

**continue;**

**gotoxy(46, 16 + n);**

**printf("◎ %d 번째 줄 완성!!!", temp / rows);**

**n++;**

**}**

**if (piece[i][j] != (j + 1) + i\*rows)**

**return 1;**

**}**

**}**

**gotoxy(46, 16 + n);**

**printf("◎ %d 번째 줄 완성!!!", temp / rows);**

**return 0;**

**}**

**void move(int key) // Extra : 방향키 추가 , 게임 재 시작**

**{**

**int i, j;**

**switch (key)**

**{**

**case KEY\_4: // left**

**case KEY\_A:**

**case LEFT:**

**if (blank\_i < rows - 1)**

**{**

**piece[blank\_j][blank\_i] = piece[blank\_j][blank\_i + 1];**

**blank\_i++;**

**}**

**break;**

**case KEY\_6: // right**

**case KEY\_D:**

**case RIGHT:**

**if (blank\_i > 0)**

**{**

**piece[blank\_j][blank\_i] = piece[blank\_j][blank\_i - 1];**

**blank\_i--;**

**}**

**break;**

**case KEY\_2: // down**

**case KEY\_X:**

**case DOWN:**

**if (blank\_j > 0)**

**{**

**piece[blank\_j][blank\_i] = piece[blank\_j - 1][blank\_i];**

**blank\_j--;**

**}**

**break;**

**case KEY\_8: // up**

**case KEY\_W:**

**case UP:**

**if (blank\_j < rows - 1)**

**{**

**piece[blank\_j][blank\_i] = piece[blank\_j + 1][blank\_i];**

**blank\_j++;**

**}**

**break;**

**case KEY\_ESC: // ECS**

**gotoxy(60, 23);**

**printf("\*\*Exit!\*\*\n\n");**

**gotoxy(0, 28);**

**exit(0);**

**break;**

**case'\t': // restart 기능 <TAB>**

**gotoxy(46, 14);**

**printf(" ");**

**init(); // 카드를 다시 섞는다.**

**move\_count = 0; // 움직인 횟수를 초기값으로 바꾼다.**

**Score(); // 스코어 점수를 초기값으로 바꾼다.**

**break;**

**}**

**piece[blank\_j][blank\_i] = rows\*cols;**

**draw\_piece(blank\_i, blank\_j);**

**for (i = 0; i < rows; i++)**

**{**

**for (j = 0; j < cols; j++)**

**{**

**gotoxy(5 + j \* 6, 5 + i \* 3);**

**if (piece[i][j] == rows\*cols)**

**printf(" ");**

**else**

**printf("%d ", piece[i][j]);**

**}**

**printf("\n");**

**}**

**}**

**void init() // Extra : 게임 난이도 설정, 경고음 추가**

**{**

**int i, j;**

**int num;**

**blank\_i = 0;**

**blank\_j = 0;**

**for (i = 0; i < rows; i++)**

**{**

**for (j = 0; j < cols; j++)**

**piece[i][j] = j + +i\*rows;**

**}**

**srand((int)time(NULL));**

**for (i = 0; i < Difficulty; i++) // Difficulty를 입력 받아 섞는 횟수 설정**

**{**

**// 섞는 횟수가 클수록 카드가 섞이는게 보임**

**num = rand() % 5;**

**switch (num)**

**{**

**case 1:**

**move('w');**

**break;**

**case 2:**

**move('x');**

**break;**

**case 3:**

**move('a');**

**break;**

**case 4:**

**move('d');**

**}**

**}**

**draw\_piece(blank\_i, blank\_j);**

**printf("\a"); // 경고음 추가**

**for (i = 0; i < rows; i++)**

**{**

**for (j = 0; j < cols; j++)**

**{**

**gotoxy(5 + j \* 6, 5 + i \* 3); // (5,5)부터 채워져 +6, +3씩 증가**

**if (piece[i][j] == 0 || piece[i][j] == rows\*cols)**

**printf(" ");**

**else**

**printf("%d ", piece[i][j]);**

**}**

**printf("\n");**

**}**

**}**

**void draw\_piece(int i, int j)**

**{**

**int a, b, c;**

**char piece\_of\_piece[3][7] = {**

**{ "┌─┐" },**

**{ "│ │" },**

**{ "└─┘" }**

**};**

**char blank[3][7] = {**

**{ " " },**

**{ " " },**

**{ " " }**

**};**

**for (c = 0; c < rows; c++)**

**{**

**for (b = 0; b < cols; b++)**

**{**

**for (a = 0; a < 3; a++)**

**{**

**gotoxy(3 + b \* 6, a + c \* 3 + 4); // 처음에 (3,4)로 시작**

**printf("%s", piece\_of\_piece[a]);**

**}**

**}**

**}**

**for (a = 0; a < 3; a++)**

**{**

**gotoxy(i \* 6 + 3, j \* 3 + 4 + a);**

**puts(blank[a]);**

**}**

**}**

**void gotoxy(int x, int y)**

**{**

**COORD Pos = { x, y };**

**SetConsoleCursorPosition(GetStdHandle(STD\_OUTPUT\_HANDLE), Pos);**

**} // 커서 이동**

**void start\_page(void) // Extra : 게임 테두리 그리기**

**{**

**int i;**

**printf("□■□■□■□■□■□■□■□■□■□■□■□■□■□■□■□■□■□■□■□■□\n");**

**for (i = 0; i < 13; i++) // 좌측 세로줄 그리기**

**{**

**printf("■\n");**

**printf("□\n");**

**}**

**for (i = 0; i < 13; i++) // 우측 세로줄 그리기**

**{**

**gotoxy(80, i \* 2); // □이 짝수 세로줄에 printf된다 & ■이 홀수 세로줄에 printf된다.