C언어의 기초부터 포인터까지 원리를 자세하고 깊이있게
C언어의 참맛을 제대로 느껴보세요

C언어의 정석

기본원리의 자세한 설명으로 암기보다 이해위주의 학습유도
개발자로서 반드시 알아야하는 핵심내용을 체계적으로 정리
특히 포인터를 쉬우면서도 자세하고 체계적으로 설명

남궁 성 지음

소스 및 동영상 다운로드 | http://www.codechoba.com
Q&A게시판 | http://cafe.naver.com/javachobastudy.cafe

퀴즈 및 풀이
ver. 20160830_1

올컬러 편집
소스코드를 이해하기 쉽게 색 전체를 컬러로 편집

저자가 운영하는 Q&A게시판
질문 고민하세요, 답변해드립니다

도우출판

이 문서는 'C언어의 정석(남궁 성 저)'의 퀴즈에 대한 답과 풀이를 제공하기 위한 것입니다. 상업적인 용도가 아니라면 수정없이 제한 없이 재배포가 가능하며, 관련 문의는 http://codechobo.com에 방문해서 글 남겨주시기 바랍니다.

[개정이력]
2016. 8. 30 처음 작성

Quiz 1-1 아래의 예제를 자신의 이름이 출력되도록 변경하고, 실행결과를 예측하시오.

▼ 예제 1-2/ch1/1_2.c

```c
#include <stdio.h>

int main(void) {
    printf("안녕하세요.");
    printf("남궁성입니다.");

    printf("안녕하세요.\n");
    printf("남궁성입니다.");

    return 0;
}
```

[Quiz1-1][답]

[실행결과]

```
안녕하세요.남궁성입니다.안녕하세요.
남궁성입니다.
```

Quiz 2-1 8 비트로 표현할 수 있는 값의 개수와 표현가능한 10진수의 범위는?

[Quiz2-1][답]

8비트는 2진수 8자리(00000000~11111111)를 표현할 수 있으므로 2^8(=256)개의 숫자를 표현할 수 있다. 그래서 10진수로는 0~255범위의 숫자를 표현할 수 있다.

Quiz 2-2 2진수 '1100101011111110'을 8진수와 16진수로 변환하시오.

[Quiz2-2][답]

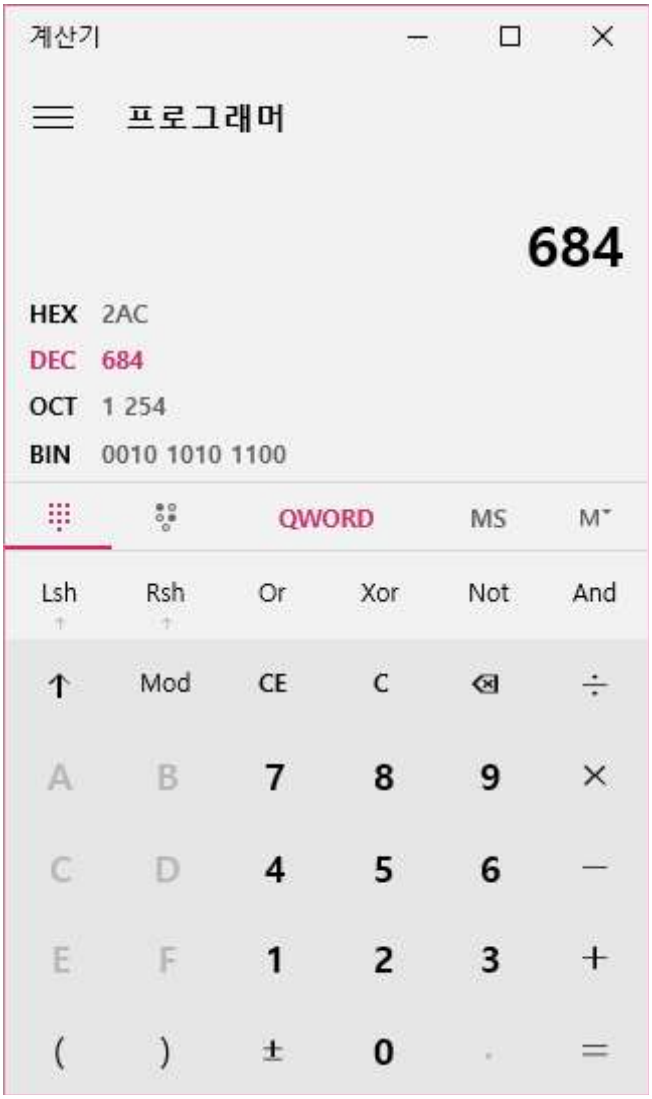
2진수	1	100	101	011	111	110
8진수	1	4	5	3	7	6

2진수	1100	1010	1111	1110
16진수	C	A	F	E

윈도우즈10의 계산기를 프로그래머 모드로 변경하면 2진수, 8진수, 10진수, 16진수로 변환된 값을 쉽게 확인할 수 있다.

Quiz 2-3 10진수 684를 2진수, 8진수, 16진수로 변환하시오.

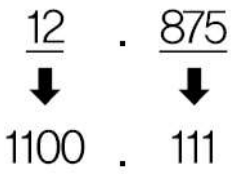
[Quiz2-3][답]
2진수 001010101100
8진수 1254
16진수 2AC



Quiz 2-4 10진 소수점수 12.875를 2진수로 변환하시오.

[Quiz2-4][답] $1100.111_{(2)}$

실수를 2진수로 변환할 때는 정수부분과 소수점이하부분을 따로 변환한다. 그래서 12.875라는 실수를 2진수로 변환하면, 정수부는 '1100'이고 소수부는 '111'이 된다.



Quiz 2-5 -8을 8 bit의 2진수로 변환하시오.

[Quiz2-5][답] $11111000_{(2)}$

8 bit의 2진수라는 말은 8자리의 2진수를 말한다. 10진수 8은 2진수로 1000이므로, 8 bit의 2진수로 00001000이다. 여기에 1의 보수를 구한 다음 1을 더하면 -8의 2진 표현을 구할 수 있다.

0	0	0	0	1	0	0	0
↓	↓	↓	↓	↓	↓	↓	↓
1	1	1	1	0	1	1	1

← '0000 1000'의 '1의 보수'

+) 1

1	1	1	1	1	0	0	0

← '0000 1000'의 '2의 보수'

Quiz 2-6 크기가 8 bit인 부호있는 정수타입과 부호없는 정수로 표현할 수있는 값의 범위를 구하시오.

[Quiz2-6][답]

8 bit로 표현할 수 있는 값의 개수는 2^8(=256)개이므로, 부호없는 정수는 0~2^8-1(0~255) 범위의 값을 표현할 수 있고, 부호있는 정수는 -2^7~2^7-1(128~127)범위의 값을 표현할 수 있다.

Quiz 2-7 126.25를 float타입의 변수에 저장하면 어떻게 저장될까? 32자리의 2진수로 표현하시오. 0.25는 2진수로 0.01이다.

[Quiz2-7][답]

① 10진수를 2진수로 변환

126.25 → 1111110.01

② **정규화**(1.xxx × 2^e의 형태로 변환)

1111110.01 → 1.11111001 × 2^6

③ **가수를 저장**('1.'을 제외하고 왼쪽부터 저장. 빈자리는 0으로 채운다.)

0		11111001000000000000000000

④ 지수에 기저(127)를 더한 후 2진수(8 bit)로 변환

6 + 127 = 133 → 1000 0101

0	100 0010 1	111 1100 1000 0000 0000 0000
4	2	F C 8 0 0 0

Quiz 3-1 아래 연산의 결과타입을 빈 칸에 넣으시오.

char + unsigned → (①)

unsingned + int → (②)

[Quiz3-1][답]① unsgined ② unsgined

char + unsigned → unsigned + unsigned → **unsigned**
unsigned + int → unsigned + unsigned → **unsigned**

Quiz 3-2 다음은 대문자 'X'를 소문자 'x'로 변환하는 식이다. 괄호안을 채우시오.

```
char x = 'X' - (      );
```

[Quiz3-2][답]

```
char x = 'X' - (-32);
```

또는

```
char x = 'X' - ('X'-'x');  // 괄호는 없어도 됨.
```

아스키에서는 대문자와 소문자의 문자코드가 32차이나기 때문에 32를 더하거나 빼면 대소문자간의 변환이 가능하다. 표준에서는 C언어의 문자셋으로 꼭 아스키를 써야한다고 되어있지 않기 때문에, 'X'-'x'와 같은 식으로 대문자와 소문자 간의 코드 차이를 구하는 것이 더 안전하다.

Quiz 3-3 아래의 코드는 변수 pi의 값을 소수점 셋째자리에서 올림하기 위한 것이다. 실행결과와 같은 결과가 나오도록 빈 곳을 채워 코드를 완성하시오.

▼ 예제 3-14/ch3/3_14.c

```
1 #include <stdio.h>
2
3 int main(void) {
4     double pi = 3.141592;
5     double shortPi = (                  );  // 빈 곳을 코드로 채우시오.
6
7     printf("%lf\n", shortPi);
8
9     return 0;
10 }
```

▼ 실행결과

```
3.150000
```

[Quiz3-3][답] (int)(pi * 100 + 0.9) / 100.0

[예제3-14]

```
#include <stdio.h>

int main(void) {
   double pi = 3.141592;
   double shortPi = (int)(pi * 100 + 0.9 ) / 100.0 ;

   printf("%lf\n", shortPi);

   return 0;
}
```

Quiz 3-4 아래의 문장이 수행되었을 때 화면에 출력되는 값은?

```
printf("%u\n", -10);
```

[Quiz3-4][답] 4294967286

-10은 16진수로 0xFFFFFFF6인데, 이 값을 부호있는 정수로 해석하면 10진수로 -10이되고, 부호없는 정수로 해석하면 10진수로 42949672860이 된다. %u는 부호없는 10진 정수형식으로 출력할 때 사용하는 지시자이므로 0xFFFFFFF6을 부호없는 10진 정수로 해석한 4294967286을 결과로 얻는 것이다.

Quiz 3-5 'i가 2의 배수도 아니고 3의 배수도 아니다.'라는 조건을 식으로 작성하시오.

[Quiz3-5][답] i%2!=0 && i%3!=0

Quiz 3-6 다음 식의 평가결과를 적으시오.

0xABCD & 0xF0F0

[Quiz3-6][답] 0xA0C0

	A	B	C	D
	1010	1011	1100	1101
	1111	0000	1111	0000
&)	F	0	F	0
	1010	0000	1100	0000
	A	0	C	0

Quiz 3-7 성적이 90점 이상이면, A학점을, 80점 이상이면 B학점을, 그 외는 C학점을 변수 grade에 저장하도록 아래의 빈 칸을 채우시오.

```
int score  = 85;
char grade = (                    );
```

[Quiz3-7][답]

[예제2-1]

```
#include <stdio.h>

int main(void) {
   int score  = 60;
   char grade = score >=90 ? 'A' : ( score >=80 ? 'B' : 'C');

   printf("score=%d\n", score);
   printf("grade=%c\n", grade);

   return 0;
}
```

Quiz 4-1 int타입의 변수 num의 값이 홀수면 '홀수'라고 출력하고 짝수면 '짝수'라고 출력하는 if-else문을 작성하시오.

[Quiz4-1][답]

```
if(num%2!=0) {    // if(num%2) {
    printf("홀수\n");
} else {
    printf("짝수\n");
}
```

Quiz 4-2 하나의 문자를 입력받아서, 이 문자가 숫자인지, 대문자인지, 소문자인지 확인하는 예제를 작성하시오.

▼ 실행결과 1

```
하나의 문자를 입력하세요. >A
대문자입니다.
```

▼ 실행결과 2

```
하나의 문자를 입력하세요. >9
숫자입니다.
```

▼ 실행결과 3

```
하나의 문자를 입력하세요. >*
숫자나 영문자가 아닙니다.
```

[Quiz1-1][답]

[Quiz4-2]

```
#include <stdio.h>

int main(void) {
   char ch;

   printf("하나의 문자를 입력하세요. >");
   scanf("%c", &ch);

   if('A' <= ch && ch <='Z') {
       printf("대문자입니다.\n");
   } else if ('a' <= ch && ch <= 'z') {
       printf("소문자입니다.\n");
   } else if ('0' <= ch && ch <= '9') {
       printf("숫자입니다.\n");
   } else {
       printf("숫자나 영문자가 아닙니다.\n");
   }

   return 0;
}
```

Quiz 4-3 예제4-5의 외부 if문도 else블럭을 생략할 수 있다. 이 else블럭을 생략하고도 같은 결과를 얻도록 예제를 변경하시오.

▼ 실행결과

```
점수를 입력해주세요.>10
당신의 점수는 10입니다.
당신의 학점은 C0입니다.
```

[Quiz4-3][답] 처음부터 변수 grade에 'C'를 저장하면, else블럭에서 grade에 'C'를 저장하지 않아도 된다.

[Quiz4-3]

```
#include <stdio.h>

int main(void) {
   int score;
   char grade = 'C'; // 처음부터 기본값으로 'C'를 저장
   char opt = '0';

   printf("점수를 입력해주세요.>");
   scanf("%d", &score);
   printf("당신의 점수는 %d입니다.\n", score);

   if (score >= 90) { // score가 90점 보다 같거나 크면 A학점(grade)
        grade = 'A';
        if (score >= 98) { // 90점 이상 중에서도 98점 이상은 A+
             opt = '+';
        }
        else if (score < 94) { // 90점 이상 94점 미만은 A-
             opt = '-';
        }
   }
   else if (score >= 80) { // score가 80점 보다 같거나 크면 B학점(grade)
        grade = 'B';
        if (score >= 88) {
             opt = '+';
        }
        else if (score < 84) {
             opt = '-';
        }
   }
// else { // 나머지는 C학점(grade)
//      grade = 'C';
// }

   printf("당신의 학점은 %c%c입니다.\n", grade, opt);

   return 0;
}
```

Quiz 4-4 조건 연산자를 이용해서 실행결과가 다음과 같이 되도록 예제4-7을 변경하시오.

