

16. 함수와 변수

16-1. 변수의 속성

● 변수의 속성

- 변수가 선언되는 위치에 따라 변수의 범위, 생존시간, 변수의 연결 등이 달라진다.

- 1) 변수의 범위
- 2) 변수의 생존시간
- 3) 변수의 연결

16-2) 변수의 범위

- 변수는 선언되는 위치에 의하여 사용하는 범위가 결정
 - 지역변수 : 함수 또는 블록 안에서 정의되는 변수, 지역 변수는 해당 블록이나 함수 안에서만 사용이 가능
 - 전역변수 : 함수의 외부에서 선언되는 변수, 전역변수는 소스 파일의 어느 곳에서도 사용이 가능하다.

16-2) 변수의 범위

- 지역 변수의 선언 위치

```
int Add(void) {  
    int x ; // 올바른 선언  
    . . . . .  
    x =100;  
  
    int y ; //잘못된 선언  
}
```

16-2) 변수의 범위 (예제 16-1)

● 지역 변수의 접근 범위

- 지역변수는 선언된 블록에서만 접근이 가능함.

```
#include <stdio.h>
int main(void)
{
    int x, y;
    printf("두 수를 입력하시오 :");
    scanf("%d %d",&x,&y);

    if(x > y )
    {
        int diff ;
        diff = x-y ;
        printf(" x가 y 보다 %d 만큼 크다\n",diff);

    }

    // printf("diff  =%d \n",diff);
    printf("프로그램 종료 !!\n");
    return 0;
}
```

16-2) 변수의 범위 (예제 16-2)

```
#include <stdio.h>

int main(coid)
{
    int i ;

    for(i = 0; i <5 ; i++ )
    {
        int cnt = 1;
        printf("cnt = %d\n",cnt);
        cnt ++ ;
    }
    return 0;
}
```

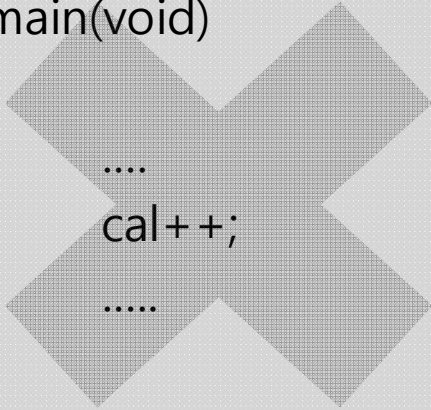
16-2) 변수의 범위

● 지역 변수의 접근 범위

- 지역변수는 선언된 함수에서만 접근이 가능함.
- 지역변수는 이름이 같아도 블록이나 함수가 다르면 다른 변수이다.

```
int ADD( int x , int y)
{
    int cal = 0;
    .....
    return sum;
}
int SUB(int x, int y)
{
    int cal = 0;
    .....
    return 0;
}
```

```
int main(void)
{
    ....
    cal++;
    ....
}
```



16-2) 변수의 범위 (예제 16-3)

● 지역 변수의 초기값

- 지역변수는 컴파일러에 의해서 '0'으로 초기화되지 않는다.
- 초기값을 지정 하지 않으면 아무 의미 없는 쓰레기 값이 들어간다.

```
#include <stdio.h>

int main(void)
{
    int tmp;

    printf("tmp = %d\n",tmp);
    return 0;
}
```


16-2) 변수의 범위 (예제 16-4)

● 함수의 매개변수

- 함수의 매개 변수도 일종의 지역변수이다. 지역변수가 지니는 모든 특징을 가지고 있다.
- 다른 지역변수와 다른 점은 함수 호출시의 인수 값으로 초기화 된다는 점이다.

```
#include <stdio.h>
int dec( int );

int main(void)
{
    int h = 11;

    printf("함수 호출전 h = %d\\n",h);
    dec(h);
    printf("함수 호출 후 h = %d\\n",h);

    return 0;
}
```

```
int dec(int cnt)
{
    cnt-- ;
    return cnt ;
}
```

16-2) 변수의 범위

● 전역변수

- 전역변수 : 함수의 외부에서 선언되는 변수, 전역변수는 소스 파일의 어느 곳에서도 사용이 가능하다

전역변수는 초보자들이 생각하기에 상당히 편리할 것 같지만 전문가들은 사용을 회피한다. 왜냐하면 어디서나 접근이 가능한 변수이기 때문에 프로그램이 복잡해지면 어느 부분에서 전역변수를 접근하고 있는지 찾기 힘들기 때문이다.

16-2) 변수의 범위 (예제 16-5)

```
#include <stdio.h>
```

```
int x = 1234;
```

```
void func1(void);
```

```
void func2(void);
```

```
int main(void)
```

```
{
```

```
    func1();
```

```
    func2();
```

```
    printf("main() :x = %d\n",x);
```

```
    return 0;
```

```
}
```

16-2) 변수의 범위 (예제 16-5)

```
void func1(void)
{
    printf("func1() : x = %d\n",x);
}
```

```
void func2(void)
{
    printf("func2() : x = %d\n",x);
}
```

16-2) 변수의 범위 (예제 16-6)

● 전역변수의 또 다른 특징

- 같은 이름의 지역변수에 의해 가려진다.

```
#include <stdio.h>

int x= 1;

void func1(void);

int main(void)
{
    int x = -10 ;

    printf("x = %d\n",x);
    func1();
    printf("x = %d\n",x);
}

void func1(void)
{
    printf("x = %d\n",x);
}
```

16-3) 변수의 생존시간

- 정적 할당 : 프로그램이 실행되는 동안에는 계속해서 변수에 저장 공간이 할당되어 있는 방법
- 자동 할당 : 블록(또는 함수)이 시작되면서 변수에 저장 공간이 할당되고 블록(또는 함수)이 종료되면 저장 공간이 회수되는 방법.