**

**printf("□");**

**gotoxy(80, i \* 2 + 1);**

**printf("■");**

**}**

**gotoxy(0, 26);**

**printf("□■□■□■□■□■□■□■□■□■□■□■□■□■□■□■□■□■□■□■□■□\n");**

**gotoxy(82, 25);**

**printf(" Seoul Tech, Computer Programming");**

**gotoxy(82, 26);**

**printf(" 12184407 이상운");**

**}**

**int black\_word(void) // 게임을 실행 시 글자 printf("SLIDE GAME")**

**{**

**int i = 0;**

**gotoxy(10, 6);**

**printf(" ■■■ ■ ■■■ ■■■ ■■■■");**

**gotoxy(10, 7);**

**printf("■ ■ ■ ■ ■ ■");**

**gotoxy(10, 8);**

**printf(" ■■■ ■ ■ ■ ■ ■■■■");**

**gotoxy(10, 9);**

**printf(" ■ ■ ■ ■ ■ ■");**

**gotoxy(10, 10);**

**printf("■■■ ■■■■ ■■■ ■■■ ■■■■");**

**gotoxy(25, 15);**

**printf(" ■■■■ ■ ■ ■ ■■■■");**

**gotoxy(25, 16);**

**printf("■ ■■ ■■ ■■ ■");**

**gotoxy(25, 17);**

**printf("■ ■■ ■ ■ ■ ■ ■ ■ ■■■■");**

**gotoxy(25, 18);**

**printf("■ ■ ■■■■ ■ ■■ ■ ■");**

**gotoxy(25, 19);**

**printf(" ■■■■ ■ ■ ■ ■ ■ ■■■■");**

**gotoxy(23, 23);**

**printf("--Press any key to start!--");**

**getch(i);**

**if (i == '\n') // 아무 키나 누를 시 다음 함수로 넘어감.**

**return 0;**

**}**

**void white\_board(void) // black\_word함수의 글자를 지워준다.**

**{**

**int i;**

**for (i = 2; i < 25; i++)**

**{**

**gotoxy(3, i);**

**printf(" ");**

**}**

**}**

**void infor(void) // Extra : 게임 난이도 설정**

**{**

**gotoxy(3, 2);**

**printf("Enter a size number (%d~%d) : ", MINP, MAXP);**

**scanf("%d", &rows);**

**gotoxy(50, 3);**

**printf("<1.Esay> <2.Normal> <3.Hard>"); // 게임의 난이도 설정**

**gotoxy(50, 2);**

**printf("Set-up Difficulty : "); // Difficulty 전역 변수에 입력**

**scanf("%d", &Difficulty);**

**if (rows < MINP || rows > MAXP)**

**rows = MINP;**

**switch (Difficulty)**

**{**

**case 1: // 1 입력 시 10번 섞는다.**

**Difficulty = 10;**

**break;**

**case 2: // 2 입력 시 50번 섞는다.**

**Difficulty = 50;**

**break;**

**case 3: // 3 입력 시 100번 섞는다.**

**Difficulty = 100;**

**break;**

**default: // 그 외의 숫자 입력 시 50번 섞는다.**

**Difficulty = 50;**

**}**

**cols = rows;**

**gotoxy(50, 5);**

**printf("⊙ KEY ⊙");**

**gotoxy(50, 10);**

**printf("Press ESC to exit.\n");**

**gotoxy(50, 11);**

**printf("Press TAB to Restart");**

**gotoxy(50, 6);**

**printf(" △ UP(8, W)");**

**gotoxy(50, 7);**

**printf("◁ ▷Left(4, A) / Right(6, D)");**

**gotoxy(50, 8);**

**printf(" ▽ Down(2, X)");**

**}**

**void Cursor(void) // 커서를 지워주는 라이브러리 함수**

**{**

**CONSOLE\_CURSOR\_INFO cur;**

**GetConsoleCursorInfo(GetStdHandle(STD\_OUTPUT\_HANDLE), &cur);**

**cur.bVisible = 0;**

**SetConsoleCursorInfo(GetStdHandle(STD\_OUTPUT\_HANDLE), &cur);**

**}**

**void Score(void) // Extra : 게임 점수 제 설정, 이동 횟수 표시**

**{**

**score = rows\*cols\*Difficulty \* 100; // 초기 점수 값 (난이도와 퍼즐 개수에 따라 점수가 다름)**

**gotoxy(46, 13);**

**printf("★ MOVE COUNT ★ ★ SCORE ★\n");**

**do**

**{**

**gotoxy(46, 14);**

**printf(" %d %d", move\_count/2, score);**

**move(getch());**

**++move\_count; // move\_count 증가**

**if (move\_count >= 100) // move\_count가 증가 할 때마다 score점수는 깎인다.**

**score -= 10; // 많이 증가할수록 점수는 더 많이 감소한다.**

**else if (move\_count >= 50)**

**score -= 5;**

**else if (move\_count >= 10)**

**score -= 2;**

**else**

**score -= 1;**

**if (score < 0) // 점수가 음수일 시, 게임 종료**

**{**

**gotoxy(60, 22);**

**printf("Game OVER!!!!");**

**break; // judge함수와 상관 없이 do~while문 탈출**

**}**

**} while (judge());**

**}**