▼ 실행결과

```
가위(1),바위(2), 보(3) 중 하나를 입력하세요.> 3
당신은 보입니다.
컴은  바위입니다.
당신이 이겼습니다.
```

[Quiz4-4][답] 조건 연산자를 두번 사용하면 된다.

[예제4-4]

```
#include <stdio.h>
#include <time.h> // time()을 사용하기 위해 추가
#include <stdlib.h> // srand(), rand()를 사용하기 위해 추가

int main(void) {
   int user, com;

   printf("가위(1),바위(2), 보(3) 중 하나를 입력하세요.>");
   scanf("%d", &user);

   srand((unsigned)time(NULL)); // 현재시간으로 난수의 씨앗값을 초기화
   com = rand() % 3 + 1; // 1,2,3중 하나가 com에 저장됨

   printf("당신은 %s입니다.\n", user==1 ? "가위" : (user==2 ? "바위" : "보"));
   printf("컴은 %s입니다.\n",  com==1 ?  "가위" : (com==2  ? "바위" : "보"));

   switch (user - com) {
   case 2: case -1:
        printf("당신이 졌습니다.\n");
        break;
   case 1: case -2:
        printf("당신이 이겼습니다.\n");
        break;
   case 0:
        printf("비겼습니다.\n");
        // break; // 마지막 문장이므로 break문을 사용할 필요가 없다.
   }

   return 0;
}
```

Quiz 4-5 예제4-8의 switch문을 if문으로 변경하시오.

[Quiz4-5][답]

[Quiz4-5]

```
#include <stdio.h>

int main(void) {
   char gender;
   char regNo[15]; // 문자열을 저장할 공간은 문자열의 길이 보다 1이 커야 한다.

   printf("당신의 주민번호를 입력하세요.(011231-1111222)>");
   scanf("%s", regNo); // 주민등록번호를 입력받는다.

   gender = regNo[7]; // 입력받은 주민등록번호의 8번째 문자를 gender에 저장

   if(gender=='1' || gender=='3') {
       printf("당신은 남자입니다.\n");
   } else if (gender == '2' || gender == '4') {
       printf("당신은 여자입니다.\n");
   } else {
       printf("유효하지 않은 주민등록번호입니다.\n");;
   }

   return 0;
}
```

Quiz 4-6 예제4-10을 변경해서 10점 단위가 아닌 5점 단위로 학점을 부여하는 예제를 만들어보자.

[Quiz4-6][답]

[Quiz4-6]

```
#include <stdio.h>

int main(void) {
   int score;
   char grade = ' ';

   printf("당신의 점수를 입력하세요.(1~100)>");
   scanf("%d", &score);

   switch (score / 5) {
        case 20:
        case 19:
            grade = 'A';
            break;
        case 18:
            grade = 'B';
            break;
        case 17:
            grade = 'C';
            break;
        case 16:
            grade = 'D';
            break;
        case 15:
            grade = 'E';
            break;
        case 14:
            grade = 'F';
            break;
        case 13:
            grade = 'G';
            break;
        case 12:
            grade = 'H';
            break;
        default:
            grade = 'I';
   }

   printf("당신의 학점은 %c입니다.\n", grade);

   return 0;
}
```

Quiz 4-7 for문과 if문을 이용해서 다음과 같이 출력하시오.

▼ 실행결과

```
123
456
789
```

[Quiz4-7][답]

[예제4-7]

```
#include <stdio.h>

int main(void) {
   int i;

   for(i=1;i<10;i++) {
        printf("%d", i);

        if(i%3==0)
            printf("\n");
   }

   return 0;
}
```

참고로 위의 예제를 while문으로 작성하면 다음과 같다.

[예제4-7]

```
#include <stdio.h>

int main(void) {
   int i=1;

   while(i < 10) {
        printf("%d", i);

        if (i++ % 3 == 0)
            printf("\n");
   }

   return 0;
}
```

Quiz 4-8 1부터 10까지의 곱을 계산하여 출력하시오.

▼ 실행결과

```
1부터  1 까지의 곱: 1
1부터  2 까지의 곱: 2
1부터  3 까지의 곱: 6
1부터  4 까지의 곱: 24
1부터  5 까지의 곱: 120
1부터  6 까지의 곱: 720
1부터  7 까지의 곱: 5040
1부터  8 까지의 곱: 40320
1부터  9 까지의 곱: 362880
1부터 10 까지의 곱: 3628800
```

[Quiz4-8][답] 예제4-14를 약간만 고치면 된다.

[예제4-8]

```
#include <stdio.h>

int main(void) {
   int sum = 1;
   int i;

   for (i = 1; i <= 10; i++) {
        sum *= i; // sum = sum * i;
        printf("1부터 %2d 까지의 곱: %d\n", i, sum);
   }

   return 0;
}
```

Quiz 4-9 예제4-16처럼 반복문에 하나의 변수만을 이용해서 아래와 같이 출력하시오.

▼ 실행결과

```
1       1
2       1
3       1
1       2
2       2
3       2
1       3
2       3
3       3
```

[Quiz4-9][답]

[예제4-9]

```
#include <stdio.h>

int main(void) {
   int i;

   for(i=1;i<10;i++) {
       printf("%d\t%d\n", (i+2)%3+1, (i+2)/3);
   }

   return 0;
}
```

Quiz 4-10 중첩된 for문을 이용해서 아래와 같이 출력하는 두 개의 프로그램을 작성하시오.

▼ 실행결과

```
1
12
123
1234
12345
```

(a)

▼ 실행결과

```
1
22
333
4444
55555
```

(b)

[Quiz4-10][답]

[예제4-10] - (a)

```
#include <stdio.h>

int main(void) {
   int i, j;

   for (i = 1; i <= 5; i++) {
        for (j = 1; j <= i; j++) {
            printf("%d", j);
        }
        printf("\n");
   }

   return 0;
}
```

[예제4-10] - (b)

```
#include <stdio.h>

int main(void) {
   int i, j;

   for (i = 1; i <= 5; i++) {
        for (j = 1; j <= i; j++) {
            printf("%d", i);
        }
        printf("\n");
   }

   return 0;
}
```

Quiz 4-11 중첩 for문을 이용해서 다음과 같이 출력하는 프로그램을 작성하시오.

▼ 실행결과

```
2 x 1 =  2      3 x 1 =  3      4 x 1 =  4      5 x 1 =  5
2 x 2 =  4      3 x 2 =  6      4 x 2 =  8      5 x 2 = 10
2 x 3 =  6      3 x 3 =  9      4 x 3 = 12      5 x 3 = 15
2 x 4 =  8      3 x 4 = 12      4 x 4 = 16      5 x 4 = 20
2 x 5 = 10      3 x 5 = 15      4 x 5 = 20      5 x 5 = 25
2 x 6 = 12      3 x 6 = 18      4 x 6 = 24      5 x 6 = 30
2 x 7 = 14      3 x 7 = 21      4 x 7 = 28      5 x 7 = 35
2 x 8 = 16      3 x 8 = 24      4 x 8 = 32      5 x 8 = 40
2 x 9 = 18      3 x 9 = 27      4 x 9 = 36      5 x 9 = 45
```

[Quiz4-11][답]

[예제4-11]

```
#include <stdio.h>

int main(void) {
   int i, j;

   for (i = 1; i <= 9; i++) {
       for (j = 2; j <= 5; j++)
           printf("%d x %d = %2d\t", j, i, i*j);
       printf("\n");
   }

   return 0;
}
```

Quiz 4-12 중첩된 for문을 이용해서 아래와 같이 출력하는 두 개의 프로그램을 작성하시오. (예제4-22에서 if문의 조건식만 변경하면 된다.)

▼ 실행결과

```
                    [1,5]
                [2,4]
            [3,3]
        [4,2]
[5,1]
```

(a)

▼ 실행결과

```
[1,1]
[2,1][2,2]
[3,1][3,2][3,3]
[4,1][4,2][4,3][4,4]
[5,1][5,2][5,3][5,4][5,5]
```

(b)

[Quiz4-12][답]

[예제4-12] - (a)

```
#include <stdio.h>

int main(void) {
   int i, j;

   for (i = 1; i <= 5; i++) {
        for (j = 1; j <= 5; j++) {
            if (i+j==6)
                printf("[%d,%d]", i, j);
            else
                printf("%5c", ' ');
        }
        printf("\n");
   }

   return 0;
}
```

[예제4-12] - (b)

```
#include <stdio.h>

int main(void) {
   int i, j;

   for (i = 1; i <= 5; i++) {
        for (j = 1; j <= 5; j++) {
            if (i>=j)
                printf("[%d,%d]", i, j);
            else
                printf("%5c", ' ');
        }
        printf("\n");
   }

   return 0;
}
```

Quiz 4-13 입력한 정수가 몇 자리 수인지 출력하는 예제를 작성하시오.

▼ 실행결과

```
정수를 입력하세요.>12345
5자리 수입니다.
```

[Quiz4-13][답]

[예제4-13]

```
#include <stdio.h>

int main(void) {
   int num;
   int len = 0;

   printf("정수를 입력하세요.>");
   scanf("%d", &num);

   if(num==0)
        len = 1;

   while (num) { // while(num!=0) {
        num /= 10; // num = num / 10; num을 10으로 나눈 값을 다시 num에 저장
        len++;
   }

   printf("%d자리 수입니다.\n", len);
   return 0;
}
```

Quiz 4-14 예제4-27을 변경하여 입력된 값의 평균도 함께 출력하는 프로그램을 작성하시오.

▼ 실행결과

```
합계를 구할 숫자를 입력하세요. (종료시 0을 입력)
>>1
>>2
>>5
>>9
>>0
합계:17
평균:4.250000
```

[Quiz4-14][답]

[예제4-14]

```
#include <stdio.h>

int main(void) {
   int num;
   int sum = 0;
   int cnt = 0;  // 입력받은 숫자의 개수를 저장하기 위한 변수
   int flag = 1; // while문의 조건식에 사용될 변수

   printf("합계를 구할 숫자를 입력하세요.(끝내려면 0을 입력)\n");

   while (flag) { // flag의 값이 1이므로 조건식은 참이 된다.
       printf(">>");
       scanf("%d", &num); // 숫자를 입력받는다.

       if (num != 0) {
           sum += num; // num이 0이 아니면, sum에 더한다.
           cnt++;
       }
       else {
           flag = 0; // num이 0이면, flag의 값을 0으로 변경. 조건식은 거짓이 된다.
       }
   }

   printf("합계:%d\n", sum);
   printf("평균:%f\n", sum/(float)cnt);

   return 0;
}
```

Quiz 4-15 예제4-28을 변경하여 몇 번만에 맞추었는지를 출력하는 프로그램을 작성하시오.

▼ 실행결과

```
1과 100사이의 정수를 입력하세요.>50
더 작은 값으로 다시 시도해보세요.
1과 100사이의 정수를 입력하세요.>25
더 작은 값으로 다시 시도해보세요.
1과 100사이의 정수를 입력하세요.>12
더 작은 값으로 다시 시도해보세요.
1과 100사이의 정수를 입력하세요.>6
더 큰 값으로 다시 시도해보세요.
1과 100사이의 정수를 입력하세요.>9
정답입니다.
5번 만에 맞추셨습니다.
```

[Quiz4-15][답]

[예제4-15]

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

int main(void) {
   int input;
   int answer;
   int cnt = 0; // 시도횟수를 저장하기 위한 변수

   srand((unsigned)time(NULL));
   answer = rand() % 100 + 1; // 1~100사이의 임의의 수를 저장

   do {
        printf("1과 100사이의 정수를 입력하세요.>");
        scanf("%d", &input);

        if (input > answer) {
            printf("더 작은 수로 다시 시도해보세요.\n");
        }
        else if (input < answer) {
            printf("더 큰 수로 다시 시도해보세요.\n");
        }
        cnt++;
   } while (input != answer);

   printf("정답입니다.\n");
   printf("%d번 만에 맞추셨습니다.\n", cnt);

   return 0;
}
```

Quiz 4-16 goto문 대신 do-while문을 써서 예제4-33을 다시 작성하시오.

[Quiz4-16][답]

[예제4-16]

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

int main(void) {
   int num1, num2;
   int sum = 0;

   srand(time(NULL)); // 난수를 초기화

   do {
        num1 = rand() % 6 + 1; // 1~6범위의 임의의 수를 얻어서 num1에 저장
        num2 = rand() % 6 + 1;

        printf("num1=%d, num2=%d \n", num1, num2);

        sum += num1 + num2;
   } while(num1 == num2); // num1과 num2의 값이 같으면 다시 주사위를 굴린다.

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

   return 0;
}
```

Quiz 5-1 길이가 10인 int배열 arr을 선언하고, 1과 100사이의 임의의 정수로 초기화한 후에 배열의 모든 요소를 출력하는 코드를 작성하시오.

[Quiz5-1][답]

[예제5-1]

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

int main(void) {
   int arr[100];
   const int LEN = sizeof(arr)/sizeof(arr[0]);
   int i;

   srand((unsigned)time(NULL));

   for(i=0;i<LEN;i++)
       arr[i] = rand()%100+1; // 1~100범위의 임의의 정수를 arr[i]에 저장

   for (i = 0; i<LEN; i++)
       printf("arr[%d]=%d\n",i, arr[i]);

   return 0;
}
```

Quiz 5-2 다음에 주어진 세 개의 배열을 합쳐서 하나의 새로운 배열을 만드는 코드를 작성하시오.

