|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **C2\_실습과제\_14주** | **이름 :** | **이갑성** | **학번 :** | **20135151** |

* **강의내용**

1. 좌우 이동

#include <stdio.h>

#include <stdlib.h>

#include <windows.h>

void gotoxy(int , int );

int main(void){

int x=0, y=1, temp=2, count=0;

while(count<3){

x+=temp;

if (x>20) // x의 최대값

temp=-2;

if (x<2){ // x의 최소값

temp=2;

count++;

}

system("cls");

gotoxy(x, y);

printf("□");

Sleep(50); //움직이는 속도를 조절

}

printf("\n");

return 0;

}

void gotoxy(int x, int y){

COORD Pos = { x - 1, y - 1 };

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

}

|  |
| --- |
| <결과> |

1. 교차 이동

int main(void){

unsigned char ch1[]={0xa1, 0xe1, 0}; // "■"

unsigned char ch2[]={0xa1, 0xdf, 0}; // "◆"

int i, check=1;

for(i=1;i<40;i+=2){

gotoxy(i,5);

if (check==1){

printf("%s",ch1);

check=0;}

else{

printf("%s",ch2);

check=1;}

Sleep(150);

gotoxy(i,5); printf(" "); // 대상지우기

}

printf("\n"); return 0; }

|  |
| --- |
| <결과> |

1. 공 튀기기

int main(){

unsigned char ch1[]= {0xa1, 0xdc, 0}; //"○"

int y, check=24;

do{

for(y=25-check;y<24;y+=1){

gotoxy(20,y); printf("%s",ch1);

Sleep(50); //공이 바닥으로 내려오는 시간 조절

gotoxy(20,y); printf(" ");

}

check=(check-check/2); // 공의높이계산

for(y=24;y>24-check;y-=1){

gotoxy(20,y); printf("%s",ch1);

Sleep(70); //공이 바닥에서 튀어 오르는 시간

gotoxy(20,y); printf(" ");

}

}while(check!=1);

printf("\n"); return 0;}

|  |
| --- |
| <결과> |

1. 시간처리

void game\_control(void){ //게임에 대한 전반적인 제어 함수

int cnt[MAX\_NUMBER], answer, user;

clock\_t start, end;

double pst;

system("cls");

print\_number(MATRIX\_SIZE, cnt);

answer = find\_num(cnt);

printf("\n\n"); printf("출현횟수가 가장 많은 숫자 입력>");

start = clock(); //시작 시간

while (!\_kbhit()){ //임의의 키가 눌렸는지를 검사

end = clock(); //끝 시간

pst = (double)(end - start) / CLK\_TCK; //경과 시간 검사

if (TIME\_LIMIT<pst){

printf("\n제한시간을 넘었습니다. \n");

exit(0);

}

}

user = \_getch() - '0';

printf("%d", user);

}

|  |
| --- |
| <소스>  #include <stdio.h>  #include <stdlib.h>  #include <windows.h>  #include <time.h>  #include <conio.h>  #define MATRIX\_SIZE 5  #define MAX\_NUMBER 5  #define TIME\_LIMIT 5  void game\_control(void);  void print\_number(int n, int cnt[]);  int find\_num(int cnt[]);  int main()  {  time\_t start, end;  long i = 0;  double pst;  start = clock();  while(i < 300000000)  {  i++;  }  end = clock();  pst = (double) (end-start)/CLK\_TCK;  printf("time: %f \n", pst);    return 0;  }  void print\_number(int n, int cnt[])  {  int i, j, r\_num;  for(i = 0; i < MAX\_NUMBER; i++)  {  cnt[i] = 0;  }  for(i = 1; i <= n; i++)  {  for(j = 1; j <= n; j++)  {  r\_num = rand() % MAX\_NUMBER;  cnt[r\_num] += 1;  printf("%d", r\_num);  }  printf("\n");  }  }  void intro\_game(void){ //소개  printf("정방행렬에 나타난 숫자 중에서\n");  printf("가장 많이 출현한 숫자를 찾아라!\n\n");  printf("제한시간은 5초입니다.\n\n");  printf("아무키나 누르면 시작합니다. \n");  \_getch();  }  void game\_control(void){ //게임에 대한 전반적인 제어 함수  int cnt[MAX\_NUMBER], answer, user;  clock\_t start, end;  double pst;  system("cls");  print\_number(MATRIX\_SIZE, cnt);  answer = find\_num(cnt);  printf("\n\n");  printf("출현횟수가 가장 많은 숫자 입력>");  start = clock(); //시작 시간  while (!\_kbhit()){ //임의의 키가 눌렸는지를 검사  end = clock(); //끝 시간  pst = (double)(end - start) / CLK\_TCK; //경과 시간 검사  if (TIME\_LIMIT<pst){  printf("\n제한시간을 넘었습니다. \n");  exit(0);  }  }  user = \_getch() - '0';  printf("%d", user);  }  int find\_num(int cnt[])  {  int max = 0;  int size = sizeof(cnt) / sizeof(int);  int i;  for(i = 0; i < size; i++)  {  if(max < cnt[i])  {  max = cnt[i];  }  }  return max;  } |
| <결과> |

