게임프로그래밍
과자먹기 게임(응용8_9_1.cpp)

소프트웨어학과 2020875029 신성우

목차

기존 소스 분석(main)

```c
int main(void)
{
    int i, start, end, cake_left=cake_number, winner;
    int cake_condition[cake_number]={0};
    char user_name[2][8];
    srand(time(NULL));

    intro_game();
    input_participant(user_name);

    start=0;
    end=cake_number-1;

    system("cls");
    game_control(user_name, cake_condition, &cake_left, 2,  &start, &end);  //user의 값을 2로 하여 과자의 초기상태를 출력

    gotoxy(10, 12);
    printf("아무키나 누르면 다음 순서를 진행합니다. ");
    getch();
  do
  {
     for(i=0;i<2;i++)
     {
        system("cls");
        game_control(user_name, cake_condition, &cake_left, i, &start, &end);
        if (cake_left<2)
        {
           winner=i;
           break;
        }
        gotoxy(10, 12);
        printf("아무키나 누르면 다음 순서를 진행합니다. ");
        getch();
     }
  }while(cake_left>2);

  gotoxy(10, 12);
  printf("%s님이 이겼습니다. ", user_name[winner]);
  gotoxy(10, 13);
  printf("게임을 종료합니다. \n");
  return 0;
```

- 전체 프로그램의 메인 루프
- 초기화 -> 이름 입력 -> 반복 턴 -> 결과 출력 순서대로 루프

기존 소스 분석(intro)

```
void intro_game(void)
{
    system("cls");
    printf("주사위로 과자먹기 게임 ₩n₩n");
    printf("두사람이 서로 양끝의 주사위 숫자만큼₩n");
    printf("과자를 먹는 게임입니다. ₩n");
    printf("마지막 남은 과자를 먹는 사람이 이깁니다. ₩n₩n");
    printf("아무키나 누르면 게임참가자를₩n");
    printf("입력합니다.₩n");
    getch();

}
```

- 게임 규칙 설명

기존 소스 분석(user)

```
void input_participant(char user_name[][8])
{
    system("cls");
    printf("1번 참가자의 이름을 입력하고 Enter>");
    scanf("%s", user_name[0]);
    printf("2번 참가자의 이름을 입력하고 Enter>");
    scanf("%s", user_name[1]);
    printf("아무키나 누르면 게임을 시작합니다...");
    getch();

}
```

- 플레이어 1, 2의 이름 입력 받음
- 매개변수 이용 이름 저장

기존 소스 분석(control)

```
void game_control(char name[][8], int condition[],  int *left, int user, int *s, int *e)
{
    int i, dice_number;
    cake_display(name, condition, *left, *s, *e);

    if (user==2)
        return;  //user가 2가 되는 경우는 main으로 복귀
    dice_number=rand()%6+1;  //주사위 난수 생성
    *left-=dice_number;
    gotoxy(10, 11);
        printf("%s님의 주사위 숫자는 %d입니다.",name[user],dice_number);

    if (user==0)
    {
        for(i=*s;i<dice_number+*s;i++)
            condition[i]=1;
        *s+=dice_number;
    }
    else
    {
        for(i=*e;i>(*e-dice_number);i--)
            condition[i]=1;
        *e-=dice_number;
    }
    cake_display(name, condition, *left, *s, *e);
}
```