```
int arr1[] = {1,2,3};
int arr2[] = {4,5,6,7};
int arr3[] = {8,9,10};
```

[Quiz5-2][답]

[예제5-2]

```
#include <stdio.h>

int main(void) {
   int arr1[] = { 1,2,3 };
   int arr2[] = { 4,5,6,7 };
   int arr3[] = { 8,9,10 };
   int arr[(sizeof(arr1)+sizeof(arr2)+sizeof(arr3))/sizeof(arr1[0])];
   int i, pos=0;

   const int LEN1 = sizeof(arr1) / sizeof(arr1[0]);
   const int LEN2 = sizeof(arr2) / sizeof(arr2[0]);
   const int LEN3 = sizeof(arr3) / sizeof(arr3[0]);
   const int LEN  = sizeof(arr)  / sizeof(arr[0]);

   for (i = 0; i<LEN1; i++)
       arr[pos++] = arr1[i];

   for (i = 0; i<LEN2; i++)
       arr[pos++] = arr2[i];

   for (i = 0; i<LEN3; i++)
       arr[pos++] = arr3[i];

   for(i=0;i<LEN;i++)
       printf("arr[%d]=%d\n", i, arr[i]);

   return 0;
}
```

memcpy()를 사용하면 3개의 for문 대신 아래와 같이 간단히 처리할 수 있지만, 이 코드를 이해하려면 7장 포인터를 배워야 하므로 참고만 하자.

```
memcpy(arr,          arr1, sizeof(arr1));
memcpy(arr+LEN1,     arr2, sizeof(arr2));
memcpy(arr+LEN1+LEN2, arr3, sizeof(arr3));
```

Quiz 5-3 배열 arr이 다음과 같이 주어졌을 때, 홀수번째 요소의 합과 짝수번째 요소의 합을 출력하시오.

```
int arr[] = { 10, 20, 30, 40, 50, 60, 70 };
```

[Quiz5-3][답]

[예제5-3]

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

int main(void) {
   int arr[] = { 10,20,30,40,50,60,70 };
   const int LEN = sizeof(arr) / sizeof(arr[0]);

   int i, oddSum=0, evenSum=0;

   for (i = 0; i<LEN; i++) {
       if(i%2!=0) {
           oddSum += arr[i];
       } else {
           evenSum += arr[i];
       }
   }

   printf("홀수번째 요소의 합:%d\n", oddSum);
   printf("짝수번째 요소의 합:%d\n", evenSum);

   return 0;
}
```

Quiz 5-4 예제5-9를 응용해서 1부터 9사이의 정수 중에서 중복되지 않은 3개의 숫자를 출력하는 예제를 작성하시오.

[Quiz5-4][답]

[예제5-4]

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

int main(void) {
   int ball[9]; // 9개의 정수값을 저장하기 위한 배열 선언.
   const int LEN = sizeof(ball) / sizeof(ball[0]);
   int i, n, tmp;

   srand((unsigned)time(NULL));

   for (i = 0; i < LEN; i++)
       ball[i] = i + 1; // 배열에 1~9를 저장한다.

                        // 배열에 저장된 값을 섞는다.
   for (i = 0; i < LEN; i++) {
       n = rand() % LEN; // 배열 인덱스 범위(0~8)의 임의의 값을 얻는다.

       tmp = ball[i];
       ball[i] = ball[n];
       ball[n] = tmp;
   }
   // 배열 ball의 앞에서 부터 3개의 요소를 출력한다.
   for (i = 0; i < 3; i++)
       printf("%d", ball[i]);
   printf("\n");

   return 0;
}
```

Quiz 5-5 배열 iArr이 다음과 같이 주어졌을 때, 실행결과처럼 배열을 내림차순으로 정렬한 후 배열의 모든 요소를 출력하시오.

```
int iArr[] = { 3, 7, 2, 4, 1, 5, 6};
```

▼ 실행결과

```
7654321
```

[Quiz5-5][답] 정렬 방법이 내림차순일 때는 오름차순일 때와 달리 조건식이 왼쪽 값이 클 때 두 값을 변경해야 한다.

[예제5-5]

```
#include <stdio.h>

int main(void) {
   int iArr[] = { 3, 7, 2, 4, 1, 5, 6 };

   const int LEN = sizeof(iArr) / sizeof(iArr[0]);
   int i, j, tmp;

   int chk; // 자리바꿈이 발생했는지 확인하는 변수

   for (i = 0; i < LEN - 1; i++) {
       chk = 0; // 매 반복마다 chk를 0으로 초기화 한다.
       for (j = 0; j < LEN - 1 - i; j++) {
           if (iArr[j] < iArr[j + 1]) { // 옆의 값이 크면 서로 바꾼다.
               tmp = iArr[j];
               iArr[j] = iArr[j + 1];
               iArr[j + 1] = tmp;

               chk = 1; // 자리바꿈이 발생했으니 chk를 1로 바꾼다.
           }
       } // end for j

       if (chk == 0) break; // 자리바꿈이 없으면 반복문을 벗어난다.
   } // end of for i

   for (i = 0; i < LEN; i++)
       printf("%d", iArr[i]); // 정렬된 결과를 출력한다.
   printf("\n");

   return 0;
}
```

Quiz 5-6 사용자에게 문자열을 입력받아서 문자열의 길이를 알려주는 프로그램을 작성하시오.

▼ 실행결과

```
단어를 하나만 입력하세요.>hello
입력한 단어:hello
입력한 단어의 문자개수:5
```

[Quiz5-6][답] 입력받은 문자열이 저장된 문자 배열 word의 첫 번째 요소부터 순서대로 널 문자를 만날 때까지 읽는다. strlen()이라는 함수를 사용해도 문자열의 길이를 알아낼 수 있는데, 이 함수를 사용하려면 'string.h'를 포함시켜야 한다.

[예제5-6]

```
#include <stdio.h>
// #include <string.h>   // strlen()을 사용하려면 필요.

int main(void) {
   char word[100]; // = {0};  // 넉넉하게 크기를 잡는다.
   int i, n;

   printf("단어를 하나만 입력하세요.>");
   scanf("%s", word);

   printf("입력한 단어:%s\n", word);

   // 입력한 문자의 수를 센다.
   for(i=0;word[i]!='\0';i++);  //   for(i=0;word[i];i++);와 동일

   printf("입력한 단어의 문자개수:%d\n", i);
// printf("입력한 단어의 문자개수:%d\n", strlen(word));

   return 0;
}
```

Quiz 5-7 다음의 두 문장이 수행된 결과는 같은가? 다르다면 왜 다른지 설명하시오.

```
int arr[3][3] = { {1, 2},  {3, 4}, {5, 6} };
int arr[3][3] = {  1, 2, 3, 4, 5, 6 };
```

[Quiz5-7][답] 2차원 배열을 초기화 할 때 괄호{}를 생략할 수 있지만, 배열의 길이에 비해 괄호{}안의 값의 개수가 적을 때는 괄호{}를 생략하면 결과가 달라질 수 있다. 아래의 예제를 실행해보면 위의 문장들이 어떤 차이가 있는지 확인할 수 있다.

[예제5-7]

```
#include <stdio.h>

int main(void) {
   int arr[3][3] = {{1,2},{3,4},{5,6}};
// int arr[3][3] = {{1,2,0},{3,4,0},{5,6,0}}; // 윗 문장과 동일

   int arr2[3][3] = { 1,2,3,4,5,6 }; // 1행 1열부터 순서대로 채운다.
   int i, j;

   for (i = 0; i<3; i++)
       for (j = 0; j<3; j++)
           printf("arr[%d][%d]=%d\tarr2[%d][%d]=%d\n", i, j, arr[i][j],
                                                       i, j, arr2[i][j]);

   printf("\n");

   return 0;
}
```

[실행결과]

```
arr[0][0]=1     arr2[0][0]=1
arr[0][1]=2     arr2[0][1]=2
arr[0][2]=0     arr2[0][2]=3
arr[1][0]=3     arr2[1][0]=4
arr[1][1]=4     arr2[1][1]=5
arr[1][2]=0     arr2[1][2]=6
arr[2][0]=5     arr2[2][0]=0
arr[2][1]=6     arr2[2][1]=0
arr[2][2]=0     arr2[2][2]=0
```

Quiz 5-8 예제5-23을 변경하여 아래와 같은 실행결과를 출력하는 프로그램을 작성하시오.

▼ 실행결과

```
100 100 100 sum=300
 20  20  20 sum=60
 30  30  30 sum=90
 40  40  40 sum=120
```

[Quiz5-8][답] 2차원 배열의 요소의 합을 행단위로 구해서 출력하는 문제이다. 안쪽 for문의 앞에서 sum의 값을 0으로 초기화해야 한다는 것만 주의하자.

[예제5-8]

```
#include <stdio.h>

#define ROW 4
#define COL 3

int main(void) {
   int score[ROW][COL] = {
        { 100, 100, 100 },
        { 20, 20, 20 },
        { 30, 30, 30 },
        { 40, 40, 40 }
   };

   int i, j, sum;

   // 배열의 모든 요소를 테이블 형태로 출력한다.
   for (i = 0; i < ROW; i++) {
        sum = 0;
        for (j = 0; j < COL; j++) {
            printf("%3d ", score[i][j]);
            sum += score[i][j]; // 각 요소의 값을 sum에 누적한다.
        }
        printf("sum=%d\n", sum);
   }

   return 0;
}
```

Quiz 5-9 변수 i와 j의 값이 아래와 같이 변할 때, 표의 세 번째 열은 식 ①에 의해 계산된 값이다. ①에 어떤 식을 넣어야 할까? 단, 이 식에는 변수 i와 j가 포함되어야 한다.

i	j	①
0	0	0
0	1	1
0	2	2
1	0	3
1	1	4
1	2	5
2	0	6
2	1	7
2	2	8

[Quiz5-9][답] i*3+j

[예제5-9]

```
#include <stdio.h>

int main(void) {
   int i, j;

   for(i=0;i<3;i++)
       for(j=0;j<3;j++)
           printf("%d\t%d\t%d\n", i, j, i*3+j);

   return 0;
}
```

[실행결과]

```
0       0       0
0       1       1
0       2       2
1       0       3
1       1       4
1       2       5
2       0       6
2       1       7
2       2       8
```

Quiz 5-10 학급수가 8개이고 한 학급에 30명인 중학교에서 10개 과목의 성적처리를 하려고 한다. 전교생의 성적을 저장할 수 있는 배열 score를 선언하시오.

[Quiz5-10][답]

```
int score[3][8][30][10]; // 3개 학년 * 8개 학급 * 30명 * 10과목
```

Quiz 5-11 예제5-28을 변경하여 다음과 같은 결과가 나오도록 하시오.

▼ 실행결과

```
[1반]
1-1번 100 100 100 sum=300
1-2번  90  90  90 sum=270
1-3번  80  80  80 sum=240
1-4번  70  70  70 sum=210

[2반]
2-1번  95  95  90 sum=280
2-2번  85  85  80 sum=250
2-3번  75  75  70 sum=220
2-4번  65  65  60 sum=190
```

[Quiz5-11][답] Quiz5-8과 거의 같은 문제이므로 쉽게 해결할 수 있을 것이다.

[예제5-11]

```
#include<stdio.h>

int main(void) {
   int score[2][4][3] = {
       { // 1반
           { 100, 100, 100 },  // 1반 1번
           { 90,  90,  90 },  // 1반 2번
           { 80,  80,  80 },  // 1반 3번
           { 70,  70,  70 },  // 2반 4번
       },
       { // 2반
           { 95,  95,  90 },  // 2반 1번
           { 85,  85,  80 },  // 2반 2번
           { 75,  75,  70 },  // 2반 3번
           { 65,  65,  60 },  // 2반 4번
       }
   };

   int i, j, k, sum;

   const int BAN = sizeof(score) / sizeof(score[0]);
   const int BUN = sizeof(score[0]) / sizeof(score[0][0]);
   const int SUB = sizeof(score[0][0]) / sizeof(score[0][0][0]);

   for (i = 0; i<BAN; i++) {
       printf("[%d반]\n", i + 1);

       for (j = 0; j<BUN; j++) {
           printf("%d-%d번 ", i + 1, j + 1);
           sum = 0;

           for (k = 0; k<SUB; k++) {
               printf("%3d ", score[i][j][k]);
               sum += score[i][j][k];
           }
           printf(" sum=%d", sum);
           puts("");  // printf("\n");
       }
```

```
        puts("");
    }

    return 0;
}
```

Quiz 5-12 예제5-35를 변경하여, 아래와 같은 결과가 나오도록 하시오.

▼ 실행결과

```
번호  국어  영어  수학   총점    평균
================================
 1    100   100   100    300   100.0
 2     25    20    20     65    21.7
 3     35    30    30     95    31.7
 4     45    40    40    125    41.7
================================
총점   205   190   190    585    48.8
```

[Quiz5-12][답] 각 개인의 총점을 누적하기 위한 변수 totalSum을 추가해야 한다.

[예제5-12]

```
#include<stdio.h>

int main(void) {
   int score[][3] = {
       { 100, 100, 100 },
       { 25,  20,  20 },
       { 35,  30,  30 },
       { 45,  40,  40 }
   };

   const int ROW = sizeof(score) / sizeof(score[0]);
   const int COL = sizeof(score[0]) / sizeof(score[0][0]);

   int koreanTotal = 0, englishTotal = 0, mathTotal = 0;
   int totalSum = 0;
   int sum, i, j;

   printf("번호 국어 영어 수학 총점  평균\n");
   printf("================================\n");

   for (i = 0; i < ROW; i++) {
       sum = 0;
       koreanTotal  += score[i][0];
       englishTotal += score[i][1];
       mathTotal    += score[i][2];
       printf(" %d   ", i + 1);

       for (j = 0; j < COL; j++) {
           sum += score[i][j];
           totalSum += score[i][j];
           printf("%3d  ", score[i][j]);
       }
       printf(" %3d  %5.1f\n", sum, sum / (float)COL);
   }

   printf("================================\n");
   printf("총점 %3d   %3d   %3d    %3d   %5.1f\n", koreanTotal, englishTotal,
mathTotal, totalSum, totalSum / (float)(ROW*COL));

   return 0;
}
```

Quiz 5-13 예제5-36을 변경하여, 아래와 같은 결과가 나오도록 하시오.

▼ 실행결과

```
[1반]
번호   국어   영어   수학    총점    평균
================================
 1    100   100   100    300   100.0
 2     90    90    90    270    90.0
 3     80    80    80    240    80.0
 4     70    70    70    210    70.0
================================
총점   340   340   340   1020    85.0