1. 과자 처리하기

#include <stdio.h>

#include <stdlib.h>

#include <time.h>

#include <windows.h>

#include <conio.h>

#define cake\_number 30 //과자의개수

void intro\_game(void);

void input\_participant(char user\_name[][8]);

void game\_control(char name[][8], int condition[], int \*left, int user, int \*start, int \*end);

void cake\_display(char name[][8], int condition[], int left, int start, int end);

void gotoxy(int x, int y);

int main(void){

int i, start, end, cake\_left = cake\_number, winner;

int cake\_condition[cake\_number] = { 0 }; //과자 상태를 나타내는 배열, 모두 0으로 초기화

char user\_name[2][8]; //사용자 이름 저장

srand((long)time(NULL)); //난수 생성을 위한 시드값

intro\_game();

input\_participant(user\_name);

start = 0;

end = cake\_number - 1;

system("cls");

//user의 값을 2로 하여 과자의 초기상태를 출력

game\_control(user\_name, cake\_condition, &cake\_left, 2, &start, &end);

gotoxy(10, 12);

printf("아무키나 누르면 다음 순서를 진행합니다. ");

\_getch();

do{

for (i = 0; i<2; i++){

system("cls");

//사용자를 구분하기 위해 i 값을 전달(user\_name[2][8])

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); //과자가 1개 남을때 까지 반복

gotoxy(10, 12);

printf("%s님이 이겼습니다. ", user\_name[winner]);

gotoxy(10, 13);

printf("게임을 종료합니다. \n");

return 0;

}

void input\_participant(char user\_name[][8])

{

system("cls");

printf("1번 참가자의 이름을 입력하고Enter>");

scanf\_s("%s", user\_name[0], 8);

printf("2번 참가자의 이름을 입력하고Enter>");

scanf\_s("%s", user\_name[1], 8);

printf("아무키나 누르면 게임을 시작합니다...");

\_getch();

}

void intro\_game(void)

{

system("cls");

printf("주사위로 과자먹기게임\n\n");

printf("두 사람이 서로양끝의 주사위 숫자만큼\n");

printf("과자를 먹는 게임입니다. \n");

printf("마지막 남은과자를 먹는사람이 이깁니다. \n\n");

printf("아무키나 누르면 게임 참가자를\n");

printf("입력합니다.\n");

\_getch();

}

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 과자수: %2d", s);

gotoxy(52, 9);

printf("먹은 과자수: %2d", 29 - e);

}

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) //user가 2가 되는 경우는 main으로 복귀

return;

dice\_number = rand() % 6 + 1; //주사위 난수생성

\*left -= dice\_number; //남은 과자 수

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; //생성한 난수 만큼 다음 시작위치를 증가(왼쪽에서 시작)

}

else{

for (i = \*e; i>(\*e - dice\_number); i--)

condition[i] = 1;

\*e -= dice\_number; //생성한 난수 만큼 다음 시작이치를 감소(오늘쪽에서 시작)

}

cake\_display(name, condition, \*left, \*s, \*e);

}

void gotoxy(int x, int y){

COORD Pos = { x - 1, y - 1 };

