게임 프로그래밍

c

Game Programming

게임확장모듈

KYUNGSUNG UNIVERSITY SINCE 1955

숫자의 변환과 표현

숫자에 대한 자리 수 분리

숫자를 단위 별로 분리하여 순서대로 출력하는 방법
입력된 금액 또는 숫자가 756826 이라면 순서대로 출력하는 방법에는 다음과 같이 두 가지 방법을 생각할 수 있다.

[방법 1] 높은 단위부터 출력

입력	출력
756896	7 5 6 8 9 6

[방법 2] 낮은 단위부터 출력(역순 출력)

입력	출력
756896	6 9 8 6 5 7

입력될 숫자를 문자열로 처리

입력된 내용을 충분한 크기의 문자형 배열에 저장하여 문자열의 길이를 계산한 다음 순서대로 또는 역순으로 배열요소를 출력.

만약 입력될 숫자 756896을 scanf를 이용하여 문자형 배열 number[]에 저장한다고 가정하면 저장된 각 첨자에 저장된 내용은 다음과 같다.

배열 첨자	number[0]	number[1]	number[2]	number[3]	number[4]	number[5]	number[6]
값	'7'	'5'	'6'	'8'	'9'	'6'	'\0'

높은 단위부터 출력

```
#include <stdio.h>
#include <string.h>
int main(void)
{
  char number[20];
  int length, i;
  printf("금액을 입력하고 Enter>");
  scanf("%s", number);
  length=strlen(number);
  for(i=0;i<length;i++)
     printf("%c₩n", number[i]);
  return 0;
}
```

낮은 단위부터 출력

```
#include <stdio.h>
#include <string.h>
int main(void)
{
  char number[20];
  int length, i;
  printf("금액을 입력하고 Enter>");
  scanf("%s", number);
  length=strlen(number);
  for(i=length;i>=0;i--)
     printf("%c₩n", number[i]);
  return 0;
}
```

입력된 숫자를 정수형으로 처리

문자열은 첨자로 구분하여 문자별로 출력이 가능하지만 숫자는 해당 단위수로 나누어야 하기에 나눗셈과 나머지 연산이 필요.

예를 들어 입력된 정수가 756896이라면 다음과 같이 나누어 가면서 각 자릿수의 숫자를 출력

[방법 1] 높은 단위부터 출력

입력	처리 과정
756896	756896 / 100000 = 7 56896 / 10000 = 5 6896 / 1000 = 6 896 / 100 = 8 96 / 10 = 9 6 / 1 = 6

[방법 2] 낮은 단위부터 출력(역순 출력)

입력	처리 과정
756896	756896 % 10 = 6 75689 % 10 = 9 7568 % 10 = 8 756 % 10 = 6 75 % 10 = 5 7 % 10 = 7

높은 단위부터 출력

높은 단위부터 출력하려면 입력된 숫자가 최대 몇 자리의 정수인지를 계산해야 하는데 이를 계산하는 방법으로 라이브러리 함수 log10을 이용.
몇 자리숫자인지를 계산했다면 위의 [방법 1]과 같이 단위별 숫자로 나누어주는데 이때 x^y를 계산하는 라이브러리 함수 pow(x,y)를 이용

[방법 1] 높은 단위부터 출력

입력	처리 과정
756896	756896 / 100000 = 7 56896 / 10000 = 5 6896 / 1000 = 6 896 / 100 = 8 96 / 10 = 9 6 / 1 = 6

```
void serial_number(long number)
{
   int num;
   int i, length=0;
   length=(int)(log10(number)+1);  //최대 자리수 계산
   for(i=length;i>=1;i--)
   {
     num=number/(long) pow(10, i-1);
     printf("%ld₩n", num);
     number=number-num*(long) pow(10,i-1);
    }
    printf("₩n");
}
```

낮은 단위부터 출력

낮은 단위부터 출력하는 [방법 2]는 나머지 연산자를 이용하여 10으로 나눈 나머지를 이용

[방법 2] 낮은 단위부터 출력(역순 출력)