[2반]
번호   국어   영어   수학   총점    평균
================================
 1     95    95    90    280    93.3
 2     85    85    80    250    83.3
 3     75    75    70    220    73.3
 4     65    65    60    190    63.3
================================
총점   320   320   300    940    78.3
```

[Quiz5-13][답]

[예제5-13]

```
#include <stdio.h>
#define STU 4 // 학생수
#define SUB 3 // 과목수

int main(void) {
   int score[][STU + 1][SUB + 1] = {
        { // 1반
            { 100, 100, 100 }, // 1반 1번
            { 90, 90, 90 },    // 1반 2번
            { 80, 80, 80 },    // 1반 3번
            { 70, 70, 70 }     // 1반 4번
        },
        { // 2반
            { 95, 95, 90 }, // 2반 1번
            { 85, 85, 80 }, // 2반 2번
            { 75, 75, 70 }, // 2반 3번
            { 65, 65, 60 }  // 2반 4번
        }
    };

    const int BAN = sizeof(score) / sizeof(score[0]);
    int i, j, k;

    for (i = 0; i < BAN; i++) {
        printf("[%d반]\n", i + 1);
        printf("번호 국어 영어 수학 총점 평균\n");
        printf("================================\n");

        for (j = 0; j < STU; j++) {
            printf(" %d ", j + 1);
```

```
            for (k = 0; k < SUB; k++) {
                score[i][j][SUB]   += score[i][j][k];
                score[i][STU][k]   += score[i][j][k];
                score[i][STU][SUB] += score[i][j][k];

                printf("%3d ", score[i][j][k]);
            }
            // 학생별 총점과 평균을 출력
            printf("%4d %5.1f\n",
                score[i][j][SUB], score[i][j][SUB] / (float)SUB);
        }

        printf("================================\n");
        printf("총점 ");

        // score[i][STU][SUB]도 출력하기 위해 <=로 변경
        for (k = 0; k <= SUB; k++) {
            printf("%3d ", score[i][STU][k]);
        }
        // 전체 총점(score[i][STU][SUB])을 학생수*과목수로 나눠서 전체 평균을 구한다.
        printf("%5.1f ", (float)score[i][STU][SUB]/(STU*SUB));

        printf("\n\n");
    }

    return 0;
}
```

Quiz 5-14 예제5-37을 변경하여, 아래와 같은 결과가 나오도록 하시오.

▼ 실행결과

```
Q1. chair의 뜻은? dmlwk
틀렸습니다. 정답은 의자입니다

Q2. computer의 뜻은? 컴퓨터
정답입니다.

Q3. integer의 뜻은? 정수
정답입니다.

전체 3문제 중 2문제 맞추셨습니다.
```

[Quiz5-14][답]

[예제5-]

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

int main(void) {
   char words[][2][10] = {
        { "chair","의자" }, // words[0][0], words[0][1]
        { "computer","컴퓨터" }, // words[1][0], words[1][1]
        { "integer","정수" } // words[2][0], words[2][1]
   };
   char answer[20];
   int i, rightAnswer=0;

   const int WORD_CNT = sizeof(words) / sizeof(words[0]);

   for (i = 0; i < WORD_CNT; i++) {
        printf("Q%d. %s의 뜻은?", i + 1, words[i][0]);
        scanf("%s", answer);

        if (strcmp(words[i][1], answer) == 0) {
            printf("정답입니다.\n\n");
            rightAnswer++;
        }
        else {
            printf("틀렸습니다. 정답은 %s입니다.\n\n", words[i][1]);
        }
   }
   printf("전체 %d문제 중 %d문제 맞추셨습니다.\n", i, rightAnswer);

   return 0;
}
```

Quiz 6-1 아래의 코드를 실행했을 때의 결과를 예측하고, 실행과정을 단계별로 설명하시오.

▼ 예제 6-3 /ch6/6_3.c

```
1 #include <stdio.h>
2
3 int add(int x, int y) {
4     int result = x + y;
5     return result;
6 }
7
8 int main(void) {
9     printf("%d\n", add(1, add(2, 3)));
10
11    return 0;
12 }
```

[Quiz6-1][답] 안쪽이 먼저 계산되어야 바깥쪽을 계산할 수 있으므로 안쪽의 add()함수가 먼저 호출된다.

[실행결과]

```
6
```

```
    printf("%d\n", add(1, add(2, 3)));
→ printf("%d\n", add(1, 5));
→ printf("%d\n", 6);
```

Quiz 6-2 두 개의 변수 중에서 큰 값을 구하는 함수(max)는 다음과 같다. 이 함수를 이용해서 3개의 변수 중에서 제일 큰 수를 구하는 함수 max3를 완성하시오.

```
int max(int x, int y) {
     return x > y ? x : y ;
}

int max3(int x, int y, int z) {
     /* 함수를 완성하시오 */
}
```

[Quiz6-2][답] max3()는 아래와 같이 max()를 이용해서 작성하면 쉽고 편리하다.

[예제6-2]

```
#include <stdio.h>

int max(int x, int y) {
   return x > y ? x : y;
}

int max3(int x, int y, int z) {
   return max(x, max(y, z));
}


int main(void) {
   printf("%d\n", max3(1, 2, 3));
   printf("%d\n", max3(3, 2, 1));
   printf("%d\n", max3(2, 1, 3));

   return 0;
}
```

[실행결과]

```
3
3
3
```

예를 들어 max(1, max(2, 3))은 다음과 같은 순서로 처리된다.

```
   max(1, max(2, 3))
→  max(1, 3)
→  3
```

Quiz 6-3 아래의 함수에서 발생할 수 있는 문제점은 무엇인지 알아내고, 그 해결책을 제시하시오.

```
int multiply(int x, int y) {
     int result = x * y;
     return result;
}
```

[Quiz6-3][답] 함수 multiply의 매개변수 x, y는 타입이 int이므로 곱한 결과가 int타입의 범위를 넘어설 수도 있다. 그래서 x와 y를 곱하기 전에 어느 한쪽을 long long과 같이 int보다 큰 타입으로 형변환 해야하고, 이 연산결과를 저장할 변수인 result의 타입도 long long타입으로 해야한다.

```
long long multiply(int x, int y) {
    long long result = (long long)x * y;
    return result;
}
```

[예제6-]

```
#include <stdio.h>

int multiply(int x, int y) {
   int result = x * y;
   return result;
}

long long multiply2(int x, int y) {
   long long result = (long long)x * y;
   return result;
}

int main(void) {
   int       result   = multiply(1000000, 10000000);
   long long result2 = multiply2(1000000, 10000000);

   printf("result =%d\n",   result);
   printf("result2=%lld\n", result2);

   return 0;
}
```

[실행결과]

```
result =1316134912
result2=10000000000000
```

Quiz 6-4 함수 printGugudan이 다음과 같이 정의되어 있을 때, 매개변수 dan의 값이 2와 9사이가 아닌 경우, 아무런 작업도 하지 않고 함수를 종료하도록 변경하시오.

```
void printGugudan(int dan) {
     int i;
     for(i=1;i<=9;i++)
          printf("%d*%d=%2d\n", dan, i, dan*i);
}
```

[Quiz6-4][답]

[예제6-4]

```
#include <stdio.h>

void printGugudan(int dan) {
   int i;
   for (i = 1; i <= 9; i++)
        printf("%d*%d=%2d\n", dan, i, dan*i);
}

void printGugudan2(int dan) {
   int i;

   if(!(2 <= dan && dan <= 9)) return; // if(2 > dan || dan > 9) return;

   for (i = 1; i <= 9; i++)
        printf("%d*%d=%2d\n", dan, i, dan*i);
}

int main(void) {
   printGugudan(10);
   printGugudan2(10);

   return 0;
}
```

Quiz 6-5 헤더파일이 변경되었을 때, 이 헤더파일을 포함하고 있는 소스파일은 다시 컴파일해야 하나? 아니면 다시 컴파일할 필요가 없을까?

[Quiz6-5][답] 헤더파일은 컴파일할 때, 소스파일의 일부로 간주되므로 헤더파일의 변경은 소스파일의 변경과 같다. 그래서 헤더파일이 변경되면 해당 헤더파일을 include하는 소스파일은 모두 다시 컴파일해야 한다.

Quiz 6-6 다음의 함수를 재귀 호출 대신 반복문을 사용하도록 변경하시오.

```
long long power(int x, int n) {
    if(n == 1)
        return x;
    return x * power(x, n-1);
}
```

[Quiz6-6][답]

[예제6-6]

```
long long power(int x, int n) {
   if (n == 1)
       return x;
   return x * power(x, n - 1);
}

long long power2(int x, int n) {
   long long result = 1;

   while (n--) {
       result *= x;
   }

   return result;
}

#include <stdio.h>

int main(void) {
   printf("%lld\n", power(2, 3));
   printf("%lld\n", power2(2, 3));

   return 0;
}
```

Quiz 7-1 64 bit OS에서 가능한 메모리 주소의 범위는?

[Quiz7-1][답] 64 bit OS에서는 주소를 표현하는데 8 byte(=64 bit,)를 사용하기 때문에 2^{64}-1 byte만큼의 메모리(0x0~0xFFFFFFFFFFFFFFFF)를 다룰 수 있다. 즉, 최대 16 EB(엑사바이트, 약 1,600만 TB)의 메모리를 다룰 수 있는 것이다.

$$
\begin{aligned}
2^{64} = 2^{32} \times 2^{32} &= 4 \times 2^{30} \times 4 \times 2^{30} \\
&= 16 \times 2^{30} \times 2^{30} \\
&= 16 \times 2^{10} \times 2^{10} \times 2^{10} \times 2^{10} \times 2^{10} \times 2^{10} \\
&= 16 \times 1024 \times 1024 \times 1024 \times 1024 \times 1024 \times 1024 \\
&\qquad\quad \text{Kilo} \quad \text{Mega} \quad \text{Giga} \quad \text{Tera} \quad \text{Peta} \quad \text{Exa} \\
&= 16\ \text{EB(Exa byte)}
\end{aligned}
$$

Quiz 7-2 다음의 문장들이 수행되고 난 후의 출력결과를 예측하시오. 단, 변수 i의 주소는 0x100이라고 가정한다.

```
int  i   = 100;
int* ptr = &i;
int  i2  = *ptr;

*ptr = 200;

printf("i    =%d\n",  i);
printf("i2   =%d\n",  i2);
printf("*ptr =%d\n", *ptr);
printf("&i   =%p\n",  &i);
printf("ptr  =%p\n",  ptr);
```

[Quiz7-2][답]

[예제7-2]

```
#include <stdio.h>

int main(void) {
   int i = 100;
   int* ptr = &i;
   int i2 = *ptr;

   *ptr = 200;
   printf("i = %d\n", i);
   printf("i2 = %d\n", i2);
   printf("*ptr = %d\n", *ptr);
   printf("&i = %p\n", &i);
   printf("ptr = %p\n", ptr);

   return 0;
}
```

[실행결과]

```
i = 200
i2 = 100
*ptr = 200
&i = 00000100
ptr = 00000100
```

Quiz 7-3 배열 arr과 이 배열을 가리키는 포인터 ptr이 아래와 같이 선언되어 있을 때, 포인터 ptr을 이용해서 배열 arr의 모든 요소를 합하고 평균을 출력하는 예제를 작성하시오.

```
int arr[] = {1,2,3,4,5,6};
int *ptr  = &arr[0];
```

▼ 실행결과

```
sum=21
avg=3.500000
```

[Quiz7-3][답]

[예제7-3]

```
#include <stdio.h>

int main(void) {
   int arr[] = {1,2,3,4,5,6};
   int *ptr  = &arr[0];

   const int LEN = sizeof(arr) / sizeof(arr[0]);
   int len = LEN;
   int sum = 0;

   while(len--)
        sum += *ptr++;

   printf("sum=%d\n", sum);
   printf("avg=%f\n", sum/(float)LEN);

   return 0;
}
```

Quiz 7-4 다음의 문장중 에러가 발생하지 않는 것을 고르시오.

```
int arr[1] = { 100 };
int *ptr = &arr[0];
int i = 1, sum = 0;
```

① arr++;　　② ptr = arr + 1;
③ arr = arr + 1;　　④ sum += *(arr+i);
⑤ sum += *arr++;　　⑥ sum += (*arr)++;

[Quiz7-4][답] ② ④ ⑥

```
① arr++;          // 에러. arr은 상수이므로 ++연산을 적용할 수 없음.
② ptr = arr + 1;  // OK.
③ arr = arr + 1;  // 에러. arr은 상수이므로 다른 값을 대입할 수 없음.
④ sum += *(arr+i); // OK. sum += arr[i];와 동일한 문장
⑤ sum += *arr++;  // 에러. arr은 상수이므로 ++연산을 적용할 수 없음.
⑥ sum += (*arr)++; // OK. *arr은 arr[0]과 같으며, arr이 아닌 배열의 첫번째 요소인
                   // arr[0], 즉 *arr을 증가시키는 것임
```

Quiz 7-5 arr이 길이가 4인 int배열일 때, sizeof(arr)과 sizeof(arr+0)의 결과는?