1) 변수가 선언되는 위치 :

전역변수는 정적 할당, 지역 변수는 자동 할당,

단, 지역 변수 앞에 저장 유형 지정자 (static)를 붙이면

정적 할당으로 변경.

2) 저장 유형 지정자 : auto, register, static, extern

16-3) 변수의 생존시간 (예제 16-7)

- auto 지정자 : 함수나 블록 내에 선언되는 지역변수는 기본적으로 자동 할당한다. 이러한 지역 변수를 자동변수라 한다. auto를 생략할 수 있다. 자동 변수는 자신이 선언된 블록에서 사용이 끝나면 자동으로 메모리에서 제거되므로 메모리를 효율적으로 사용하게 된다.

```
#include <stdio.h>
int main(void)
{
    auto int i, sum = 0;

    for( i = 1; i <=10 ;i++ )
        sum = sum + i;

    printf("sum = %d\n",sum);
}
```

16-3) 변수의 생존시간

- static 지정자 : 지역(자동) 변수는 블록에서만 사용된다. 블록을 벗어나도 제거되지 않는 변수를 만들기 위해서는 지역변수를 정적 변수로 만들어 주어야 한다.
정적 변수는 지역변수 앞에 static를 붙여준다.
- 정적 변수는 전역 변수와 같이 프로그램이 시작할 때 메모리에 생성되고 프로그램이 실행을 종료하면 메모리에서 제거 된다 .

16-3) 변수의 생존시간 (예제 16-8)

```
#include <stdio.h>

void func(void);
int main(void)
{
    int i;

    for(i = 0 ; i < 4 ; i ++ )
        func();

    return 0 ;
}
```

16-3) 변수의 생존시간 (예제 16-8)

```
void func(void)
{
    int auto_cnt = 0;
    static int static_cnt = 0;

    auto_cnt++;
    static_cnt++;

    printf("auto_cnt = %d\n",auto_cnt);
    printf("static_cnt = %d\n",static_cnt);
}
```

16-3) 변수의 생존시간 (예제 16-9)

- register 지정자 : 변수를 레지스터(register) 변수로 만든다. 레지스터 변수는 변수를 메모리에 저장하는 것이 아니라 CPU내의 레지스터에 저장하기 때문에 읽고 쓰는 속도가 아주 빠르다. 레지스터 변수는 지역 변수만 가능하다.

```
# include <stdio.h>
int main(void)
{
    register int i, sum = 0;
    int num ;

    printf("수를 입력하시오 :");
    scanf("%d",&num);

    for( i = 1; i <=num ;i++ )
        sum = sum + i;

    printf("sum = %d\n",sum);
}
```

16-4) 변수의 연결

- extern 지정자 : 전역변수를 다른 파일에서 사용하기 위해서는 extern을 붙인다.

```
// link1.c
# include <stdio.h>
int xy ;
extern void Add(void);

int man(void)
{
    Add();
    printf("%d\n", xy);
    return 0;
}
```

```
//link2.c

extern int xy ;
void Add(void)
{
    xy = xy + 10;
}
```

16-5) 순환함수 (예제 16-10)

- 함수는 자기 자신을 호출 할 수 있다.
이것을 순환 함수라고 부른다.

```
#include <stdio.h>

long factorial (int) ;
int main(void)
{
    int num ;
    long int result ;

    printf("수를 입력하시오 :");
    scanf("%d",&num);
    result = factorial(num);
    printf("result = %d\n",result);

    return 0;
}
```

16-5) 순환함수 (예제 16-10)

```
long factorial(int n)
{
    printf("factorial (%d)\n",n);

    if( n <=1) return 1;
    else return n* factorial(n-1);
}
```

16. 연습문제

1. 다음과 같이 10진수를 2진으로 표시하는 프로그램(함수)을 작성하시오
void print_10to2(int x) // 10진수를 2진수로 출력하는 함수

프로그램 예)

```
#include <stdio.h>
```

```
void print_10to2(int);
```

```
int main(void)
```

```
{
```

```
    int num ;
```

```
    printf("수를 입력하시오: ");
```

```
    scanf("%d",&num);
```

```
    print_10to2(num); // 이 함수를 작성 하시오
```

```
    return 0;
```

```
}
```

16-5) 확장 예제 (예제 16-11)

- 날짜와 시간을 문자열로 출력

```
#include <time.h>
#include <stdio.h>

int main(void)
{
    time_t now ;
    time(&now);

    printf("현재 날짜와 시간:%s",asctime(localtime(&now)));
    printf("현재 날짜와 시간 :%s", ctime(&now));
    return 0;
}
```


16-5) 확장 예제 (예제 16-12)

● kbhit() 함수 예제

```
#include <stdio.h>

int main(void)
{
    int i = 0;
    char key = 0;

    for( ; ; ) {
        printf("i = %d\n",i++);

        key = kbhit();
        if(key==1) break;
    }

    printf("key = %d\n",key);
    return 0;
}
```

16-5) 확장 예제 (예제 16-13)