입력	처리 과정
756896	756896 % 10 = 6 75689 % 10 = 9 7568 % 10 = 8 756 % 10 = 6 75 % 10 = 5 7 % 10 = 7

```
void reverse_number(long number)
{
    while(number>0)
    {
            printf("%ld₩n", number%10);
            number/=10;
    }
}
```

```
#include <stdio.h>
#include <conio.h>
#include <math.h>
void serial_number(long number);
void reverse_number(long number);

int main(void)
{
   long number=12345698;
   printf("입력 숫자 : %ld₩n₩n", number);
   printf("높은 단위부터 출력₩n");
   serial_number(number);
   printf("₩n낮은 단위부터 출력₩n");
   reverse_number(number);
   printf("press any key to continue.....");
   getch();
   return 0;
}
```

```
void serial_number(long number)
{
   int num;
   int i, length=0;
   length=(int)(log10(number)+1);  //최대 자리수 계산
   for(i=length;i>=1;i--)
   {
     num=number/(long) pow(10, i-1);
     printf("%ld₩n", num);
     number=number-num*(long) pow(10,i-1);
    }
    printf("₩n");
}

void reverse_number(long number)
{
   while(number>0)
   {
                printf("%ld₩n", number%10);
                number/=10;
   }
}
```

재귀적 호출을 이용하는 방법

입력된 숫자가 123이라고 가정할 때 높은 단위부터 순서대로 출력하는 방법과 낮은 단위부터 역순으로 출력하는 또 다른 방법은 다음과 같다.

[방법 1] 높은 단위부터 출력

```
(123/100) % 10 →   1
(123/10) % 10  →   2
(123) % 10     →   3
```

[방법 2] 낮은 단위부터 출력(역순 출력)

```
    (123) % 10      →   3
    (123/10) % 10   →   2
    (123/100) % 10  →   1
```

높은 단위부터 출력

[방법 1] :10으로 나누어가는 과정을 재귀적으로 호출함으로써 처리
재귀 함수를 serial이라 가정하면 자기 자신을 호출하면서 printf부분이 보류되고, 반환 과정에서 보류된 부분이 역순으로 처리.
입력된 숫자는 123이고 변수 n에 저장되었다고 가정

[방법 1] 높은 단위부터 출력

```
(123/100) % 10 →    1
(123/10) % 10  →    2
(123) % 10     →    3
```

[호출과정]

단계 1(n=123)	단계 2(n=12)	단계 3(n=1)	단계 4(n=0)

if(123>0)
{
serial(123/10);
printf보류됨
}
else
return;
	if(12>0)
	{
	serial(12/10);
	printf보류됨
	}
	else
	return;
		if(1>0)
		{
		serial(1/10);
		printf보류됨
		}
		else
		return;
			if (0>1)
			(조건이 거짓이므로)
			return;

[반환과정] 각 단계에서 보류되었던 과정이 역순으로 처리됨

단계 7(n=123)	단계 6(n=12)	단계 5(n=1)
printf("%d",123%10); 3을 출력	printf("%d",12%10); 2를 출력	printf("%d",1%10); 1을 출력

```
void serial_number(long number)
{
  if (number>0)
  {
     serial_number(number/10);
     printf("%ld₩n", number%10);
  }
  else
     return;
}
```

[호출과정]

단계 1(n=123)	단계 2(n=12)	단계 3(n=1)	단계 4(n=0)

if(123>0)
{
serial(123/10);
printf보류됨
}
else
return;
	if(12>0)
	{
	serial(12/10);
	printf보류됨
	}
	else
	return;
		if(1>0)
		{
		serial(1/10);
		printf보류됨
		}
		else
		return;
			if (0>1)
			(조건이 거짓이므로)
			return;

[반환과정] 각 단계에서 보류되었던 과정이 역순으로 처리됨

단계 7(n=123)	단계 6(n=12)	단계 5(n=1)
printf("%d",123%10); 3을 출력	printf("%d",12%10); 2를 출력	printf("%d",1%10); 1을 출력

낮은 단위부터 출력

[방법 2] 낮은 단위부터 출력(역순 출력)

```
(123) % 10      →   3
(123/10) % 10   →   2
(123/100) % 10  →   1
```

```
void reverse_number(long number)
{
        printf("%ld₩n", number%10);
        if((number/10)>0)
                reverse_number(number/10);
        else
                return;
}
```

단계 1(n=123)	단계 2(n=12)	단계 3(n=1)

printf("%d\n", 123%10);
if((123/10)>0)
reverse_number(123/10);
	printf("%d\n", 12%10);
	if((12/10)>0)
	reverse_number(12/10);
		printf("%d\n", 1%10);
		if((1/10)>0)
		(조건이 거짓이므로)
		return;