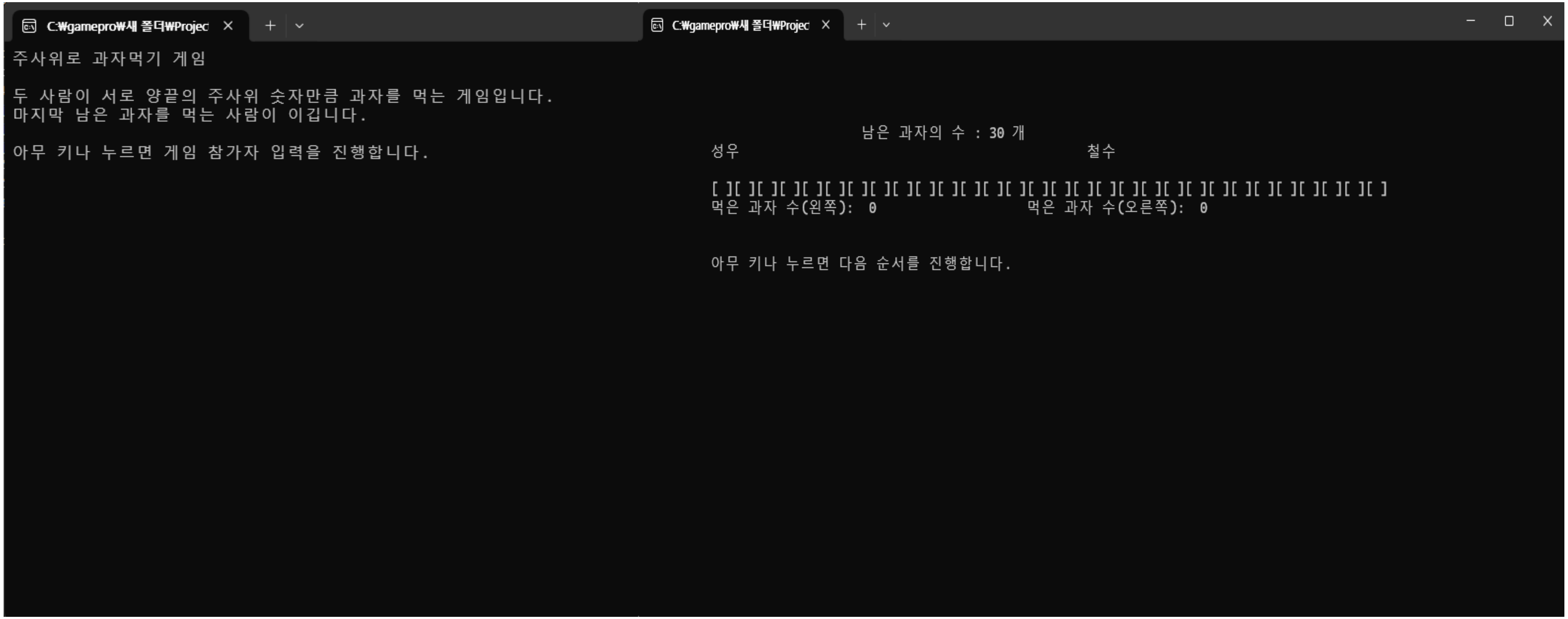
- 주사위를 굴려서 해당 플레이어가 과자를 먹고 상태 갱신

기존 소스 분석(display)

```
void cake_display(char name[][8], int condition[], int left, int s, int e)
{
    int i;
    char *eat_cake="■", *remain_cake="□";

    gotoxy(30,5);
    if (left<0)
        left=0;
    printf("남은 과자의 수 : %2d 개 ", left);

    for(i=0;i<2;i++)
    {
        gotoxy(i*50+10, 6);
        printf("%s", name[i]);
    }

    for(i=0;i<30;i++)
        if (condition[i]==1)
        {
            gotoxy(10+i*2, 8);
            printf("%s", eat_cake);
        }
        else
        {
            gotoxy(10+i*2, 8);
            printf("%s", remain_cake);
        }

    gotoxy(10, 9);
    printf("먹은 과자 수: %2d", s);
    gotoxy(52, 9);
    printf("먹은 과자 수: %2d", 29-e);
}
```

- 남은 수, 각자 이름, 과자 30개 상태, 먹은 개수 출력

기존 소스 실행 화면


C:₩gameproW새 폴더₩Projec
주사위로 과자먹기 게임
두 사람이 서로 양끝의 주사위 숫자만큼 과자를 먹는 게임입니다.
마지막 남은 과자를 먹는 사람이 이깁니다.
아무 키나 누르면 게임 참가자 입력을 진행합니다.
C:₩gameproW새 폴더₩Projec
남은 과자의 수 : 30 개
성우
철수
[ ]
먹은 과자 수(왼쪽): 0
먹은 과자 수(오른쪽): 0
아무 키나 누르면 다음 순서를 진행합니다.