● gotoxy(int x, int y) 함수 예제

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

void gotoxy(int, int);

int main(void)
{
    gotoxy(1,1); printf("0");

    gotoxy(80,1); printf("1");

    gotoxy(1,24); printf("2");

    gotoxy(80,24) ; printf("3");
    return 0 ;
}
```

16-5) 확장 예제 (예제 16-13)

- gotoxy(int x, int y) 함수 예제

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

16-5) 확장 예제 (예제 16-14)

- 첫 번째 바이트가 0xa1으로 시작하는 완성형 특수기호 코드의 출력

```
#include <stdio.h>

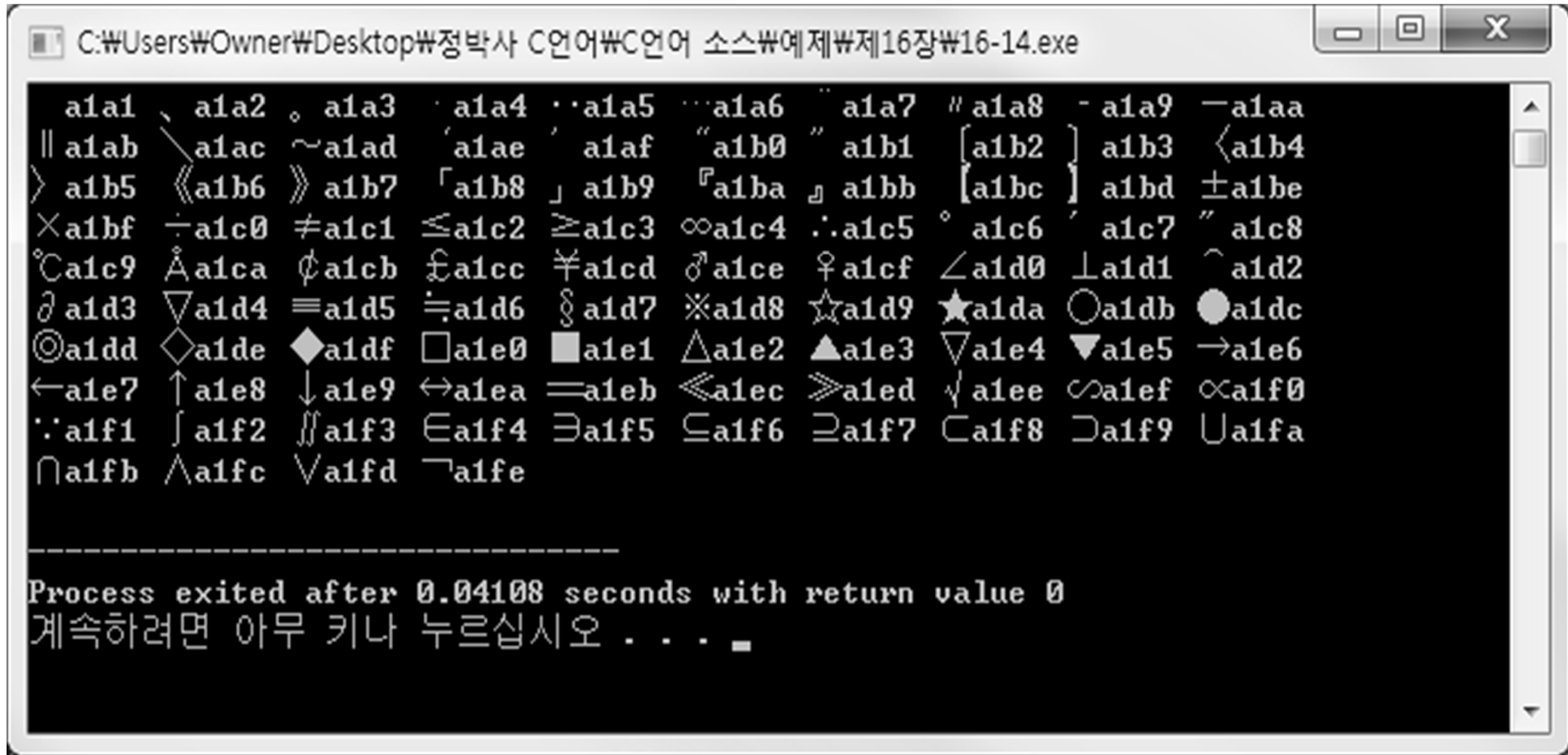
int main(void)
{
    unsigned char b1 = 0xa1;
    unsigned char b2 ;

    for( b2 = 0xa1 ; b2 <=0xfe ;b2++ )
    {
        printf("%c%c%2x%2x ",b1,b2,b1,b2);
        if(b2 % 10 ==0) printf("\n");
    }
    printf("\n");