재귀적 호출

```
#include <stdio.h>
#include <conio.h>
#include <math.h>
void serial_number(long number);
void reverse_number(long number);

int main(void)
{
   long number=12345698;
   printf("입력 숫자 : %ld₩n₩n", number);
   printf("높은 단위부터 출력₩n");
   serial_number(number);
   printf("₩n낮은 단위부터 출력₩n");
   reverse_number(number);
   printf("press any key to continue.....");
   getch();
   return 0;
}
```

```
void serial_number(long number)
{
  if (number>0)
  {
     serial_number(number/10);
     printf("%ld₩n", number%10);
  }
  else
     return;
}

void reverse_number(long number)
{
   printf("%ld₩n", number%10);
   if((number/10)>0)
            reverse_number(number/10);
   else
            return;
}
```

디지털 숫자

입력된 숫자가 정수 123이라고 가정한다면 아래와 같은 형식으로 다섯 줄을 출력해야 합니다.
다음 표에서 각 배열의 값이 1인 경우는 기호 ■을 출력하고 값이 0인 경우에는 두 자리의 공백을 출력

해당 숫자	1	2	3
출력될 첫 번째 줄	■	■■■■	■■■■
출력될 두 번째 줄	■	■	■
출력될 세 번째 줄	■	■■■■	■■■■
출력될 네 번째 줄	■	■	■
출력될 다섯 번째 줄	■	■■■■	■■■■

정수 0~9까지의 디지털 숫자는 1차원 배열에 저장
0~9까지의 디지털 숫자를 배열 one[]~nine[]까지 저장한다고 가정

숫자	0	1	2	…	8	9
배열	int zero[20]= {1,1,1,1, 1,0,0,1, 1,0,0,1, 1,0,0,1, 1,1,1,1};	int one[20]= {0,0,1,0, 0,0,1,0, 0,0,1,0, 0,0,1,0, 0,0,1,0};	int two[20]= {1,1,1,1, 0,0,0,1, 1,1,1,1, 1,0,0,0, 1,1,1,1};	생략	int eight[20]= {1,1,1,1, 1,0,0,1, 1,1,1,1, 1,0,0,1, 1,1,1,1};	int nine[20]= {1,1,1,1, 1,0,0,1, 1,1,1,1, 0,0,0,1, 0,0,0,1};

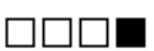
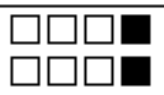
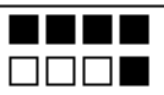
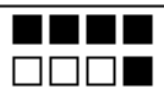
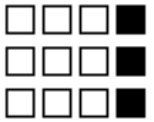
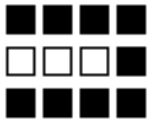
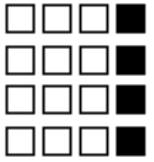
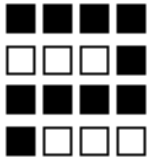
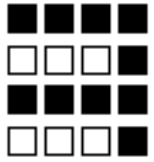
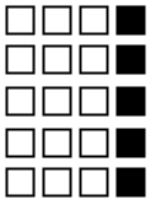
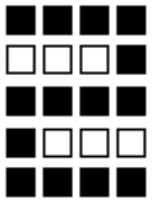
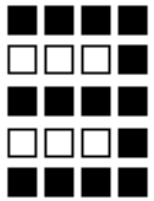
출력할 line	1	2	3
	출력할 배열 요소	출력할 배열 요소	출력할 배열 요소
첫 번째	one[0]에서 one[3]	two[0]에서 two[3]	three[0]에서 three[3]
두 번째	one[4]에서 one[7]	two[4]에서 two[7]	three[4]에서 three[7]
세 번째	one[8]에서 one[11]	two[8]에서 two[11]	three[8]에서 three[11]
네 번째	one[12]에서 one[15]	two[12]에서 two[15]	three[12]에서 three[15]
다섯 번째	one[16]에서 one[19]	two[16]에서 two[19]	three[16]에서 three[19]

입력된 숫자에 대해 앞의 표와 같이 연속적으로 line을 분리해 가면서 출력해야 하는데, 출력할 line 마다 각 자리수의 숫자를 차례로 출력해야 하므로 재귀적 호출 방법을 이용

프로그램을 구현하기 위해 작성할 함수들

함수	처리할 내용
number_check	입력된 숫자에 해당하는 배열로 구분하여 호출
digit_print	배열의 이름을 인자로 하여 디지털 숫자를 출력

정수 123에 대한 디지털 숫자의 처리과정은 함수 main에서 입력된 숫자에 대해 line을 인자로 하여 함수 number_check을 호출하는데 number_check은 재귀적 호출에 의해 입력된 숫자에 대해 차례대로 해당 line에 대한 디지털 숫자를 출력