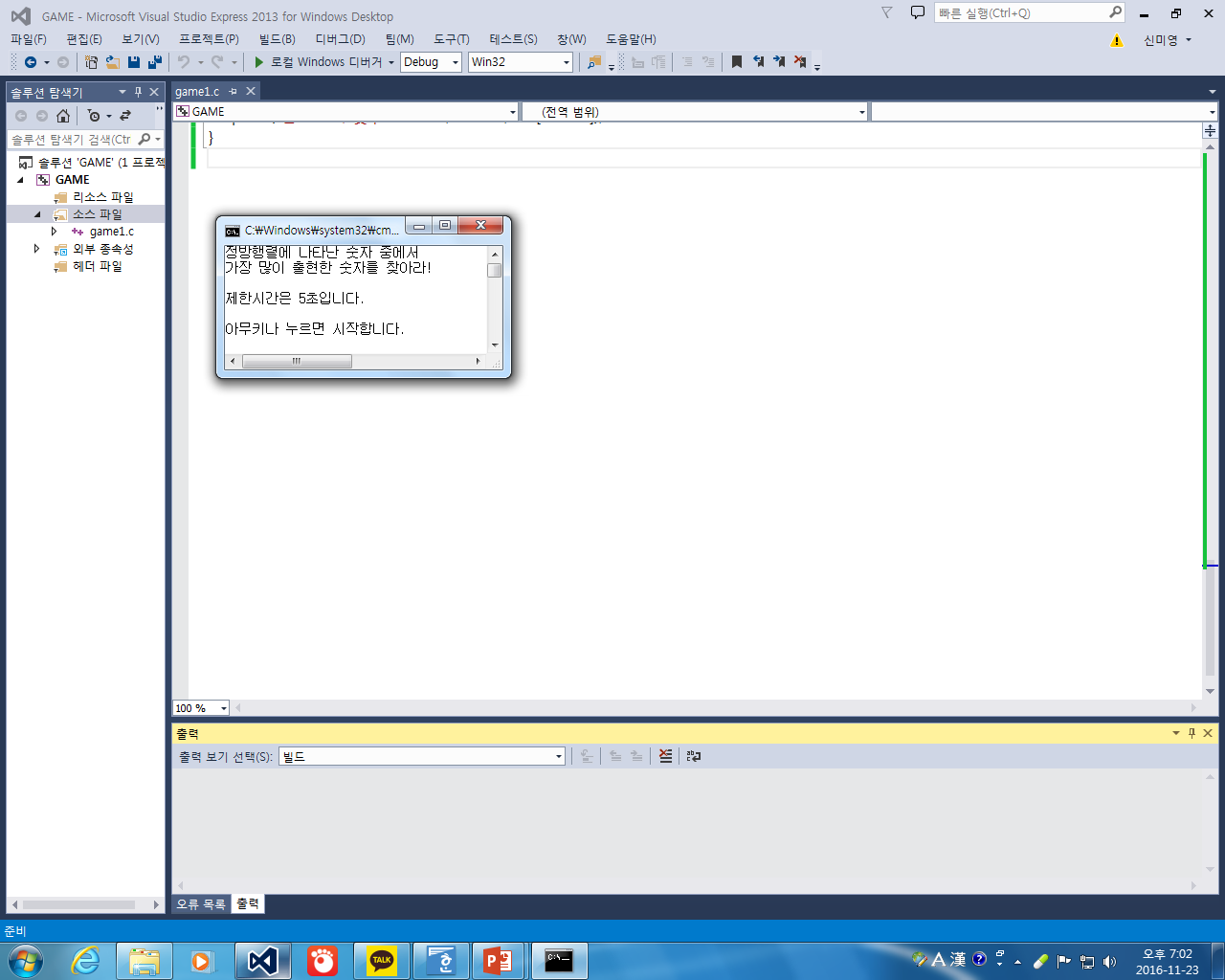
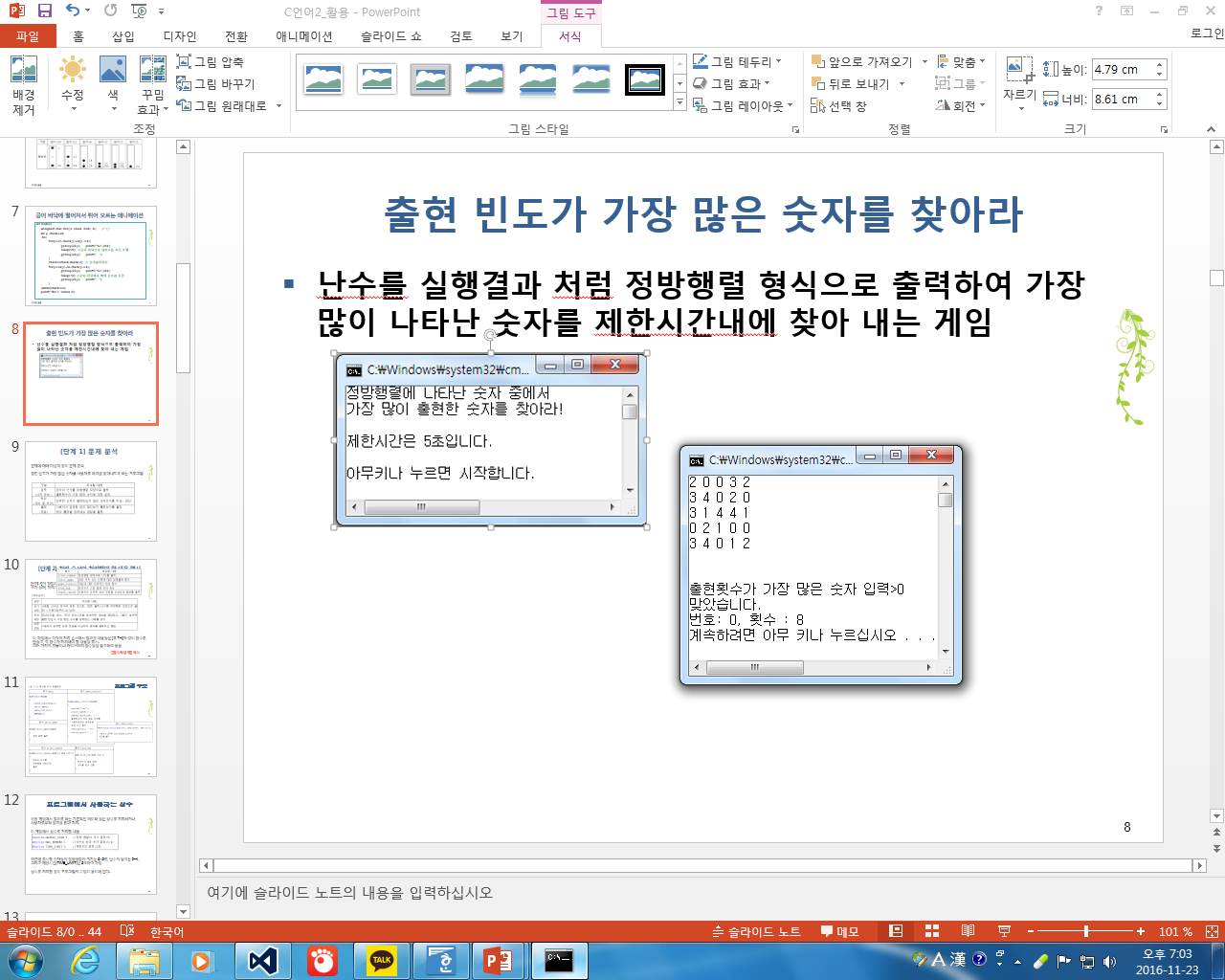
SetConsoleCursorPosition(GetStdHandle(STD\_OUTPUT\_HANDLE), Pos);