    return 0;
}
```

16-5) 확장 예제 (예제 16-14)

- 첫 번째 바이트가 0xa1으로 시작하는 완성형 특수기호 코드의 출력



```
C:\Users\Owner\Desktop\정박사 C언어\C언어 소스\예제\제16장\16-14.exe

a1a1 \ a1a2 ° a1a3 · a1a4 ∙ a1a5 ∙ a1a6 " a1a7 " a1a8 - a1a9 -a1aa
|| a1ab \ a1ac ~ a1ad ' a1ae ' a1af " a1b0 " a1b1 [a1b2 ] a1b3 <a1b4
> a1b5 《a1b6 》 a1b7 「a1b8 」 a1b9 『a1ba 』 a1bb [a1bc ] a1bd ±a1be
×a1bf ÷a1c0 ≠a1c1 ≤a1c2 ≥a1c3 ∞a1c4 ∴a1c5 ° a1c6 ' a1c7 " a1c8
°Ca1c9 Åa1ca ¢a1cb £a1cc ¥a1cd ♂a1ce ♀a1cf ∠a1d0 ⊥a1d1 ^ a1d2
∂ a1d3 ∇a1d4 ≡a1d5 ≡a1d6 § a1d7 ※a1d8 ☆a1d9 ★a1da ○a1db ●a1dc
◎a1dd ◇a1de ◆a1df □a1e0 ■a1e1 △a1e2 ▲a1e3 ▽a1e4 ▼a1e5 →a1e6
←a1e7 ↑a1e8 ↓a1e9 ↔a1ea =a1eb ≪a1ec ≫a1ed √a1ee ∞a1ef ∞a1f0
∴a1f1 ∫a1f2 ∫a1f3 ∈a1f4 ∃a1f5 ⊆a1f6 ⊇a1f7 ⊂a1f8 ⊃a1f9 ∪a1fa
∩a1fb ∧a1fc ∨a1fd ¬a1fe

-----
Process exited after 0.04108 seconds with return value 0
계속하려면 아무 키나 누르십시오 . . .
```

16-5) 확장 예제 (예제 16-15)

● 특수 기호 ■ 와 → 를 출력하는 프로그램을 작성 하시오

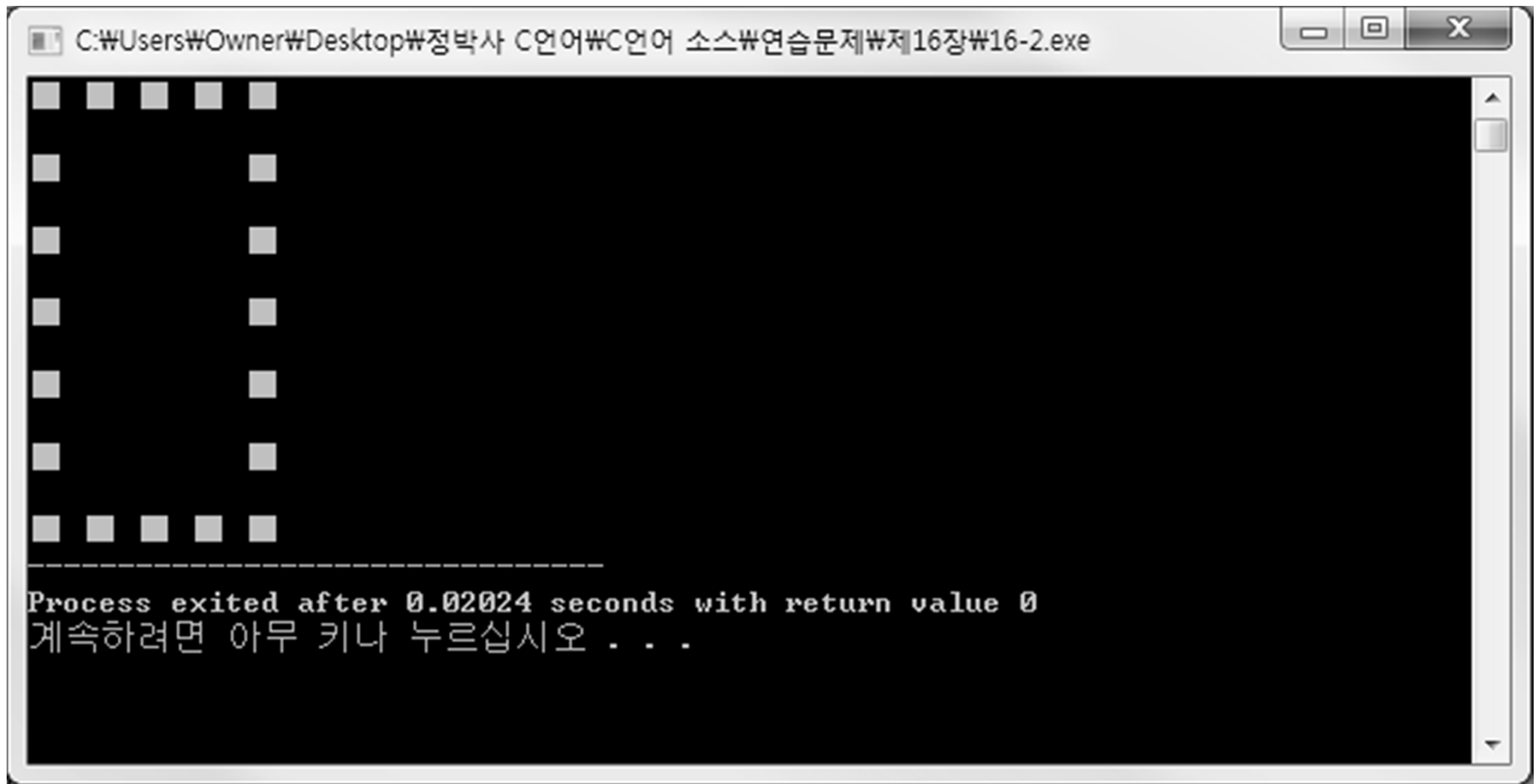
```
#include <stdio.h>

int main(void)
{
    unsigned char a1 = 0xa1;
    unsigned char a2 = 0xe1;
    unsigned char b1 = 0xa1;
    unsigned char b2 = 0xe6 ;