단계	호출	출력 line	화면에 출력되는 과정		
			1	2	3
단계 1	number_check(123, 0)	1			
단계 2	number_check(123, 1)	2			
단계 3	number_check(123, 2)	3			
단계 4	number_check(123, 3)	4			
단계 5	number_check(123, 4)	5			

```
#include <stdio.h>
void number_check(int k, int i);
void digit_print(int dim[], int line);

int zero[20] =
            {1,1,1,1,
             1,0,0,1,
             1,0,0,1,
             1,0,0,1,
             1,1,1,1};

int one[20]=
 {0,0,1,0,
  0,0,1,0,
  0,0,1,0,
  0,0,1,0,
  0,0,1,0};

int two[20]=
 {1,1,1,1,
  0,0,0,1,
  1,1,1,1,
  1,0,0,0,
  1,1,1,1};
```

```
int three[20]=
 {1,1,1,1,
  0,0,0,1,
  1,1,1,1,
  0,0,0,1,
  1,1,1,1};

int four[20]=
 {1,0,0,1,
  1,0,0,1,
  1,1,1,1,
  0,0,0,1,
  0,0,0,1};

int five[20]=
 {1,1,1,1,
  1,0,0,0,
  1,1,1,1,
  0,0,0,1,
  1,1,1,1};
```

```
int six[20]=
 {1,0,0,0,
  1,0,0,0,
  1,1,1,1,
  1,0,0,1,
  1,1,1,1};

int seven[20]=
 {1,1,1,1,
  0,0,0,1,
  0,0,0,1,
  0,0,0,1,
  0,0,0,1};

int eight[20]=
 {1,1,1,1,
  1,0,0,1,
  1,1,1,1,
  1,0,0,1,
  1,1,1,1};
```

```
int nine[20] =
            {1,1,1,1,
             1,0,0,1,
             1,1,1,1,
             0,0,0,1,
             0,0,0,1
            };
```

```
int main(void)
{
        int num, line;
        printf("디지털 숫자 출력 프로그램₩n");
        printf("1 이상의 정수만 입력합니다. ₩n₩n");
        printf("₩n정수 숫자입력 후 Enter> ");
        scanf("%d", &num);
        printf("₩n₩n");
        for(line=0;line<=4;line++)
        {
                number_check(num, line);
                printf("₩n");
        }
        return 0;
}
```

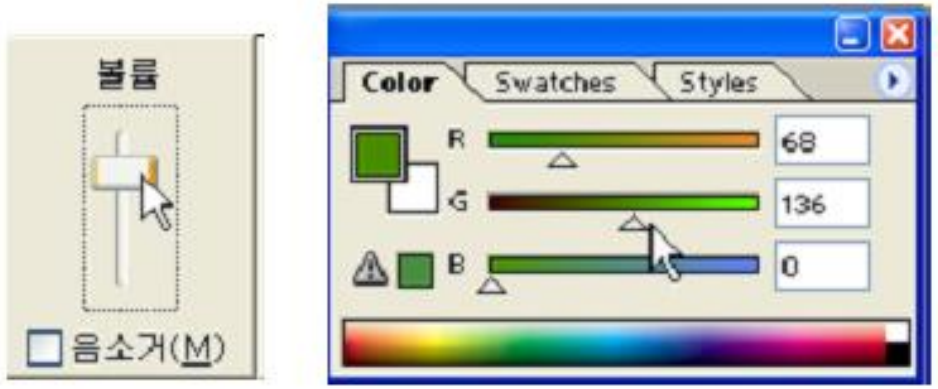
```
void digit_print(int dim[], int line)
{
 int i;
 for(i=line*4;i<=line*4+3; i++)
  if (dim[i]==1)
                printf("■");
  else
                printf("   ");
 printf("   ");
}
```

```
void number_check(int k, int i)
{
 if (k>=1)
  {
     number_check(k/10, i);
     switch(k%10)
     {
        case 0 :
                digit_print(zero, i);
                break;
        case 1 :
                digit_print(one, i);
                break;
        case 2 :
                digit_print(two, i);
                break;
        case 3 :
                digit_print(three, i);
                break;
```