[Quiz7-5][답] 16, 4
arr과 arr+0의 값은 동일하지만, 타입이 다르다. 배운 것과 같이 배열은 덧셈연산에서 포인터 타입으로 자동형변환 된다. 심지어 0을 더할때도 마찬가지이다.

[예제7-5]

```
#include <stdio.h>

int main(void) {
   int arr[4];

   printf("%d\n", sizeof(arr));
   printf("%d\n", sizeof(arr+0));

   return 0;
}
```

[실행결과]

```
16
4
007EF9F0
007EF9F0
```

Quiz 7-6 배열 arr이 다음과 같이 선언되어 있을 때, 제시된 식의 결과를 적으시오. arr의 값은 0x100이라고 가정한다.

```
int arr[1][1] = { 5 };
```

① &arr[0][0]　　② *arr[0]　　③ *(int**)arr

[Quiz7-6][답]

[예제7-6]

```
#include <stdio.h>

int main(void) {
   int arr[1][1] = {5};

   printf("&arr[0][0]=%p\n", &arr[0][0]);
   printf("*arr[0]=%d\n", *arr[0]);
   printf("*(int**)arr=%d\n", *(int**)arr);

   return 0;
}
```

[실행결과]

```
&arr[0][0]=005EFC58
*arr[0]=5
*(int**)arr=5
```

Quiz 7-7 다음과 같이 배열 arr과 이 배열을 가리키는 포인터 ptr이 선언되어 있을 때, ①~⑤의 값과 타입을 구하시오. 단, 배열의 주소는 0x100이라고 가정한다.

```
int arr[2][3] = {1,2,3,4,5,6};
int (*ptr)[3] = arr;
```

① **ptr ② ptr+2 ③ &ptr[1][2] ④ &ptr[2] ⑤ ptr[1]+0

[Quiz7-7][답]

[예제7-7]

```
#include <stdio.h>

int main(void) {
   int arr[2][3] = { 1,2,3,4,5,6 };
   int(*ptr)[3] = arr;

   printf("%d\n", **ptr);

   ptr = (int(*)[3])0x100;

   printf("%p\n", ptr+2);
   printf("%p\n", &ptr[1][2]);
   printf("%p\n", &ptr[2]);
   printf("%p\n", ptr[1]+0);
}
```

[실행결과]

```
1
00000118
00000114
00000118
0000010C
```

Quiz 7-8 포인터 ptr이 다음과 같이 선언되어 있을 때, ①~④의 결과를 계산하시오.
[Hint] 예제7-25의 실행결과를 참고

```
int (*ptr)[3][4] = (int(*)[3][4])0x100;
```

① *(ptr[0]+1)+2 ② sizeof(ptr[1]) ③ sizeof(ptr+1) ④ &ptr[1]

[Quiz7-8][답]

[예제7-8]

```
#include <stdio.h>

int main(void) {
   int(*ptr)[3][4] = (int(*)[3][4])0x100;

   printf("%p\n", *(ptr[0] + 1) + 2);
   printf("%d\n", sizeof(ptr[1]));
   printf("%d\n", sizeof(ptr + 1));
   printf("%p\n", &ptr[1]);
}
```

[실행결과]

```
00000118
48
4
00000130
```

Quiz 7-9 다음은 char배열 chArr의 내용을 포인터 pch가 가리키는 문자열 상수"ABC"로 변경하는 예제이다. 빈 칸을 채워 예제를 완성하시오. 단, 반복문을 사용해야 한다.

▼ 예제 7-28/ch7/7_28.c

```
#include <stdio.h>

int main(void) {
    char chArr[10];
    char *ps = "ABC";
    int  i = 0;

    /*
         알맞은 내용을 넣어 예제를 완성하시오.
    */

    printf("chArr=%s\n", chArr);
    return 0;
}
```

▼ 실행결과

```
chArr=ABC
```

[Quiz7-9][답]

[예제7-9]

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

int main(void) {
   char chArr[10];
   char *ps = "ABC";
   int i = 0;

//  strcpy(chArr, ps);
   for(i=0;ps[i];i++)  // ps[i]가 널이 아닌동안 반복한다.
       chArr[i] = ps[i];

   chArr[i] = '\0';  // 마지막에 널 문자를 넣어준다.

   printf("chArr=%s\n", chArr);
   return 0;
}
```

위의 for문 대신 strcpy()를 사용해도 되지만 너무 간단해서 for문을 사용하도록 했다. 다음과 같이 for문을 더욱 간단히 하거나 while문을 사용할 수도 있다.

```
for(;chArr[i]=ps[i]; i++); // 널 문자까지 chArr에 저장됨
      또는
while(chArr[i++]=ps[i]);   // 널 문자까지 chArr에 저장됨
```

Quiz 7-10 위의 예제를 변경해서 사전의 역순(reverse order)으로 정렬하는 프로그램을 작성하시오.

▼ 실행결과

```
[ABC,abc,123,2,aa,AAA,111,CCC,AAB,]
[111,123,2,AAA,AAB,ABC,CCC,aa,abc,]
```

[Quiz7-10][답]

[예제7-10]

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

int main(void) {
   char* strArr[] = { "ABC", "abc", "123", "2", "aa", "AAA", "111",
        "CCC", "AAB" };

   const int LEN = sizeof(strArr) / sizeof(strArr[0]);
   int i, j;
   char *tmp;

   printf("[");
   for (i = 0; i < LEN; i++) {
       printf("%s,", strArr[i]);
   }
   printf("]\n");

   for (i = 0; i < LEN - 1; i++) {
       for (j = 0; j < LEN - i - 1; j++) {
           if (strcmp(strArr[j], strArr[j + 1]) > 0) {
               tmp = strArr[j];
               strArr[j] = strArr[j + 1];
               strArr[j + 1] = tmp;
           }
       }
   }
   printf("[");
   for (i = 0; i < LEN; i++) {
       printf("%s,", strArr[i]);
   }
   printf("]\n");

   return 0;
}
```

Quiz 7-11 ①과 ②에 들어갈 내용은?

```
char* name[5][3];
[   ①   ] = name;        // ptr을 선언하고, 배열 name의 주소를 저장

char name2[3][4][6];
[   ②   ] = name2;       // ptr2를 선언하고, 배열 name2의 주소를 저장
```

[Quiz7-][답]
① char* (*ptr)[3]
② char (*ptr2)[4][6]

Quiz 8-1 함수 swapStr이 두 포인터가 가리키는 문자열을 서로 바꾸는 일을 한다고 가정하고 작성된 예제이다. 함수 swapStr을 작성하여 아래의 예제를 완성하시오.

▼ 예제 8-5/ch8/8_5.c

```
1  #include <stdio.h>
2
3   /*
4     ① 함수 swapStr을 정의하시오.
5   */
6
7  int main(void) {
8     char* str1 = "ABC";
9     char* str2 = "123";
10
11    printf("str1=%s, str2=%s\n", str1, str2);
12
13    /*
14    ② 여기에 위에서 정의한 swapStr함수를 호출하는 문장을 넣으시오.
15    */
16
17    printf("str1=%s, str2=%s\n", str1, str2);
18
19    return 0;
20 }
```

▼ 실행결과

```
str1=ABC, str2=123
str1=123, str2=ABC
```

[Quiz8-1][답]

[예제8-]

```
#include<stdio.h>

void swapStr(char** p1, char** p2) {
   char* tmp = *p1;
   *p1 = *p2;
   *p2 = tmp;
}

int main(void) {
   char* str1 = "ABC";
   char* str2 = "123";

   printf("str1=%s, str2=%s\n", str1, str2);
   swapStr(&str1, &str2);    // swapStr함수를 호출
   printf("str1=%s, str2=%s\n", str1, str2);

   return 0;
}
```

Quiz 8-2 예제8-6에 배열의 최대값을 반환하는 함수 maxArr를 추가하고, 이 함수를 이용해서 아래와 같은 실행결과를 얻도록 예제를 변경하시오.

▼ 실행결과

```
arr[0]=1
arr[1]=2
arr[2]=3
arr[3]=4
arr[4]=5
sum=15
max=5
```

[Quiz8-2][답]

[예제8-2]

```
#include <stdio.h>

int sumArr(int arr[], int len);     // 배열의 모든 요소의 합을 반환
void printArr(int arr[], int len); // 배열의 모든 요소를 출력
int maxArr(int arr[], int len);     // 배열의 요소중 가장 큰 값을 반환

int main(void) {
   int arr[] = { 1,2,3,4,5 };
   const int LEN = sizeof(arr) / sizeof(arr[0]);

   int sum = sumArr(arr, LEN);

   printArr(arr, LEN);
   printf("sum=%d\n", sum);
   printf("max=%d\n", maxArr(arr, LEN));
   return 0;
}

int maxArr(int arr[], int len) {
   int max = arr[0];

   while (len--) {
        if(max < *arr)
            max = *arr;
        arr++;
   }

   return max;
}

int sumArr(int arr[], int len) { // int sumArr(int* arr, int len) {
   int sum = 0;

   while (len--)
        sum += *arr++; // 배열의 모든 요소의 값을 sum에 누적한다.

   return sum;
}

void printArr(int arr[], int len) { // void printArr(int *arr, int len) {
```

```
    int i;

    for (i = 0; i < len; i++)
        printf("arr[%d]=%d\n", i, arr[i]);
}
```

Quiz 8-3 예제8-9에 2차원 문자열 상수 배열을 사전순으로 정렬하는 sort( )라는 함수를 추가하고, 이 함수를 이용해서 shuffle( )에 의해 뒤섞인 배열을 정렬해서 출력하도록 예제를 변경하시오.

```
void sort(char** pArr, int len) {
        /* 내용을 완성하시오. */
}
```

[Quiz8-3][답] 1차원 배열을 정렬하는 것과 똑같이 하면 된다.

[예제8-3]

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

void shuffle(char* (*arr)[4], int row, int col); // 배열을 섞는다.
void printArr(char* (*ptr)[4], int row, int col); // 배열을 출력한다.
void sort(char** pArr, int len);

int main(void) {
   char* str2DArr[][4] = {
       { "(0,0)","(0,1)","(0,2)","(0,3)" },
       { "(1,0)","(1,1)","(1,2)","(1,3)" },
       { "(2,0)","(2,1)","(2,2)","(2,3)" }
   };

   const int ROW = sizeof(str2DArr) / sizeof(str2DArr[0]);
   const int COL = sizeof(str2DArr[0]) / sizeof(str2DArr[0][0]);

   srand((unsigned)time(NULL));

   printArr(str2DArr, ROW, COL);
   printf("\n");
   shuffle(str2DArr, ROW, COL);
   printArr(str2DArr, ROW, COL);
   printf("\n");
   sort(str2DArr, ROW * COL);
   printArr(str2DArr, ROW, COL);

   return 0;
}

void sort(char** pArr, int len) {
   int i,j;
   char* tmp;

   for (i = 0; i < len - 1; i++) {
       for (j = 0; j < len - 1 - i; j++) {
           if(strcmp(pArr[j], pArr[j+1]) > 0) {
               tmp = pArr[j];
               pArr[j] = pArr[j+1];
               pArr[j+1] = tmp;
           }
       }
   }
}
```

```
void shuffle(char* (*arr)[4], int row, int col) {
   int i, j;
   int idx1, idx2;
   char* tmp;

   for (i = 0; i < row; i++) {
        for (j = 0; j < col; j++) {
            idx1 = rand() % row;
            idx2 = rand() % col;
            tmp = arr[i][j];
            arr[i][j] = arr[idx1][idx2];
            arr[idx1][idx2] = tmp;
        }
   }
}

void printArr(char* (*ptr)[4], int row, int col) {
   int i, j;
   for (i = 0; i < row; i++) {
        for (j = 0; j < col; j++)
            printf("%s ", ptr[i][j]);
        printf("\n");
   }
}
```

[실행결과]

```
(0,0) (0,1) (0,2) (0,3)
(1,0) (1,1) (1,2) (1,3)
(2,0) (2,1) (2,2) (2,3)

(0,1) (2,2) (1,2) (0,3)
(2,3) (1,3) (0,0) (2,0)
(0,2) (2,1) (1,0) (1,1)

(0,0) (0,1) (0,2) (0,3)
(1,0) (1,1) (1,2) (1,3)
(2,0) (2,1) (2,2) (2,3)
```

Quiz 8-4 두 배열이 다음과 같이 선언되어 있을 때, 두 배열의 차이점을 모두 말하시오.

```
int gv_arr[100];

int main(void) {
     int lv_arr[100];
     ...
}
```

[Quiz8-4][답]

1. gv_arr은 같은 프로그램 내의 모든 소스파일 내에서 접근가능하지만, lv_arr은 main함수 내에서만 접근가능하다.
2. gv_arr은 모든 요소가 0으로 자동 초기화되지만, lv_arr은 자동 초기화되지 않으므로 쓰레기 값이 들어있다.
3. lv_arr은 main함수의 종료와 함께 메모리에서 자동 제거되지만, gv_arr은 프로그램이 종료되어 메모리에서 제거될 때 자동 제거된다. main함수가 종료되면 프로그램 전체가 종료되므로 두 배열 모두 거의 같은 시기에 종료되지만, main함수가 종료된 후에 프로그램 전체가 종료되는 것이므로 gv_arr이 lv_arr보다 약간 더 늦게 메모리에서 제거된다.

Quiz 8-5 2차원 배열 arr과 같은 크기의 메모리를 동적으로 할당받아서, 배열 arr의 내용을 그대로 복사하고 출력하는 예제를 작성하시오.