}

|  |
| --- |
| <결과> |

* 프로그램 과제

1. 출현 횟수가 많은 숫자와 적은 숫자를 동시에 맞추는 프로그램을 작성하시오.



|  |
| --- |
| <소스>  #include <stdio.h>  #include <time.h>  #include <conio.h>  #include <stdlib.h>  #define MATRIX\_SIZE 5  #define MAX\_NUMBER 5  #define TIME\_LIMIT 5  void intro\_game(void); //소개  void game\_control(void); //전반적인 제어  void print\_number(int n, int cnt[]); //정방 행렬 출력  int find\_num(int cnt[], int, int\* maxcnt); //빈도수가 가장 많은 숫자를 계산하여 반환  int lessFind\_num(int cnt[], int, int\* mincnt); //빈도수가 가장 적은 숫자를 계산하여 반환  void check\_result(int user[], int answer[], int \*maxcnt, int \*mincnt); //사용자 입력 값과 정답 비교, 결과 출력  int main(void){  srand((long)time(NULL)); //시드값 배정  intro\_game();  game\_control();  return 0;  }  void print\_number(int n, int cnt[])  {  int i, j, r\_num;  for(i = 0; i < MAX\_NUMBER; i++)  {  cnt[i] = 0;  }  for(i = 1; i <= n; i++)  {  for(j = 1; j <= n; j++)  {  r\_num = rand() % MAX\_NUMBER;  cnt[r\_num] += 1;  printf("%d", r\_num);  }  printf("\n");  }  }  void intro\_game(void){ //소개  printf("정방행렬에 나타난 숫자 중에서\n");  printf("가장 많이 출현한 숫자를 찾아라!\n\n");  printf("제한시간은 5초입니다.\n\n");  printf("아무키나 누르면 시작합니다. \n");  \_getch();  }  void game\_control(void){ //게임에 대한 전반적인 제어 함수  int cnt[MAX\_NUMBER], answer[2], user[2];  clock\_t start, end;  double pst;  int size = sizeof(cnt)/sizeof(int);  int maxcnt=0;  int mincnt=0;  system("cls");  print\_number(MATRIX\_SIZE, cnt);  answer[0] = find\_num(cnt, size, &maxcnt); //가장많은 숫자 답 저장  printf("가장 많은 숫자: %d ", answer[0]);  answer[1] = lessFind\_num(cnt, size, &mincnt); //가장적은 숫자 답 저장  printf("가장 적은 숫자: %d", answer[1]);  printf("\n\n");  printf("출현횟수가 가장 많은 숫자 입력> \n");  start = clock(); //시작 시간  while (!\_kbhit()) //임의의 키가 눌렸는지를 검사  {  end = clock(); //끝 시간  pst = (double)(end - start) / CLK\_TCK; //경과 시간 검사  if (TIME\_LIMIT<pst)  {  printf("\n제한시간을 넘었습니다. \n");  exit(0);  }  }  printf("가장많은 숫자 >>");  scanf("%d", &user[0]);  printf("\n");  printf("가장적은 숫자 >>");  scanf("%d", &user[1]);  printf("\n");  printf("입력한 정답 : 가장 많은 숫자: %d 가장 적은 숫자: %d", user[0], user[1]);  printf("\n<< 정답 확인 >> \n\n");  check\_result(user, answer, &maxcnt, &mincnt);  }  int find\_num(int cnt[], int size, int \*maxcnt)  {  \*maxcnt = cnt[0];  int maxIndex = 0;  int i;  for(i = 1; i < size; i++)  {  if(\*maxcnt < cnt[i])  {  \*maxcnt = cnt[i];  maxIndex = i;  }  }  return maxIndex;  }  int lessFind\_num(int cnt[], int size, int \*mincnt)  {  \*mincnt = cnt[0];  int minIndex = 0;  int i;  for(i = 1; i < size; i++)  {  if(\*mincnt > cnt[i])  {  \*mincnt = cnt[i];  minIndex = i;  }  }    return minIndex;  }  void check\_result(int user[], int answer[], int \*maxcnt, int \*mincnt)//사용자 입력 값과 정답 비교, 결과 출력  {  if(user[0] == answer[0])  {  printf("가장 많은 숫자를 입력한 것은 정답입니다.\n");  printf("가장 많은 숫자의 개수: %d \n", \*maxcnt);  }  else  printf("가장 많은 숫자를 입력한 것은 틀렸습니다. \n");  if(user[1] == answer[1])  {  printf("가장 적은 숫자를 입력한 것은 정답입니다. \n");  printf("가장 적은 숫자의 개수: %d \n", \*mincnt);  }  else  printf("가장 적은 숫자를 입력한 것은 틀렸습니다. \n");  } |
| <결과> |

1. 3번 예제에서 공이 튀는 모양이 포물선이 되도록 수정하시오.

|  |
| --- |
| <소스>  #include <stdio.h>  #include <stdlib.h>  #include <windows.h>  void gotoxy(int , int );  int main(){  unsigned char ch1[]= {0xa1, 0xdc, 0}; //"○"  int y, check=24;  int x = 0;  int temp = 2;  int cnt = 0;  do{  for(y=25-check;y<24;y+=1){  gotoxy(x,y); printf("%s",ch1);  x++;  Sleep(50); //공이 바닥으로 내려오는 시간 조절  gotoxy(x,y); //printf(" ");      }  check=(check-check/2); // 공의높이계산  for(y=24;y>24-check;y-=1){  gotoxy(x,y); printf("%s",ch1);  x++;    Sleep(70); //공이 바닥에서 튀어 오르는 시간  gotoxy(x,y); //printf(" ");        }  }while(check!=1);  printf("\n"); return 0;  }  void gotoxy(int x, int y){  COORD Pos = { x - 1, y - 1 };  SetConsoleCursorPosition(GetStdHandle(STD\_OUTPUT\_HANDLE), Pos);  } |
| <결과> |