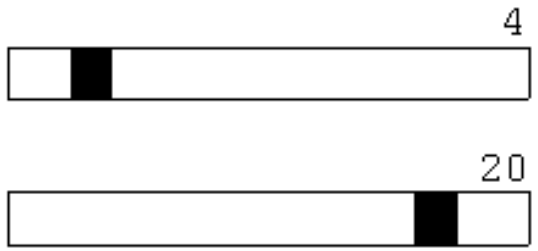
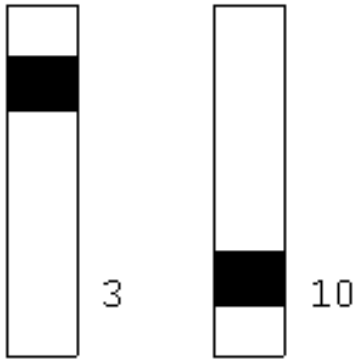
```
case 4 :
                digit_print(four, i);
                break;
        case 5 :
                digit_print(five, i);
                break;
        case 6 :
                digit_print(six, i);
                break;
        case 7 :
                digit_print(seven, i);
                break;
        case 8 :
                digit_print(eight, i);
                break;
        case 9 :
          digit_print(nine, i);
                break;
     }
  }
}
```

수평 수직 슬라이드 바

슬라이드 바(slide bar)는 아래의 그림과 같이 마우스를 이용한 제어에 유용하게 사용되는 인터페이스 도구



긴 막대부분(bar)의 표시는 사각형을 그리는 함수 draw_rectangle을 이용
화살표 키의 이동은 함수 move_arrow_key를, 현재의 크기를 나타낼 검은 사각형 부분(슬라이드, slide)은 확장 아스키코드에서 기호 ■을 사용
적당한 위치에 bar가 움직인 크기를 숫자로 나타낸다.

구분	수평 슬라이드 바	수직 슬라이드 바
사용키	화살표 키 (←, →)	화살표 키 (↑, ↓)
표시형식		

좌우 방향의 화살표 키는 수평 슬라이드 바에서의 증가(→)와 감소(←)를 제어하고, 상하 방향의 화살표 키는 수직 슬라이드 바에서 증가(↑)와 감소(↓)를 제어한다고 가정
수평과 수직 슬라이드 바의 크기는 입력 값에 의해 결정하고, 바의 표현은 직사각형을 그리는 함수인 draw_rectangle을 이용
만약 수평과 수직 슬라이드 바의 길이가 각각 10과 15라 한다면 직사각형 그리기 함수 draw_rectangle은 다음과 같이 사용

길이가 10인 수평 슬라이드 바의 표시	`draw_rectangle(10, 1);`
길이가 15인 수직 슬라이드 바의 표시	`draw_rectangle(1, 15);`

함수	처리할 내용
draw_vertical_slide	수직 슬라이드바 표시
draw_horizontal_slide	수평 슬라이드바 표시
move_arrow_key	화살표 키의 제어
draw_rectangle	직사각형 그리기
gotoxy	커서의 위치 제어

사용자 정의 함수

함수 main

```
int main(void)
{
  변수 선언과 초기화
  수평 슬라이드바의 길이 입력
  수직 슬라이드바의 길이 입력
  do
  {
     draw_vertical_slide(…);
     draw_horizontal_slide(…);
     key=getch();
     move_arrow_key(…);
  }while(key!=27);
  return 0;
}
```

함수 draw_vertical_slide

```
void draw_horizontal_slide(…)
{
  커서 이동
  draw_rectangle(…);
  커서 이동
  바의 움직임을 표시
}
```

함수 draw_horizontal_slide

```
void draw_horizontal_slide(…)
{
  커서 이동
  draw_rectangle(…);
  커서 이동
  바의 움직임을 표시
}
```

슬라이드바를 제어하고 출력하기 위한 호출 함수들

함수에서 매개변수 char *s는 슬라이드 바의 움직임을 표시할 문자열을 의미.
이 예에서는 특수기호 ■(완성형)을 이용

수평 슬라이드바를 표시하는 함수

```
void draw_horizontal_slide(int x, int y, int length, char *s)
{
        int real_length=length/2;
        gotoxy(1, y);
        draw_rectangle(real_length+1, 1);
        gotoxy(x+2, y+1);
        printf("%s", s);
        gotoxy(real_length*2+2, y-1);
        printf("%2d", x);
}
```