```
int arr[3][4] = { 1,2,3,4,5,6,7,8,9,10,11,12 };
```

[Quiz8-][답]

[예제8-]

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

int main(void) {
   int arr[3][4] = {1,2,3,4,5,6,7,8,9,10,11,12};
   const int LEN = sizeof(arr) / sizeof(int);
   int i, len = LEN;

   int *arr2 = malloc(sizeof(arr));

   int* pArr  = (int*)arr;
   int* pArr2 = arr2;

   while(len--)
       *pArr2++ = *pArr++;

   for(i=0;i<LEN;i++) {
       printf("arr2[%d]=%d\n", i, arr2[i]);
   }

   return 0;
}
```

[실행결과]

```
arr2[0]=1
arr2[1]=2
arr2[2]=3
arr2[3]=4
arr2[4]=5
arr2[5]=6
arr2[6]=7
arr2[7]=8
arr2[8]=9
arr2[9]=10
arr2[10]=11
arr2[11]=12
```

Quiz 8-6 지정된 문자열을 대문자로 변환하여 반환하는 함수 toUpperCase( )를 아래와 같은 실행결과가 나오도록 완성하시오.

▼ 예제 8-18/ch8/8_18.c

```
1  #include <stdio.h>
2  #include <stdlib.h>
3  #include <string.h>
4
5  char* toUpperCase(char* str) {
6    /* 코드를 넣어 완성하시오. */
7  }
8
9  int main(void) {
10    printf("%s -> %s\n", "abc",    toUpperCase("abc"));
11    printf("%s -> %s\n", "a1b2c3", toUpperCase("a1b2c3"));
12
13    return 0;
14 }
```

▼ 실행결과

```
abc -> ABC
a1b2c3 -> A1B2C3
```

[Quiz8-6][답]

[예제8-6]

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

char* toUpperCase(char* str) {
   char* newStr = malloc((strlen(str) + 1));
   char* p = newStr;

   if (str == NULL) return str;

   while (*str) {
       if ('a' <= *str && *str <= 'z')
           *p++ = (char)(*str++ - 32);
       else
           *p++ = *str++;
   }

   *p = '\0';

   return newStr;
}

int main(void) {
   printf("%s -> %s\n", "abc", toUpperCase("abc"));
```

```
    printf("%s -> %s\n", "a1b2c3", toUpperCase("a1b2c3"));

    return 0;
}
```

Quiz 8-7 예제8-19에 함수 sort( )를 추가하여, 아래의 실행결과와 같이 입력받은 단어들을 사전순으로 정렬해서 보여주도록 하시오.

▼ 실행결과

```
저장할 단어를 입력하세요.(ENTER:종료)
>tiger
>lion
>cat
>hippo
>dog
>

입력하신 단어는 다음과 같습니다.
[cat, dog, hippo, lion, tiger, ]
```

[Quiz8-7][답]

[예제8-7]

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

void add(char* str); // 문자열 배열(포인터 배열)에 문자열(str)을 추가하는 함수
void printArr(void);
void sort(void);

char* strArr[10]; // 문자열을 저장할 배열
int pos = 0; // 문자열을 저장할 위치(배열 strArr의 index)

int main(void) {
   char input[50]; // 사용자가 입력한 단어를 저장할 배열

   printf("저장할 단어를 입력하세요.(ENTER:종료)\n>");

   // 엔터를 입력하면 gets()의 결과는 빈 문자열이 되고, strlen()의 결과는 0이 된다.
   while (strlen(gets(input))) { // while(strlen(gets(input))!=0) {
       add(input);
       printf(">");
   }

   printf("\n입력하신 단어는 다음과 같습니다.\n");
   sort();
   printArr();
   return 0;
}

void sort(void) {
   int i, j;
   char* tmp;

   for (i = 0; i < pos - 1; i++) {
       for (j = 0; j < pos - 1 - i; j++) {
           if (strcmp(strArr[j], strArr[j + 1]) > 0) {
               tmp = strArr[j];
               strArr[j] = strArr[j + 1];
               strArr[j + 1] = tmp;
```

```
            }
        }
    }
}

void add(char* str) {
    // 문자열(str)을 저장할 메모리를 동적 할당받음.(널 문자 저장할 공간 포함)
    char* tmp = malloc(sizeof(char)*(strlen(str) + 1));

    strcpy(tmp, str); // 사용자가 입력한 문자열을 동적 할당받은 메모리(tmp)에 복사
    strArr[pos++] = tmp; // tmp(문자열의 주소)를 배열에 저장하고 pos의 값을 1 증가
}

void printArr(void) {
    int i;
    printf("[");

    for (i = 0; i < pos; i++)
        printf("%s, ", strArr[i]);
    printf("]\n");
}
```

Quiz 8-8 예제 8-17을 malloc( )대신 calloc( )을 사용하도록 변경하시오.

[Quiz8-8][답]

[예제8-8]

```
#include<stdio.h>
#include<stdlib.h>
#include<string.h>  // strlen(), memcpy()

// 지정된 문자열(str)의 일부를 반환하는 함수
char* mySubstr(char* src, int fr, int to) { // fr <= x < to
   char* newStr;
   int   diff = to - fr;
   int   len;

   if (src == NULL) return src;

   len = strlen(src);

   if (fr > to || to > len || fr < 0) return NULL;

   // 문자열의 일부를 저장할 공간을 할당받음.(널 문자'\0'가 저장될 공간도 필요)
   newStr = (char*)calloc(sizeof(char), (diff + 1));
   memcpy(newStr, src + fr, diff); // src+fr부터 diff개의 문자를 newStr에 복사
// *(newStr + diff) = '\0';  // 이미 모두 0으로 초기화 되어 있어서 '\0'을 안넣어도 됨

   return newStr;
}

int main(void) {
   char str[] = "0123456789";

   printf("str=%s\n", str);
   printf("mySubstr(str, 3, 6)=%s\n", mySubstr(str, 3, 6));
   printf("mySubstr(str, 5, 10)=%s\n", mySubstr(str, 5, 10));
   printf("mySubstr(str, 6, 6)=%s\n", mySubstr(str, 6, 6));
   printf("mySubstr(str, 0, 11)=%s\n", mySubstr(str, 0, 11));
   printf("mySubstr(str, 7, 5)=%s\n", mySubstr(str, 7, 5));
   printf("mySubstr(str,-1, 5)=%s\n", mySubstr(str, -1, 5));

   return 0;
}
```

[실행결과]

```
str=0123456789
mySubstr(str, 3, 6)=345
mySubstr(str, 5, 10)=56789
mySubstr(str, 6, 6)=       ← 빈 문자열""
mySubstr(str, 0, 11)=(null)
mySubstr(str, 7, 5)=(null)
mySubstr(str,-1, 5)=(null)
```

Quiz 8-9 다음은 문자열을 대소문자 구별없이 오름차순으로 정렬하는데 사용할 함수이다. 이 함수의 내용을 완성해서 예제8-27에 추가하고 올바르게 동작하는지 테스트 하시오.

```
int cmpDicAscIgnoreCase(char* s1, char* s2) {
    /* 내용을 완성하시오 */
}
```

[Quiz8-9][답]

[예제8-9]

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

int cmpDicAsc(char*, char*);  // 사전순 오름차순
int cmpDicDsc(char*, char*);  // 사전순 내림차순
int cmpLenAsc(char*, char*);  // 문자열 길이 오름차순
int cmpLenDsc(char*, char*);  // 문자열 길이 내림차순
int cmpDicAscIgnoreCase(char* s1, char* s2); // 사전순 오름차순(대소문자 구분안함)

char* toUpperCase(char* str);
void printArr(char**, int);
void sort(char**, int, int(*)(char*, char*));

int main(void) {
   char* strArr[] = { "ABC", "aaa", "abb" , "Abc" };
   int len = sizeof(strArr) / sizeof(char*);

   int(*pf[])(char*, char*) = { cmpDicAsc, cmpDicDsc, cmpLenAsc, cmpLenDsc,
cmpDicAscIgnoreCase };
   int i;

   printArr(strArr, len);

   while (1) {
       printf("\n0.사전순 오름차순  \t");
       printf("1.사전순 내림차순    \n");
       printf("2.문자열 길이 오름차순 \t");
       printf("3.문자열 길이 내림차순 \n");
       printf("4.사전순 오름차순(대소문자 구분안함)\n\n");
       printf("정렬방법을 선택해주세요.>");

       scanf("%d", &i);

       if (!(0 <= i && i <= 4))
           break;

       sort(strArr, len, pf[i]);
       printArr(strArr, len);
   }

   return 0;
}
```

```
int cmpDicAsc(char* s1, char* s2) { return strcmp(s1, s2) > 0 ? 1 : 0; }
int cmpDicDsc(char* s1, char* s2) { return strcmp(s2, s1) > 0 ? 1 : 0; }
int cmpLenAsc(char* s1, char* s2) { return strlen(s1) > strlen(s2) ? 1 : 0; }
int cmpLenDsc(char* s1, char* s2) { return strlen(s2) > strlen(s1) ? 1 : 0; }
int cmpDicAscIgnoreCase(char* s1, char* s2) {
   return strcmp(toUpperCase(s1), toUpperCase(s2)) > 0 ? 1 : 0;
}

char* toUpperCase(char* str) {
   char* newStr = malloc((strlen(str) + 1));
   char* p = newStr;

   if (str == NULL) return str;

   while (*str) {
       if ('a' <= *str && *str <= 'z')
           *p++ = (char)(*str++ - 32);
       else
           *p++ = *str++;
   }

   *p = '\0';

   return newStr;
}


void sort(char** strArr, int len, int(*pf)(char*, char*)) {
   char* tmp;
   int i, j;

   for (i = 0; i<len - 1; i++)
       for (j = 0; j<len - 1 - i; j++)
           if (pf(strArr[j], strArr[j + 1])) {
               tmp = strArr[j];
               strArr[j] = strArr[j + 1];
               strArr[j + 1] = tmp;
           }
}

void printArr(char** strArr, int len) {
   int i;

   printf("[");
   for (i = 0; i<len; i++)
       printf("%s, ", strArr[i]);
   printf("]\n");
}
```

[실행결과]

```
[ABC, aaa, abb, Abc, ]

0.사전순 오름차순       1.사전순 내림차순
2.문자열 길이 오름차순  3.문자열 길이 내림차순
4.사전순 오름차순(대소문자 구분안함)
```

```
정렬방법을 선택해주세요.>0
[ABC, Abc, aaa, abb, ]

0.사전순 오름차순       1.사전순 내림차순
2.문자열 길이 오름차순  3.문자열 길이 내림차순
4.사전순 오름차순(대소문자 구분안함)

정렬방법을 선택해주세요.>4
[aaa, abb, ABC, Abc, ]

0.사전순 오름차순       1.사전순 내림차순
2.문자열 길이 오름차순  3.문자열 길이 내림차순
4.사전순 오름차순(대소문자 구분안함)

정렬방법을 선택해주세요.>
```

Quiz 8-10 함수 getFunc가 다음과 같이 정의되어 있을 때 이 함수를 가리키기 위한 함수 포인터 pf의 타입은?

```
void(*getFunc(void))(void) { return NULL; }

int main(void) {
    [ ① ]; // 함수 포인터 pf를 선언
    pf = getFunc;    // 함수 getFunc의 주소를 함수 포인터 pf에 저장
    pf(); // getFunc()를 호출
            ...
```

[Quiz8-10][답] void(*(*pf)(void))(void)
함수 이름 대신에 (*포인터이름)을 넣으면 된다.

```
void(*getFunc(void))(void)
        ⬇
void(*(*pf)(void))(void)
```

Quiz 9-1 예제9-9의 sortStuArr2( )가 qsort( )를 이용해서 정렬하도록 변경하시오.

[Quiz9-1][답]

[예제9-1]

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

typedef struct student {
   int   no;
   char  name[10];
   int   kor, math, eng;
   int   totalScore;
   float average;
} Student;

void calculateScore(Student* stuArr, const int LEN);
void sortStuArr(Student* stuArr, const int LEN);
void sortStuArr2(Student* pArr[]);
void printStuArr(const Student* stuArr, const int LEN);
void printStuArr2(const Student* pArr[]);
int compareStuNo(const void* v1, const void* v2);

int main(void) {
   Student stuArr[] = {
        { 1, "LEE",   90,  90,  90, },
        { 2, "KIM",  100, 100, 100, },
        { 3, "PARK",  80,  80,  80, },
        { 4, "HONG", 100, 100, 100, }
   };

   const int LEN = sizeof(stuArr) / sizeof(stuArr[0]);

   Student* pArr[] = { &stuArr[0], &stuArr[1], &stuArr[2], &stuArr[3], NULL };

   calculateScore(stuArr, LEN);   // 총점과 평균을 계산

   printf("= 총점 내림차순 정렬=\n");
   sortStuArr(stuArr, LEN);       // 구조체 배열을 정렬(총점 내림차순)
   printStuArr(stuArr, LEN);      // 구조체 배열을 출력

   printf("\n= 번호 내림차순 정렬 =\n");
   sortStuArr2(pArr);          // 구조체 포인터 배열을 정렬(번호 내림차순)
   printStuArr2(pArr);         // 구조체 포인터 배열을 출력

   return 0;
}

void sortStuArr(Student* stuArr, const int LEN) {
   int i, j;
   Student tmp;

   for (i = 0; i<LEN - 1; i++)
        for (j = 0; j<LEN - 1 - i; j++)
             if (stuArr[j].totalScore < stuArr[j + 1].totalScore) {
                  tmp = stuArr[j];
```

```
                stuArr[j] = stuArr[j + 1];
                stuArr[j + 1] = tmp;
            }
    }

// 정렬할 배열의 요소 타입은 Student*이고, 포인터 v1, v2가 각 요소를 가리키므로
Student**로 형변환해야 한다.
int compareStuNo(const void* v1, const void* v2) {
   Student* s1 = *(Student**)v1;
   Student* s2 = *(Student**)v2;

   return s2->no - s1->no;
}

void sortStuArr2(Student* pArr[]) {
   int i;

   if(pArr==NULL) return;

   for(i=0;pArr[i]!=NULL;i++); // pArr에 저장된 요소의 개수를 센다.

   qsort(pArr, i, sizeof(Student*), compareStuNo);
}

void calculateScore(Student* stuArr, const int LEN) {
   int i, sum;

   for (i = 0; i<LEN; i++) {
        sum = stuArr[i].kor + stuArr[i].math + stuArr[i].eng;
        stuArr[i].totalScore = sum;
        stuArr[i].average = sum / 3.0f;
   }
}

void printStu(const Student* p) {
   printf("%2d %10s %3d %3d %3d %3d %6.2f\n",
        p->no, p->name, p->kor, p->math, p->eng, p->totalScore, p->average);
}

void printStuArr(const Student* stuArr, const int LEN) {
   int i;
   for (i = 0; i<LEN; i++)
        printStu(stuArr + i);
}

void printStuArr2(const Student* pArr[]) {
   while (*pArr)
        printStu(*pArr++);
}
```

void 포인터 v1과 v2를 이용해서 배열의 각 요소에 접근하는 것이므로, 구조체 포인터 배열 pArr의 각 요소를 가리키는 포인터의 타입은 배열요소의 타입인 'Student*'에 '*'을 붙인 'Student**'이어야 맞다. 그래서 v1, v2를 'Student**'으로 형변환해서 다뤄야 올바른 결과를 얻을 수 있다.(void포인터로는 할 수 있는게 없으므로)

```
int compareStuNo(const void* v1, const void* v2) {
   Student* s1 = *(Student**)v1;
   Student* s2 = *(Student**)v2;

   return s2->no - s1->no;
}
```