    printf("%c%c\n",a1,a2);
    printf("%c%c\n",b1,b2);
    return 0;
}
```

16. 연습문제

2. gotoxy(int x, int y) 함수와 완성형 특수기호 코드를 이용하여 다음과 같은 그림 형태로 출력하는 프로그램을 작성 하시오

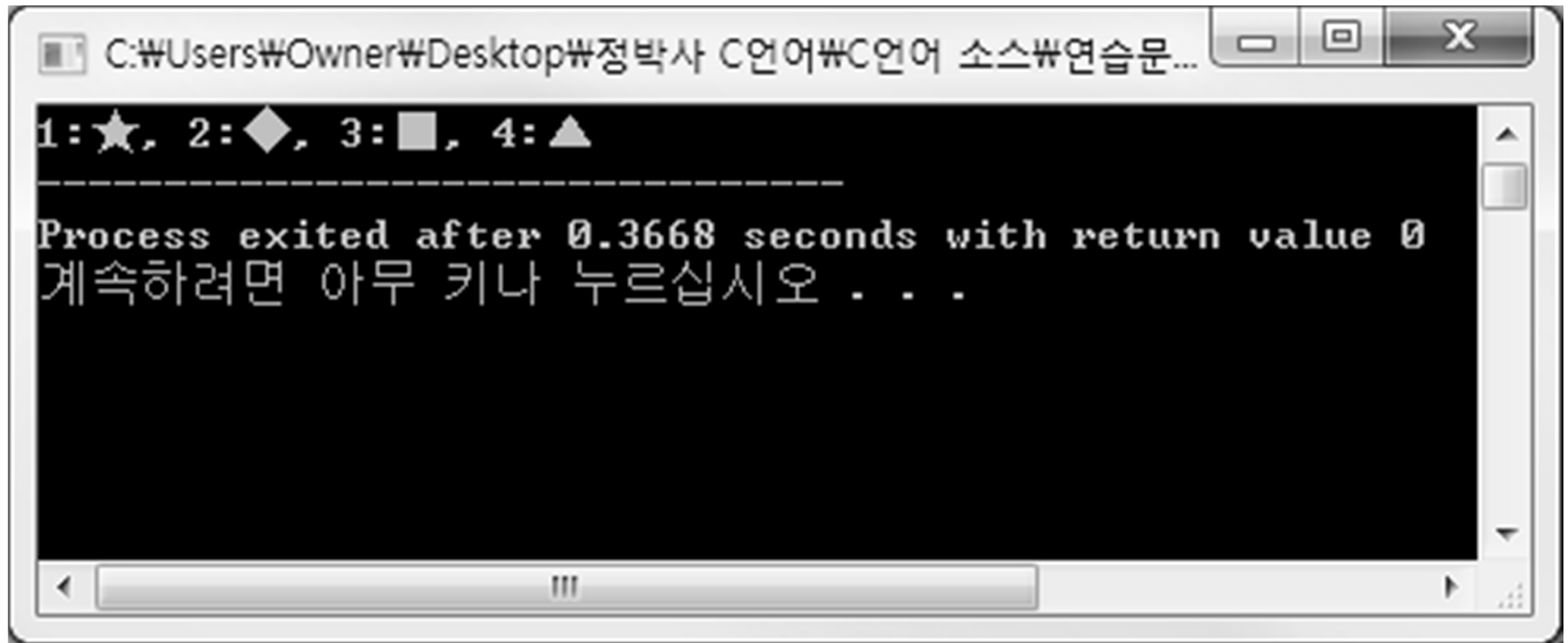


```
C:\Users\Owner\Desktop\정박사 C언어\C언어 소스\연습문제\제16장\16-2.exe

*****
*       *
*       *
*       *
*       *
*       *
*       *
*****
-----
Process exited after 0.02024 seconds with return value 0
계속하려면 아무 키나 누르십시오 . . .
```

16. 연습문제

3. 다음과 같이 출력되는 프로그램을 작성 하시오.



The screenshot shows a Windows command prompt window with the title bar text "C:\Users\Owner\Desktop\정박사 C언어\C언어 소스\연습문...". The window contains the following text:

```
1:★, 2:◆, 3:■, 4:▲  
-----  
Process exited after 0.3668 seconds with return value 0  
계속하려면 아무 키나 누르십시오 . . .
```


16-5) 확장 예제 (예제 16-16)

- getch() 함수를 이용하여 입력되는 아스키 키 값을 16 진으로 출력하는 프로그램을 작성하시오

```
#include <stdio.h>

int main(void)
{
    char key ;

    for( ; ; ) {
        printf("아무키나 누르시오!!\n");
        key = getch();
        if(key=='q') break;

        printf("key = %c 0x%x \n",key,key);
    }
}
```

16-5) 확장 예제 (예제 16-17)

- getch() 함수를 이용하여 입력되는 아스키 값과 특수키 값을 16 진으로 출력하는 프로그램을 작성하시오

```
#include <stdio.h>

int main(void)
{
    int key ;