함수에서 매개변수 char *s는 슬라이드 바의 움직임을 표시할 문자열을 의미.
이 예에서는 특수기호 ■(완성형)을 이용

수직 슬라이드바를 표시하는 함수

```
void draw_vertical_slide(int x, int y, int length, char *s)
{
        gotoxy(x, 1);
        draw_rectangle(1, length);
        gotoxy(x+2, y+1);
        printf("%s", s);
        gotoxy(x+6, length+1);
        printf("%2d", y);
}
```

```
#include <stdio.h>
#include <stdlib.h>
#include <conio.h>
#include <windows.h>
void draw_horizontal_slide(int x, int y, int length, char *s);
void draw_vertical_slide(int x, int y, int length, char *s);
void draw_rectangle(int c, int r);
void move_arrow_key(char key, int *x1, int *y1, int x_b, int y_b);
void gotoxy(int x, int y);

int main(void)
{
   char *slide="■", key;
   int x=1, y=1;
   int h_slide_length, v_slide_length;
   printf("슬라이드바 표시₩n₩n");
   printf("수평 슬라이드바의 길이(최대 70)를 ₩n");
   printf("입력하고 Enter>");
   scanf("%d", &h_slide_length);
   printf("수직 슬라이드바의 길이(최대 19)를 ₩n");
   printf("입력하고 Enter>");
   scanf("%d", &v_slide_length);
   system("cls");
```

```
  do
    {
     draw_vertical_slide(1, y, v_slide_length, slide);
     draw_horizontal_slide(x,v_slide_length+3,h_slide_length,slide);
     key=getch();
     move_arrow_key(key, &x, &y, h_slide_length, v_slide_length);
    }while(key!=27);
    return 0;
}
```

```
void draw_rectangle(int c, int r)
{
    int i, j;
    unsigned char a=0xa6;
    unsigned char b[7];
    for(i=1;i<7;i++)
            b[i]=0xa0+i;

    printf("%c%c",a, b[3]);
    for(i=0;i<c;i++)
            printf("%c%c", a, b[1]);
    printf("%c%c", a, b[4]);
    printf("₩n");
```

```
void draw_horizontal_slide(int x, int y, int length, char *s)
{
            int real_length=length/2;
            gotoxy(1, y);
            draw_rectangle(real_length+1, 1);
            gotoxy(x+2, y+1);
            printf("%s", s);
            gotoxy(real_length*2+2, y-1);
            printf("%2d", x);
}
```

```
 for(i=0;i<r;i++)
    {
            printf("%c%c", a, b[2]);
            for(j=0;j<c;j++)
                    printf("  ");
            printf("%c%c",a, b[2]);
            printf("₩n");
    }
    printf("%c%c", a, b[6]);
    for(i=0;i<c;i++)
            printf("%c%c", a, b[1]);
    printf("%c%c", a, b[5]);
    printf("₩n");
}
```

```
void draw_vertical_slide(int x, int y, int length, char *s)
{
        gotoxy(x, 1);
        draw_rectangle(1, length);
        gotoxy(x+2, y+1);
        printf("%s", s);
        gotoxy(x+6, length+1);
        printf("%2d", y);
}
```

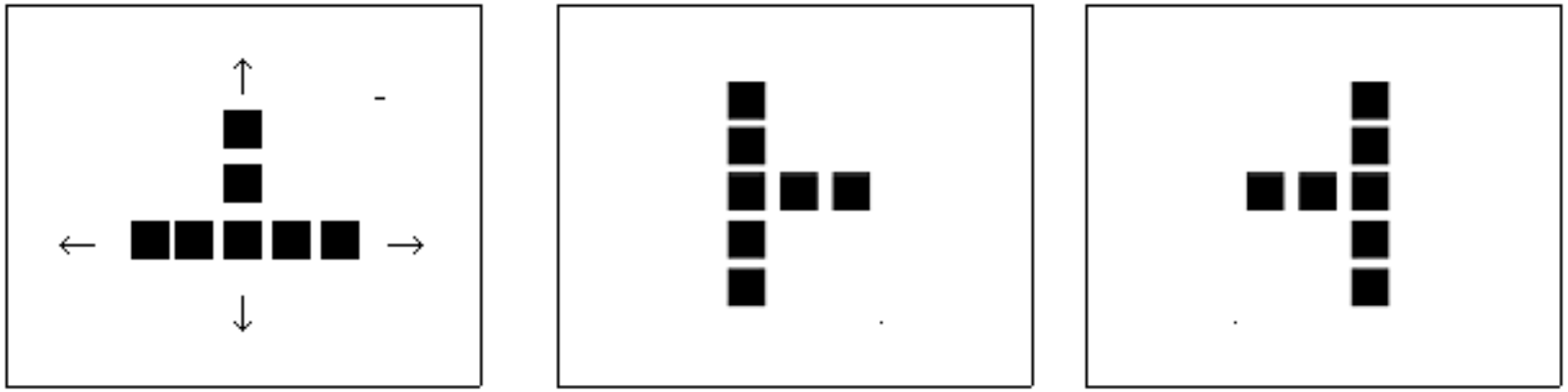
```
void gotoxy(int x, int y)
{
   COORD Pos = {x - 1, y - 1};
   SetConsoleCursorPosition(GetStdHandle(STD_OUTPUT_HANDLE), Pos);
}
```

```
void move_arrow_key(char key, int *x1, int *y1, int x_b, int y_b)
{
        switch(key)
        {
        case 72:  //위쪽(상) 방향의 화살표 키 입력
                *y1=*y1-1;
                if (*y1<1) *y1=1; //y좌표의 최소값
                break;
        case 75:  //왼쪽(좌) 방향의 화살표 키 입력
                *x1=*x1-1;
                if (*x1<1) *x1=1; //x좌표의 최소값
                break;
        case 77:  //오른쪽(우) 방향의 화살표 키 입력
                *x1=*x1+1;
                if (*x1>x_b)   *x1=x_b; //x좌표의 최대값
                break;
        case 80:  //아래쪽(하) 방향의 화살표 키 입력
                *y1=*y1+1;
                if (*y1>y_b)   *y1=y_b; //y좌표의 최대값
                break;
        default:
                return;
        }
}
```