업그레이드 소스 분석(main)

```
// 폭탄 수 1~2개 랜덤 배치(숨김)
int bomb_count = 1 + (rand() % 2);
place_bombs_random(bomb_hidden, CAKE_NUMBER, bomb_count);

intro_game();
input_participant(user_name);

start = 0;
end = CAKE_NUMBER - 1;
do {
    for (i = 0; i < 2; i++) {
        system("cls");
        int ret = game_control(user_name, cake_condition, bomb_hidden, &cake_left, i, &start, &end);
        if (ret == 1) {
            gameOver = 1;
            break;
        }
        if (cake_left <= 0) {
            winner = i;
            gameOver = 1;
            break;
        }
        gotoxy(10, 12);
        set_color(COL_DEFAULT);
        printf("아무 키나 누르면 다음 순서를 진행합니다. ");
        getch();
    }
} while (!gameOver && cake_left > 0);
```

- 폭탄 기능, 배경음, 색상, 즉시 패배 로직이 추가

업그레이드 소스 분석(intro)

```
void intro_game(void)
{
    system("cls");
    set_color(COL_DEFAULT);
    printf("주사위로 과자먹기 게임 ₩n₩n");
    printf("두 사람이 서로 양끝의 주사위 숫자만큼 과자를 먹는 게임입니다.₩n");
    printf("폭탄 과자를 먹으면 즉시 패배합니다!₩n₩n");

    draw_cookie_ascii(20, 8);
    countdown_start(3);

    // MP3 재생
    mci_play_mp3("bgm.mp3");
    Sleep(500);
}
```

- 쿠키 아트 + 카운트다운 + MP3 재생 추가

업그레이드 소스 분석(control)

```
if (user == 0) { // 왼쪽부터 먹기
    for (i = 0; i < dice_number && *s <= *e; i++) {
        if (bomb[*s]) { // 폭탄이면 즉시 패배 처리
            bomb[*s] = 0;
            condition[*s] = 2; // 폭탄 먹힘(표시)
            (*left)--; if (*left < 0) *left = 0;
            cake_display(name, condition, *left, *s, *e);
            gotoxy(10, 13);
            set_color(COL_YELLOW);
            printf("[!] %s님이 폭탄 과자를 먹었습니다! (패배)", name[user]);
            return 1;
        }
        if (condition[*s] == 0) {
            condition[*s] = 1;     // 일반 과자 먹음
            (*left)--; if (*left < 0) *left = 0;
        }
        (*s)++;
    }
}
```

- 폭탄 검출 및 즉시 패배 처리 로직 추가

업그레이드 소스 분석(display)

```
void cake_display(char name[][8], int condition[], int left, int s, int e)
{
    int i;
    const char* EAT = "[X]";  // 먹은 과자(초록)
    const char* NORM = "[ ]";  // 남은 과자(기본)
    const char* BOOM = "[!]";  // 폭탄 먹힌 과자(빨강)

    gotoxy(30, 5);
    set_color(COL_BLUE);
    if (left < 0) left = 0;
    printf("남은 과자의 수 : %2d 개 ", left);

    set_color(COL_DEFAULT);
    for (i = 0; i < 2; i++) {
        gotoxy(i * 50 + 10, 6);
        printf("%s", name[i]);
    }

    gotoxy(10, 7);
    set_color(COL_GREEN); printf("%s", EAT);
    set_color(COL_DEFAULT); printf("=먹음   ");
    set_color(COL_RED); printf("%s", BOOM);
    set_color(COL_DEFAULT); printf("=폭탄(먹힘)   ");
    printf("%s=남음", NORM);
```

```
for (i = 0; i < CAKE_NUMBER; i++) {
    gotoxy(10 + i * 3, 9);
    if (condition[i] == 1) {
        set_color(COL_GREEN);
        printf("%s", EAT);
    }
    else if (condition[i] == 2) {
        set_color(COL_RED);
        printf("%s", BOOM);
    }
    else {
        set_color(COL_DEFAULT);
        printf("%s", NORM);
    }
}

// 먹은 개수 표기
set_color(COL_DEFAULT);
gotoxy(10, 10);
printf("먹은 과자 수(왼쪽): %2d", s);
gotoxy(52, 10);
printf("먹은 과자 수(오른쪽): %2d", (CAKE_NUMBER - 1) - e >= 0 ? (CAKE_NUMBER - 1) - e : 0);
```