    for( ; ; ) {
        printf("아무키나 누르시오!!\n");
        key = getch();
        if(key==0 || key==0xe0) {
            key=getch();
            printf("확장키 key = %c 0x%x \n",key,key);
        }
    }
}
```

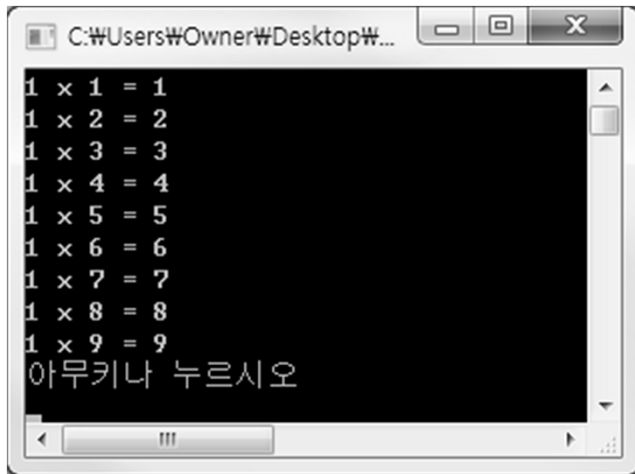
16-5) 확장 예제 (예제 16-17)

- getch() 함수를 이용하여 입력되는 아스키 값과 특수키 값을 16 진으로 출력하는 프로그램을 작성하시오

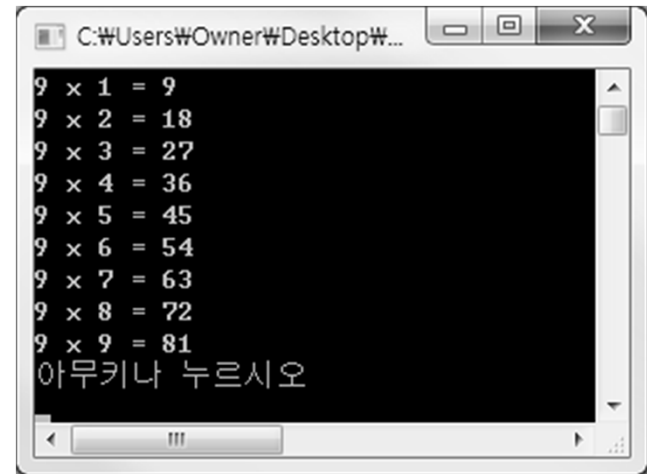
```
else {  
    printf("아스키 키|key = %c 0x%x Wn",key,key);  
}  
  
if(key=='q') break;  
  
}  
}
```

16. 연습문제

4. `system("cls")` ; 함수는 화면을 clear 하는 함수이고, `getch()`는 키값을 받는 함수이다. 이 두 함수를 이용하여 키보드를 칠 때마다 구구단이 1단 부터 9단 까지 바뀌는 프로그램을 작성 해 보도록 한다.



```
C:\Users\Owner\Desktop\...>
1 x 1 = 1
1 x 2 = 2
1 x 3 = 3
1 x 4 = 4
1 x 5 = 5
1 x 6 = 6
1 x 7 = 7
1 x 8 = 8
1 x 9 = 9
아무키나 누르시오
```

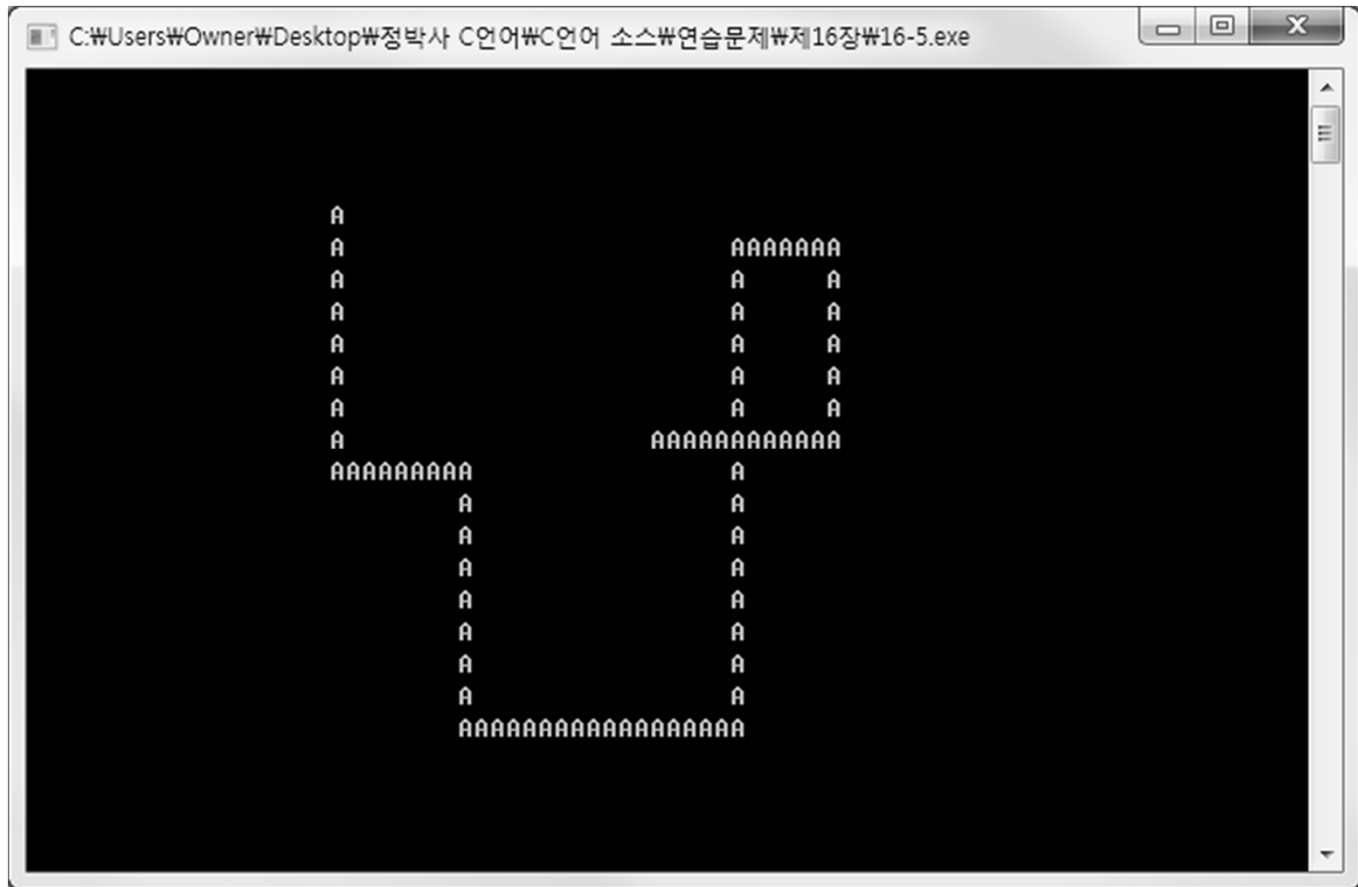