아래와 같이 해도 되지만, 이렇게 하면 no를 읽기위해 구조체 전체를 복사해서 가져오는 것이므로 비효율적이다. 위의 코드 처럼, 포인터를 이용해서 값이 저장된 곳을 찾아가서 읽어오는 것이 더 효율적이다.

```
int compareStuNo(const void* v1, const void* v2) {
    Student s1 = **(Student**)v1; // 구조체 전체를 읽어서 s1으로 복사
    Student s2 = **(Student**)v2;

    return s2.no - s1.no;
}
```

Quiz 9-2 예제9-11에 새로운 데이터를 입력받아서 추가하는 insertData( )를 새로 작성하시오.

[Quiz9-2][답] 아래에 작성된 답안은 성적처리 프로그램의 기본적인 틀을 갖추어 작성되었다. 여러모로 미흡한 점이 많지만, 이 예제를 보완, 발전시켜보면 공부에 도움이 많이 될 것이다. scanf()로 입력받는 것이 여러모로 불편하다. 10장과 11장에서 보다 다양한 입력 함수들을 배울 것이니 너무 개의치 않길 바란다.

[예제9-]

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

typedef struct student {
   int   no;
   char  name[10];
   int   kor, math, eng;
   int   totalScore;
   float average;
} Student;

void init(void);
int showMenu(void);

int  deleteStu(Student* pArr[], int no);
void printStuArr(const Student* pArr[]);
void inputData(Student* pArr[]);
void deleteData(Student* pArr[]);

Student* pArr[100];
int size; // 포인터 배열에 저장된 데이터의 개수

int main(void) {
   int menu;

   init();

   while(menu=showMenu()) {
        switch (menu) {
             case 1:
                  printStuArr(pArr);
                  break;
             case 2:
                  inputData(pArr);
                  break;
             case 3:
                  deleteData(pArr);
                  break;
             default:
                  printStuArr(pArr);
        }
   }

   return 0;
}
```

```
void deleteData(Student* pArr[]) {
   int no, result;

   printStuArr(pArr);

   while(1) {
        printf("삭제하려는 데이터의 번호(no)를 선택하세요(이전 메뉴:0)>");
        scanf("%d", &no);

        if(no==0) return;

        result = deleteStu(pArr, no);

        if(result==1) {
            printf("정상적으로 삭제 되었습니다.\n");
        } else {
            printf("삭제되지 않았습니다. 번호(no)를 다시 입력해주세요.\n");
        }
        printStuArr(pArr);
   }
}

int deleteStu(Student* pArr[], int no) {
   int i;
   for (i = 0; pArr[i]; i++)
        if (pArr[i]->no == no) {
            free(pArr[i]);
            memmove(&pArr[i], &pArr[i + 1], sizeof(Student*) * (size - i - 1));
            pArr[--size] = NULL;

            return 1;
        }
   return 0;
}

int showMenu(void) {
   int menu;

   printf("[성적처리 프로그램]\n");
   printf("1. 학생성적 목록 보기\n");
   printf("2. 학생성적 입력\n");
   printf("3. 학생성적 삭제\n");
   printf("0. 종료\n");
   puts("");

   printf("원하는 메뉴를 선택하세요>");
   scanf("%d", &menu);

   return menu;
}

void init(void) {
   Student stuArr[] = {
        { 1, "LEE", 100, 100, 100, 300, 100.0f },
        { 2, "KIM", 90, 90, 90, 270, 90.0f },
        { 3, "PARK", 80, 80, 80, 240, 80.0f },
```

```
        { 4, "CHOI", 100, 100, 100, 300, 100.0f }
    };

    const int LEN = sizeof(stuArr) / sizeof(stuArr[0]);
    int i, no;

    for (i = 0; i < LEN; i++) {
        pArr[i] = malloc(sizeof(Student)); // 구조체 변수를 동적할당
        *pArr[i] = stuArr[i]; // 구조 배열의 요소를 복사
    }

    size = LEN;
}

void printStudent(const Student* p) {
    printf("%2d %-10s %4d %4d %4d %4d %6.2f\n",
        p->no, p->name, p->kor, p->math, p->eng,
        p->totalScore, p->average);
}

void printStuArr(const Student* pArr[]) {
    printf("번호 이름 국어 수학 영어 총점 평균\n");
    printf("=========================================\n");

    while (*pArr)
        printStudent(*pArr++);
    puts("");
}

void inputData(Student* pArr[]) {
    Student* p;
    int n;

    printf("학생 정보를 입력하세요.(번호, 이름, 국어, 수학, 영어 순서로)\n");
    printf("입력을 모두 마친 후에는 q를 입력하고 Enter키를 누르세요.\n");
    while (1) {
        p = malloc(sizeof(Student)); // 구조체 변수를 동적할당 받음

        printf(">");
        n = scanf("%d%s%d%d%d", &p->no, p->name, &p->kor, &p->math, &p->eng);

        if (n != 5) { // 읽어들인 값의 개수가 5가 아니면 반복 종료
            free(p);
            while(getchar()!='\n'); // 잘못 입력된 내용을 버퍼에서 읽어서 없앤다.

            break;
        }

        p->totalScore = p->kor + p->math + p->eng;
        p->average    = p->totalScore / 3.0f;

        pArr[size++] = p; // 동적 할당 받은 구조체 변수의 주소를 포인터 배열에 저장
    }

    printStuArr(pArr);
}
```

Quiz 10-1 getchar( )대신 getche( )를 사용하도록 예제10-1을 변경하시오.

[Quiz10-1][답] getchar()과 달리 getche()는 입력받을 때 버퍼를 사용하지 않으므로 버퍼를 비워줄 필요가 없다. 그리고 Enter키를 누르지 않아도 바로 입력된다.

[예제10-1]

```
#include <stdio.h>

int main(void) {
   int ch;

   do {
        printf("문자 하나를 입력해주세요.(종료:x)>");
        ch = getche();
        printf("\n");

        printf("입력하신 문자는 '%c'입니다.\n", ch);

//      while (getchar() != '\n'); // 입력 버퍼를 비운다.
   } while (ch != 'x' && ch != 'X');      // 'x'또는 'X'를 입력하면 반복 종료

   printf("'%c'를 입력하셔서 종료되었습니다.\n", ch);

   return 0;
}
```

Quiz 10-2 예제10-2를 3개의 숫자만 입력할 수 있도록 변경하시오.

[Quiz10-2][답]

[예제10-2]

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

int main(void) {
   char input[50] = {0}; // 입력을 저장할 배열
   int pos = 0;
   int ch;

   printf("숫자만 입력하세요. >");

   while(1) {
        ch = getch();

        if(ch=='\r') {   // '\r'==13   엔터를 입력하면 13이 저장됨
            puts("");    // new line
            break;
        }

        if('1'<=ch && ch<='9' && pos <= 2) {
            putch(ch);
            input[pos++] = ch;
        } else if(ch=='\b' && pos!=0) {  // backspace를 눌렀을 때
          // 마지막 글자를 지운다. backspace, 공백, backspace를 연속으로 출력
            putch('\b');  //  '\b'==8
            putch(' ');
            putch('\b');
            input[--pos] = '\0'; // 마지막 문자를 널 문자로 변경
        }
   }

   printf("input:%s\n", input);
   return 0;
}
```

Quiz 10-3 ①과 ②에 알맞은 코드를 넣어 예제를 완성하시오.

▼ 예제 10-5/ch10/10_5.c

```
#include <stdio.h>

typedef struct { int hour, min, sec; } myTime;

int timeToSec(myTime t)    { /* ① 알맞은 코드를 넣어 완성하시오. */ }
myTime secToTime(int sec) { /* ② 알맞은 코드를 넣어 완성하시오. */ }

int main(void) {
    myTime t = secToTime(45296);

    printf("%d시간 %d분 %d초\n", t.hour, t.min, t.sec);
    printf("%d초\n", timeToSec(t));
    return 0;
}
```

▼ 실행결과

```
12시간 34분 56초
45296초
```

[Quiz10-3][답]

[예제10-3]

```
#include <stdio.h>

typedef struct { int hour, min, sec; } myTime;

int timeToSec(myTime t) { return t.hour * 3600 + t.min * 60 + t.sec; }

myTime secToTime(int sec) {
    myTime t;

    t.hour = sec / 3600;
    sec -= t.hour * 3600;

    t.min = sec / 60;
    t.sec = sec % 60;

    return t;
}

int main(void) {
    myTime t = secToTime(45296);

    printf("%d시간 %d분 %d초\n", t.hour, t.min, t.sec);
    printf("%d초\n", timeToSec(t));
    return 0;
}
```

Quiz 10-4 예제10-15를 변경하여 아래와 같이 그래프가 출력되도록 하시오.

▼ 실행결과

```
A=22
A=26
B=26
C=26
== sum ==
A=48################################################
B=26##########################
C=26##########################
```

[Quiz10-4][답]

[예제10-4]

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

int main(void) {
   char data[] = { 'A', 'A', 'B', 'C' };
   int count[sizeof(data) / sizeof(data[0])] = { 0 };

   char dArr[(sizeof(data) / sizeof(data[0])) + 1] = { 0 };

   const int LEN = sizeof(data) / sizeof(data[0]);
   int i, j;
   int pos = 0;

   srand((unsigned)time(NULL));

   for (i = 0; i<100; i++) {
       count[rand() % LEN]++;
   }

   for (i = 0; i<LEN; i++) {
       printf("%c=%d\n", data[i], count[i]);
   }

   printf("== sum ==\n");

   for (i = 0; i<LEN; i++) {
       if (i == 0)
           dArr[pos++] = data[i];
       else {
           for (j = 0; j<pos && dArr[j] != data[i]; j++);

           if (j == pos)
               dArr[pos++] = data[i];
       } // if-else
   }

   for (i = 0; dArr[i]; i++) {
       int sum = 0;
       for (j = 0; j<LEN; j++) {
```

```
            if (dArr[i] == data[j])
                sum += count[j];
        }
        printf("%c=\%d", dArr[i], sum);
        for (j = 0; j<sum; j++) {
            printf("#");
        }
        printf("\n");
    }

    return 0;
}
```

Quiz 10-5 함수 attack은 지정된 확률(%)로 1을 반환하고 그 외에는 0을 반환한다. 예를 들어 'attack(20)'과 같이 호출했을 때, 이 함수가 1을 반환할 확률은 약 20%이다. 주석대신 알맞은 식을 넣어서 이 함수를 완성하시오.

▼ 예제 10-17/ch10/10_17.c

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

int attack(int percent) {
    return ( /* 알맞은 식을 넣으시오. */ ) ? 1 : 0;
}

int main(void) {
    int i;

    srand((unsigned)time(NULL));

    for (i=0;i<10;i++)
        printf("%d", attack(25));
    puts("");

    return 0;
}
```

▼ 실행결과

```
1001100000
```

[예제10-5]

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

int attack(int percent) {
   return (rand() % 100  < percent) ? 1 : 0;
}

int main(void) {
   int i;

   srand((unsigned)time(NULL));  // 난수의 씨앗값(seed)를 현재시간으로 초기화

   for (i = 0; i<10; i++) {
       if (attack(25))
           printf("성공!!!\n");
       else
           printf("실패\n");
   }

   return 0;
}
```

Quiz 10-6 예제10-20을 변경하여, 문자열 상수에 포함된 숫자 중에서 가장 큰 숫자의 자리수를 함께 출력하시오.

▼ 실행결과

```
str=abcd999xyz3456oo2z
cnt=3
max_length=4
```

[Quiz10-6][답]

[예제10-6]

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

int main(void) {
   char* str = "abc999xyz3456oo2z";
   char* numStr = "0123456789";
   int cnt = 0;
   int len = 0, maxLen = 0;

   printf("str=%s\n", str);

   while (str && *str) {   // while(str!=NULL && *str!='\0') {
        str = strpbrk(str, numStr); // str에 포함된 첫 번째 숫자의 주소를 반환

        if (str) { // if(str!=NULL)) { // str이 NULL이 아니면,
            cnt++;
            len = strspn(str, numStr);

            if (len > maxLen)
                maxLen = len;

            str += len; // 숫자의 길이만큼 str을 증가시킨다.
        }
   }

   printf("cnt=%d\n", cnt);
   printf("max_length=%d\n", maxLen);
   return 0;
}
```

Quiz 10-7 strtol( )를 이용해서 2진법으로 표현된 문자열 "100101011"을 10진수로 변환한 결과를 출력하시오.