1. 예제5번의 과자처리 프로그램에서 한 번에 두개의 주사위를 던져 그 차이만큼 과자를 먹으며 많이 먹은 사람이 승자가 되도록 프로그램을 수정하시오.

|  |
| --- |
| <소스>  #include <stdio.h>  #include <stdlib.h>  #include <time.h>  #include <windows.h>  #include <conio.h>  #define cake\_number 30 //과자의개수  void intro\_game(void);  void input\_participant(char user\_name[][8]);  void game\_control(char name[][8], int condition[], int \*left, int user, int \*start, int \*end);  void cake\_display(char name[][8], int condition[], int left, int start, int end);  void gotoxy(int x, int y);  int main(void){  int i, start, end, cake\_left = cake\_number, winner;  int cake\_condition[cake\_number] = { 0 }; //과자 상태를 나타내는 배열, 모두 0으로 초기화  char user\_name[2][8]; //사용자 이름 저장  srand((long)time(NULL)); //난수 생성을 위한 시드값  intro\_game();  input\_participant(user\_name);  start = 0;  end = cake\_number - 1;  system("cls");  //user의 값을 2로 하여 과자의 초기상태를 출력  game\_control(user\_name, cake\_condition, &cake\_left, 2, &start, &end);  gotoxy(10, 12);  printf("아무키나 누르면 다음 순서를 진행합니다. ");  \_getch();  do{  for (i = 0; i<2; i++){  system("cls");  //사용자를 구분하기 위해 i 값을 전달(user\_name[2][8])  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); //과자가 1개 남을때 까지 반복  gotoxy(10, 12);  printf("%s님이 이겼습니다. ", user\_name[winner]);  gotoxy(10, 13);  printf("게임을 종료합니다. \n");  return 0;  }  void input\_participant(char user\_name[][8])  {  system("cls");  printf("1번 참가자의 이름을 입력하고Enter>");  scanf\_s("%s", user\_name[0], 8);  printf("2번 참가자의 이름을 입력하고Enter>");  scanf\_s("%s", user\_name[1], 8);  printf("아무키나 누르면 게임을 시작합니다...");  \_getch();  }  void intro\_game(void)  {  system("cls");  printf("주사위로 과자먹기게임\n\n");  printf("두 사람이 서로양끝의 주사위 숫자만큼\n");  printf("과자를 먹는 게임입니다. \n");  printf("마지막 남은과자를 먹는사람이 이깁니다. \n\n");  printf("아무키나 누르면 게임 참가자를\n");  printf("입력합니다.\n");  \_getch();  }  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 과자수: %2d", s);  gotoxy(52, 9);  printf("먹은 과자수: %2d", 29 - e);  }  void game\_control(char name[][8], int condition[], int \*left, int user, int \*s, int \*e){  int i, dice\_number01, dice\_number02, dice\_number = 0;  cake\_display(name, condition, \*left, \*s, \*e);  if (user == 2) //user가 2가 되는 경우는 main으로 복귀  return;  dice\_number01 = rand() % 6 + 1; //첫번째 주사위 난수생성  dice\_number02 = rand() % 6 + 1; //두번째 주사위 난수생성  if(dice\_number01 > dice\_number02)  {  dice\_number = dice\_number01 - dice\_number02;  }  else if(dice\_number01 < dice\_number02)  {  dice\_number = dice\_number02 - dice\_number01;  }    \*left -= dice\_number; //남은 과자 수  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; //생성한 난수 만큼 다음 시작위치를 증가(왼쪽에서 시작)  }  else{  for (i = \*e; i>(\*e - dice\_number); i--)  condition[i] = 1;  \*e -= dice\_number; //생성한 난수 만큼 다음 시작이치를 감소(오늘쪽에서 시작)  }  cake\_display(name, condition, \*left, \*s, \*e);  }  void gotoxy(int x, int y){  COORD Pos = { x - 1, y - 1 };  SetConsoleCursorPosition(GetStdHandle(STD\_OUTPUT\_HANDLE), Pos);  } |
| <결과> |