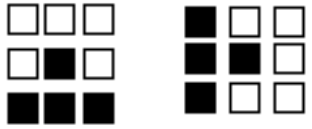
도형의 연속적인 이동과 회전

화면에 표시된 임의의 도형을 화살표(방향) 키를 이용하여 연속적으로 이동시키거나 시계방향 또는 반 시계방향으로 회전시키는 프로그램

예를 들어 화면에 아래와 같이 도형을 표시하였다면 화살표(방향) 키를 눌렀을 때 상하좌우로 도형을 이동시키고, 스페이스(space) 키를 눌렀을 경우에는 시계방향으로 회전시키거나, 또는 반 시계방향으로 회전하도록 표시



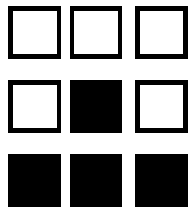
그래픽에 있어서 도형의 회전은 x, y, z 축의 좌표를 중심으로 각 점의 좌표를 주어진 각도만큼 회전시키데 있어서 회전 행렬을 이용하지만 여기에서는 도형 자체의 모양을 변화시킴으로서 회전된 효과를 나타내도록 한다.

도형	행렬
 [도형 1] [도형 2]	$\begin{bmatrix}0&0&0\\0&1&0\\1&1&1\end{bmatrix}$ $\begin{bmatrix}1&0&0\\1&1&0\\1&0&0\end{bmatrix}$ [행렬 1] [행렬 2]

[도형 2]는 [도형 1]을 시계방향으로 90도 회전시킨 결과와 같고 이는 개념적으로 [행렬 1]을 [행렬 2]로 변환시킨 것과 같다. 따라서 시계방향으로 90도 회전은 [행렬 1]을 [행렬 2]로 변환시킨 다음 값 1은 ■로, 값 0은 □(또는 공백)로 출력

도형의 출력

3×3 행렬을 이용하여 출력



도형의 표현	도형의 출력

int m[3][3]=
{0,1,0,
0,1,0,
1,1,1};
	for(i=0;i<3;i++)
	{
	for(j=0;j<3;j++)
	if (m[i][j]==1)
	printf("■");
	else
	printf(" ");
	printf("\n");
	}

도형의 회전

예를 들어 3×3 행렬로 가정할 경우 [행렬 a]를 중앙의 ●을 축의 중심으로 회전시켜 [행렬 b]로 만든다.

이는 행렬의 첨자의 위치를 변환시키는 것으로 일반화를 하면 다음과 같다

<table>
<tr><th colspan="2">도형의 회전</th><th>행렬 첨자의 변환 과정</th><th>일반화</th></tr>
<tr>
<td>1 2 3
0 ● 0
0 0 0
[행렬 a]</td>
<td>0 0 1
0 ● 2
0 0 3
[행렬 b]</td>
<td>a[i][j] b[j][2-i]
a[0][0] → b[0][2]
a[0][1] → b[1][2]
a[0][2] → b[2][2]</td>
<td><pre>for(i=0;i<3;i++)
    for(j=0;j<3;j++)
        b[j][2-i]=a[i][j];</pre></td>
</tr>
</table>

상하좌우로의 이동은 화살표(방향) 키를 이용하고, 회전은 스페이스(공백) 키를 눌렀을 때 시계방향 또는 반 시계방향으로 90도 회전

정적인 움직임

```
switch(key)
{
   case 32 :
        rotation_right(matrix);
        break;
   case 72 :
        y--;
        break;
   case 75 :
        x--;
        break;
   case 77 :
        x++;
        break;
   case 80 :
        y++;
        break;
   default :
        break;
}
move_shape(matrix);
```