```
C:\Users\Owner\Desktop\...>
9 x 1 = 9
9 x 2 = 18
9 x 3 = 27
9 x 4 = 36
9 x 5 = 45
9 x 6 = 54
9 x 7 = 63
9 x 8 = 72
9 x 9 = 81
아무키나 누르시오
```

16. 연습문제

5. 초기에 화면 (40,12)에 A자를 찍고 화살표 키에 따라 A자가 이동하도록 하시오. (위쪽 화살표 값 :72, 아래 쪽 화살표 값 : 80, 오른쪽 화살표 값:77, 왼쪽 화살표 값 75)

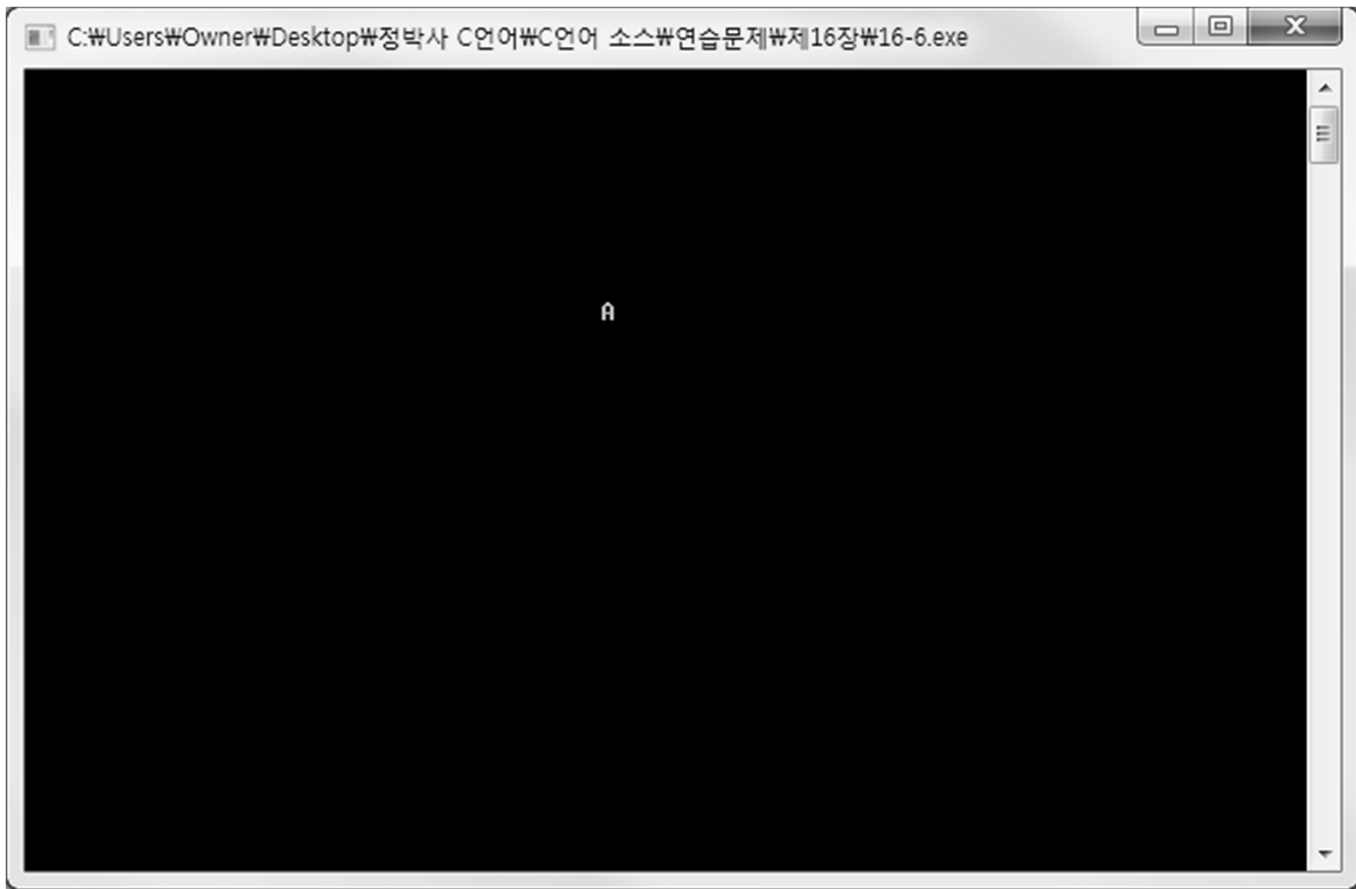
실행 예)



16. 연습문제

6. 연습문제 5번에서는 A자가 계속 나타난다. 이번에는 A자가 이동하면서 그전에 있던 A자를 지우도록 하는 프로그램을 작성해 보도록 하자

실행 예)



16. 연습문제

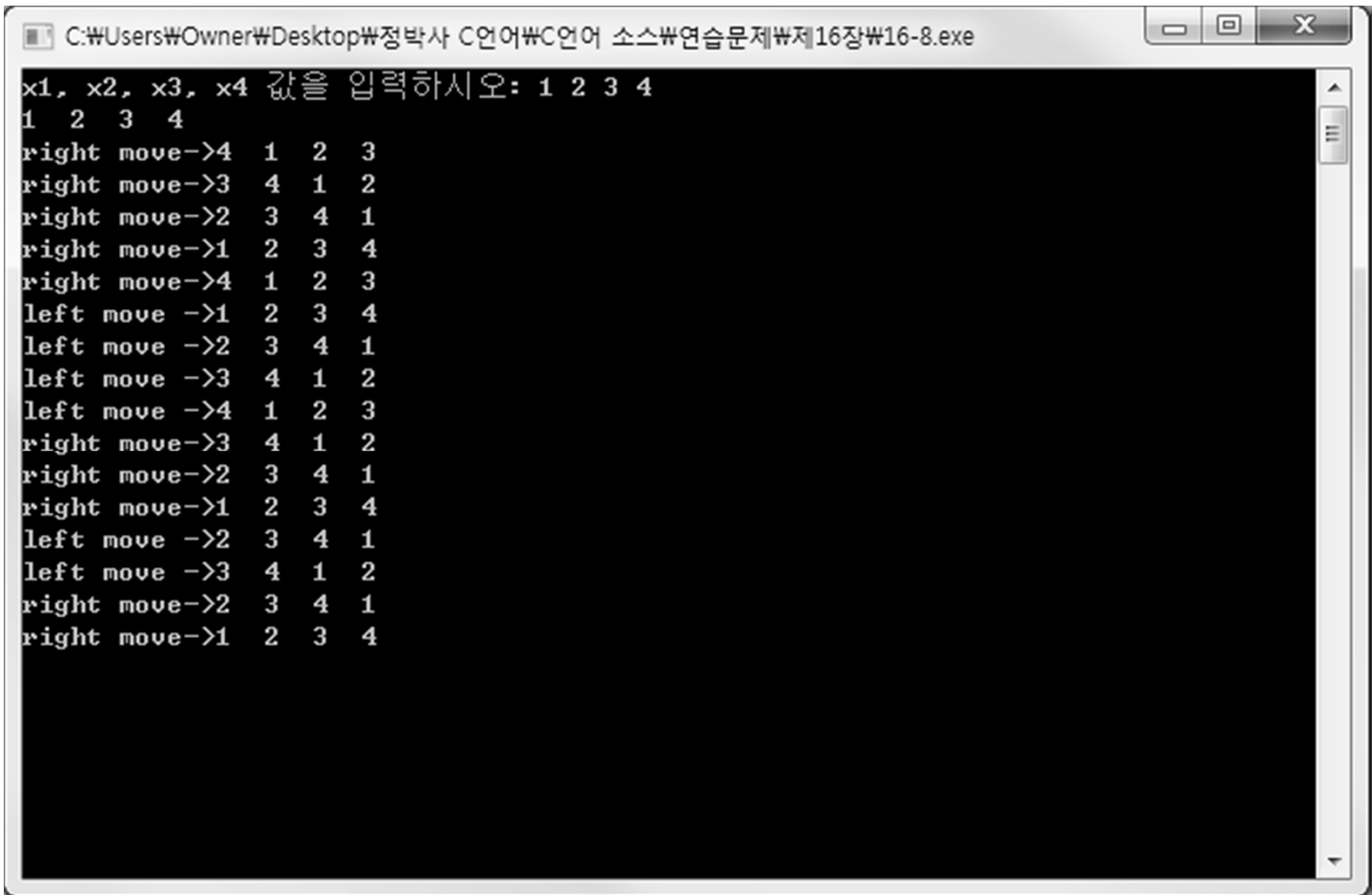
7. 임의의 실수 a , b , c 에 대한 이차방정식 $ax^2 + bx + c = 0$ 을 만족하는 x 를 근의 공식을 이용하여 출력하는 부분을 사용자 정의 함수(void 형)로 작성하여 프로그램을 완성하시오. 단, 근을 갖지 못하는 경우(허근)에는 “근이 없습니다.”을 출력하도록 하시오.

$$\text{근의 공식 } x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

16. 연습문제

8. 변수 x1, x2, x3, x4에 정수 입력을 받아서 화살표 키에 (오른쪽, 왼쪽) 따라서 변수 값이 이동하는 프로그램을 작성 하시오. 'q'가 눌리면 프로그램은 종료 한다. (오른쪽 화살표 값 : 77, 왼쪽 화살표 값 :75)

실행 예)



```
C:\Users\Owner\Desktop\정박사 C언어\소스\연습문제\16장\16-8.exe
x1, x2, x3, x4 값을 입력하시오: 1 2 3 4
1 2 3 4
right move->4 1 2 3
right move->3 4 1 2
right move->2 3 4 1
right move->1 2 3 4
right move->4 1 2 3
left move ->1 2 3 4
left move ->2 3 4 1
left move ->3 4 1 2
left move ->4 1 2 3
right move->3 4 1 2
right move->2 3 4 1
right move->1 2 3 4
left move ->2 3 4 1
left move ->3 4 1 2
right move->2 3 4 1
right move->1 2 3 4
```