-컬러 추가 + 폭탄/먹음 표시 구분 추가

업그레이드 소스 분석(인트로 화면)

```
void draw_cookie_ascii(int x, int y)
{
    const char* art[] = {
        "          .-╦"╦"╦"╦"╦"-.          ",
        "        .'   .-╦"╦"-.   '.        ",
        "      /   /   ..  ╦╦╦   ╦╦╦       ",
        "     ;   ;  (..)  ;   ;     ",
        "     |   |   ..   |   |     ",
        "     ;   ;  (  )  ;   ;     ",
        "      ╦╦╦   ╦╦╦  ''  /   /      ",
        "        '.   '-..-'   .'        ",
        "          '-.____.-'           ",
    };
    int lines = (int)(sizeof(art) / sizeof(art[0]));
    for (int i = 0; i < lines; i++) {
        gotoxy(x, y + i);
        printf("%s", art[i]);
    }
    gotoxy(x, y + lines + 1);
    set_color(COL_YELLOW);
    printf("맛있는 쿠키가 준비됐어요!");
    set_color(COL_DEFAULT);
}
```

- 인트로용 쿠키 그림 출력 기능 추가

업그레이드 소스 분석(카운트다운)

```
void countdown_start(int seconds)
{
    for (int s = seconds; s > 0; s--) {
        int freq;
        if (s >= 3)      freq = 261;   // C4
        else if (s == 2) freq = 329;   // E4
        else             freq = 392;   // G4

        gotoxy(10, 20);
        set_color(COL_YELLOW);
        printf("곧 시작합니다... %d  ", s);

        Beep(freq, 180);   // 180ms 동안 울림
        Sleep(820);        // 남은 1초 동안 대기
    }

    gotoxy(10, 20);
    set_color(COL_DEFAULT);
    printf("게임을 시작합니다!     ");
    Beep(523, 200); // C5
    Sleep(500);
}
```

- 인트로 화면 전환 시 카운트다운 시 저음 비프 출력 기능 추가

업그레이드 소스 분석(폭탄)

```
void place_bombs_random(int bomb[], int n, int count)
{
    int placed = 0;
    while (placed < count) {
        int idx = rand() % n;
        if (!bomb[idx]) { bomb[idx] = 1; placed++; }
    }
}
```

- 폭탄 무작위 배치 기능 추가

업그레이드 소스 분석(bgm)

```
void mci_play_mp3(const char* path)
{
    char cmd[512];


    mciSendStringA("stop bgm", NULL, 0, NULL);
    mciSendStringA("close bgm", NULL, 0, NULL);


    _snprintf(cmd, sizeof(cmd), "open \"%s\" type mpegvideo alias bgm", path);
    if (mciSendStringA(cmd, NULL, 0, NULL) != 0) {
        // 실패 시 그냥 무시(또는 Beep 등 폴백 가능)
        return;
    }

    mciSendStringA("play bgm from 0", NULL, 0, NULL);
}
```

- 플레이어 이름 선택 시 bgm 기능 추가

업그레이드 소스 실행 화면