동적인 움직임

```
switch(key)
{
   case 32 :
        rotation_right(matrix);
        move_shape(matrix);
        break;
   case 72 :
        inx=0; iny=-1;
        move_shape(matrix);
        break;
   case 75 :
        inx=-1; iny=0;
        move_shape(matrix);
        break;
   case 77 :
        inx=1; iny=0;
        move_shape(matrix);
        break;
   case 80 :
        inx=0; iny=1;
        move_shape(matrix);
        break;
   default :
        break;
}
```

함수	함수 설명
rotation_right	도형의 회전
move_shape	도형의 이동
print_shape	도형의 출력
move_control	화살표 키에 따른 제어
gotoxy	커서의 위치 제어
print_direction	지시문의 출력

함수 main

```
int x=35, y=12;
int inx=0, iny=0;
int main(void)
{
   int shape1[3][3]= {0,1,0,
                      0,1,0,
                      1,1,1};
    move_control(shape1);
    return 0;
}
```

함수 move_shape

```
void move_shape(int m[][3])
{
   do
   {
        system("cls");
        print_shape(m);
        print_direction();
        이동할 x좌표 값
        이동할 y좌표 값
        Sleep(100);
   }while(!kbhit());
}
```

함수 move_control

```
void move_control(int m[][3])
{
   char key;
   do
   {
        while(!kbhit())
        {
           system("cls");
           move_shape(m);
        }
        key=getch();
                  동적인 움직임 참고
        }while(key!=27);
        printf("\n");
}
```

함수 print_shape

```
void print_shape(int m[][3])
{
   배열에 저장된 형태로
   도형 출력
}
```

```
#include <stdio.h>
#include <stdlib.h>
#include <conio.h>
#include <windows.h>

void rotation_right(int m[][3]);
void move_shape(int m[][3]);
void print_shape(int m[][3]);
void move_control(int m[][3]);
void gotoxy(int x, int y);
void print_direction(void);

int x=35, y=12;
int inx=0, iny=0;
```

```
void rotation_right(int m[][3])
{
        int i, j;
        int temp[3][3];
        for(i=0;i<3;i++)
                for(j=0;j<3;j++)
                        temp[j][2-i]=m[i][j];
        for(i=0;i<3;i++)
                for(j=0;j<3;j++)
                        m[i][j]=temp[i][j];
}
```

```
int main(void)
{
        int shape1[3][3]= {0,1,0,
                           0,1,0,
                           1,1,1};
        move_control(shape1);
        return 0;
}
```

```
void print_direction(void)
{
        gotoxy(25, 1);
        printf("화살표:이동,  스페이스 키:회전");
}
```

```
void move_control(int m[][3])
{
        char key;
        do
        {
                while(!kbhit())
                {
                        system("cls");
                        move_shape(m);
                }
                key=getch();
                switch(key)
                {
                        case 32 :
                                rotation_right(m);
                                move_shape(m);

                                break;
                        case 72 :
                                inx=0; iny=-1;
                                move_shape(m);

                                break;
                        case 75 :
                                inx=-1; iny=0;
                                move_shape(m);

                                break;
                        case 77 :
                                inx=1; iny=0;
                                move_shape(m);

                                break;
                        case 80 :
                                inx=0; iny=1;
                                move_shape(m);

                                break;
                        default :
                                break;
                }
        }while(key!=27);
        printf("₩n");
}
```

```
void print_shape(int m[][3])
{
        int i, j;
        for(i=0;i<3;i++)
        {
                gotoxy(x,y+i);
                for(j=0;j<3;j++)
                        if (m[i][j]==1)
                                printf("□");
                        else
                                printf("  ");
                printf("₩n");
        }
}
```

```
void gotoxy(int x, int y)
{
   COORD Pos = {x - 1, y - 1};
   SetConsoleCursorPosition(GetStdHandle(STD_OUTPUT_HANDLE), Pos);
}
```

```
void move_shape(int m[][3])
{
        do
        {
                system("cls");
                print_shape(m);
                print_direction();
                x=x+inx;
                y=y+iny;
                if (y>23)
                        y=23;
                else if (y<2)
                        y=2;
                if (x>75)
                        x=75;
                else if (x<1)
                        x=1;
                Sleep(100);
        }while(!kbhit());
}
```

Reference

- ✓ 명품 C언어 프로젝트, 생능출판, 안기수