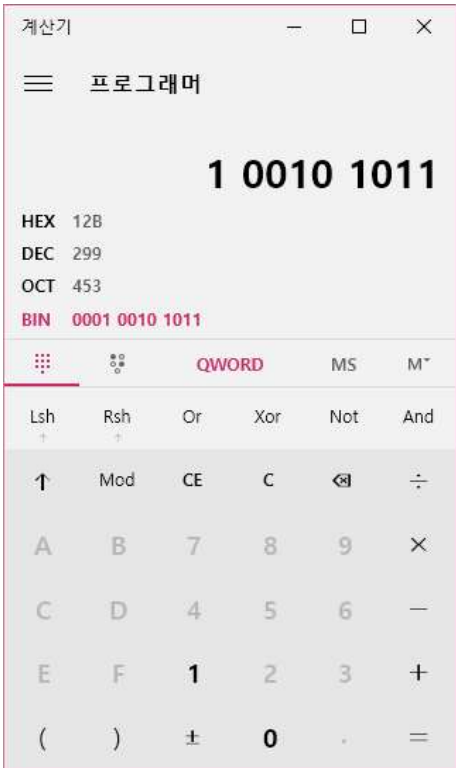
[Quiz10-7][답]

[예제10-7]

```
#include <stdio.h>

int main(void) {
   char* str = "100101011";
   char* ptr;

   long l = strtol(str, &ptr, 2); // "100101011"을 10진수로 변환
   printf("%s -> %d\n ", str, l);

   return 0;
}
```

[실행결과]

```
100101011 -> 299
```



Quiz 10-8 특정 문자열의 n번째 문자의 다음에 지정된 문자열을 삽입하는 함수 strinsert( )를 완성하시오.

[Quiz10-8][답] 문자열에 문자열을 삽입하는 것은 기존의 공간이 부족하거나 문자열 상수의 경우에는 삽입할 수 없으므로, 동적 할당으로 새로운 문자열을 생성해서 삽입하는 것이 좋다.

[예제10-8]

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

// src의 n번째 index에 dst를 삽입한다.
char* strinsert(char* dst, char* src, int n) {
   int len1 = strlen(src);
   int len2 = strlen(dst);
   char* newStr;

   if (src == NULL || dst == NULL || n > len1) return src;

   newStr = calloc(len1 + len2 + 1, sizeof(char)); // 결과를 저장할 공간을 동적할당 받는다.(널 문자 필요)

   strcpy(newStr, src);  // 기존의 내용을 동적할당 받은 공간에 복사

   // n번째 위치 이후의 문자열들을 len2만큼 뒤로 이동해서 빈 공간을 확보
   memmove(newStr + n + len2, newStr + n, len1 - n);
   memcpy(newStr + n, dst, len2); // 확보된 빈 공간에 삽입할 문자열을 복사

   return newStr;
}

int main(void) {
   char str[10] = "abcd";

   printf("%s\n", strinsert("123", str, 1));
   printf("%s\n", strinsert("123", str, 2));
   printf("%s\n", strinsert("123", str, 3));

   return 0;
}
```

[실행결과]

```
a123bcd
ab123cd
abc123d
```

Quiz 10-9 세 개의 변수 중에서 중간값을 구하는 매크로 함수 mid를 완성하시오.
[Hint] 매크로 함수 max와 min, 그리고 조건 연산자를 사용하시오.

[Quiz10-9][답]
1. 모든 값을 더한 다음에 최대값과 최소값을 빼는 방법
   (x+y+z) - max(x, max(y,z)) - min(x, min(y,z))

2. x가 최대값(max(x, max(y,z))이면, y와 z중에 큰 것이 중간값
   x가 최소값(min(x, min(y,z))이면, y와 z중에 작은 것이 중간값
   x가 최대값도, 최소값도 아니면 x가 중간값
   x==max(x,max(y,z)) ? max(y, z) : (x==min(x, min(y,z))? min(y, z) : x)

[예제10-9]
```
#include <stdio.h>

#define max(x, y)       ((x) > (y) ? (x) : (y))
#define min(x, y)       ((x) < (y) ? (x) : (y))
#define mid(x, y, z)    ((x+y+z) - max(x, max(y,z)) - min(x, min(y,z)))

int main(void) {
   printf("mid(1, 2, 3) = %d\n", mid(1, 2, 3));
   printf("mid(1, 1, 1) = %d\n", mid(1, 1, 1));
   printf("mid(1, 2, 2) = %d\n", mid(1, 2, 2));
   printf("mid(1, 1, 2) = %d\n", mid(1, 1, 2));
   printf("mid(3, 2, 1) = %d\n", mid(3, 2, 1));
   printf("mid(2, 1, 3) = %d\n", mid(2, 1, 3));
   printf("mid(2, 3, 1) = %d\n", mid(2, 3, 1));

   return 0;
}
```

[실행결과]
```
mid(1, 2, 3) = 2
mid(1, 1, 1) = 1
mid(1, 2, 2) = 2
mid(1, 1, 2) = 1
mid(3, 2, 1) = 2
mid(2, 1, 3) = 2
mid(2, 3, 1) = 2
```

Quiz 11-1 만일 아래와 같은 문장이 실행되면, 파일에 출력되는 내용은 무엇일까?

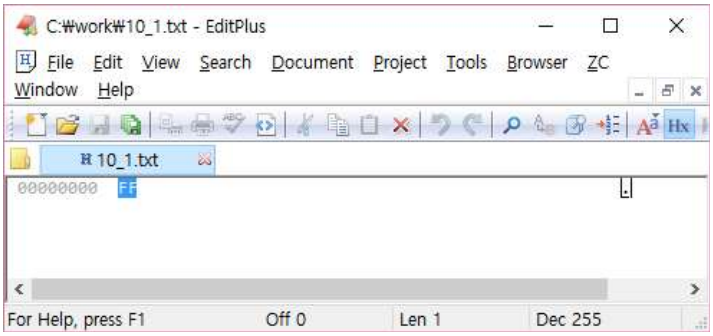
```
fputc(-1, fp);    // 파일에 -1을 출력?
```

[Quiz11-1][답] 0xFF가 출력된다.

[예제11-1]

```
#include <stdio.h>

int main(void) {
   FILE* fp = fopen("c:\\work\\10_1.txt", "w");

   fputc(-1, fp);

   return 0;
}
```

위의 예제를 실행한 후에 'c:₩work₩10_1.txt'를 열어보면, 아무 내용도 보이지 않을 것이다. 파일의 내용을 16진수로 볼 수 있는 에디터로 보면 아래와 같이 0xFF가 출력된 것을 확인할 수 있다.



-1은 int타입(4 byte)이며, 16진수로 표현하면 0xFFFFFFFF이다.

```
    fputc(-1, fp);
 → fputc(0xFFFFFFFF, fp);
```

fputc()의 첫 번째 매개변수의 타입이 int이지만, char타입의 데이터를 제공받기 위한 것이므로, 첫 번째 매개변수의 전체 4 byte중에서 마지막 1 byte만 출력한다. 그래서 0xFFFFFFFF중에서 마지막 1 byte인 0xFF만 출력된 것이다. 0xFF는 10진수로 255인데, 이에 해당하는 문자가 없기 때문에 화면에 아무 것도 나타나지 않는 것이다.

Quiz 11-2 예제11-2에서 아래와 같이 원본 파일과 복사본 파일이 같을 경우 어떤 결과를 얻게 될지 한번 생각해 보자.

```
char* fname1 = "c:\\work\\ch11\\aaa.txt";
char* fname2 = "c:\\work\\ch11\\aaa.txt";
```

[Quiz11-2][답] 예제11-2에서 알 수 있듯이 fname1은 읽기모드로 파일을 열고, fnmae2는 쓰기모드로 파일을 연다. 쓰기모드는 해당 파일의 기존 내용을 모두 삭제하기 때문에, aaa.txt의 내용은 읽기도 전에 모두 지워진다.

```
char* fname1 = "c:₩₩work₩₩ch11₩₩aaa.txt"
char* fname2 = "c:₩₩work₩₩ch11₩₩aaa.txt"

FILE* in_f = fopen(fname1, "r");
FILE* out_f = fopen(fname2, "w");  // aaa.txt의 내용을 모두 삭제한다.
```

예제11-2를 가지고 테스트해보면 알겠지만, aaa.txt의 내용이 모두 지워지며 아무런 내용도 복사되지 않는다.

Quiz 11-3 예제11-6을 변경하여 화면에서 행 단위로 성적 데이터를 입력받아서 파일에 저장한 후, 이 파일로부터 성적 데이터를 읽어서 총점과 평균을 계산한 결과를 화면에 동시에 출력하도록 하시오.

▼ 실행결과

```
성적 데이터를 학번, 이름, 국어, 수학, 영어 순으로 입력하세요.(종료:Enter)
>1 aaa 100 100 100
>2 bbb 90 90 90
>3 ccc 80 80 70
>

no  name      kor math eng total average
=========================================
 1 aaa        100 100 100  300   100.00
 2 bbb         90  90  90  270    90.00
 3 ccc         80  80  70  230    76.67
```

[Quiz11-3][답]

[예제11-3]

```
#include <stdio.h>

typedef struct student {
   int no;
   char name[10];
   int kor;
   int math;
   int eng;
   int totalScore;
   float average;
} Student;

int main(void) {
   char* fname = "c:\\work\\scoredata.txt";
   FILE* in_f, *out_f;
   Student s;

   int i=0;
   char input[50];

   char* inFmt     = "%d  %s    %d  %d  %d";
   char* outFmt    = "%2d %-10s %3d %3d %3d\n";
   char* stdoutFmt = "%2d %-10s %3d %3d %3d %4d %8.2f\n";

   if ((out_f = fopen(fname, "wt")) == NULL) {
       printf("파일[%s]을 열 수 없습니다.\n", fname);
       return 1;
   }

   printf("성적 데이터를 학번, 이름, 국어, 수학, 영어 순으로 입력하세요.(종
료:Enter)\n");
   printf(">");

   while (strlen(gets(input))) {  // while(strlen(gets(input))!=0) {
       sscanf(input, inFmt, &s.no, &s.name, &s.kor, &s.math, &s.eng);
```

```
        fprintf(out_f, outFmt, s.no, s.name, s.kor, s.math, s.eng);

        printf(">");
    }

    fclose(out_f);

    if ((in_f = fopen(fname, "rt")) == NULL) {
        printf("파일[%s]을 열 수 없습니다.\n", fname);
        exit(1);
    }

    printf("\nno  name     kor math eng total average \n");
    printf("=========================================\n");

    while (1) {
        i = fscanf(in_f, inFmt, &s.no, &s.name, &s.kor, &s.math, &s.eng);
        if(i<=0) break;
        s.totalScore = s.kor + s.math + s.eng;
        s.average = s.totalScore / 3.0f;

        fprintf(stdout, stdoutFmt, s.no, s.name, s.kor, s.math, s.eng,
            s.totalScore, s.average);
    }

    fclose(in_f);

    return 0;
}
```

Quiz 11-4 fgets( )를 호출할 때 사용하는 char배열의 크기를 4096으로 하지 않는 이유는 무엇일까?

[Quiz11-4][답] fgets()와 같이 f로 시작하는 파일 입출력함수들은 버퍼를 이용해서 입출력을 하는데, 이 버퍼의 크기가 보통 4096 byte(=4 KB)이다. fgets()는 별도의 char배열을 이용해서 입출력을 하는데, 이 char배열의 크기가 버퍼의 크기와 일치하면 가장 효율이 좋을 것이다. 그러나 fgets()는 char배열이 가득차거나 개행문자 '₩n'을 만날 때까지만 읽기 때문에, char배열의 크기가 한 행의 최대길이+1(널 문자) 정도면 충분하다.

대부분의 경우 한 행의 최대길이는 255를 넘지 않으므로 char배열의 길이 역시 255정도면 된다.

Quiz 11-5 이진 파일을 텍스트 모드로 읽어서 복사할 때 발생할 수 있는 문제들에 대해 설명하시오.

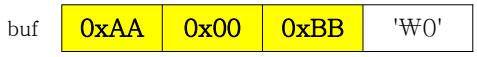
[Quiz11-5][답]

텍스트 모드로 출력한 파일은 텍스트 파일(text file)이 되며, 이진 모드로 출력한 파일은 이진 파일(binary file)이 된다. 그리고 파일을 읽을 때는 파일의 종류에 맞는 모드로 파일을 열어야 한다. 이진 모드는 텍스트 파일이건 이진 파일이건 변환없이 그대로 읽고 쓰기 때문에 별 문제가 없지만, **이진 파일을 텍스트 모드로 읽으면, 숫자를 널문자나 개행문자로 인식하기 때문에 문제가 발생한다.**

아래와 같이 텍스트 모드에서 fgets()를 통해 읽어온 데이터 3 byte에 0x00이 포함되어 있으면 fputs()는 0x00을 널 문자로 판단하고 이후의 내용을 출력하지 않는다. 즉, 읽어온 3 byte의 데이터 '0xAA00BB'중에서 '0xAA'만 출력되고 '0x00'과 '0xBB'는 출력되지 않는다.

[참고] 널 문자의 문자 코드는 0이므로, '₩0'과 0x00는 같은 값이다.

fputs(buf, fp);



또 다른 문제는 앞 단원에서 배운 것처럼 윈도우에서는 문자 코드 '0x1A'를 입력의 끝을 의미하는 용도로 쓰기 때문에 발생한다. **파일에 '0x1A'가 포함되어 있는 경우**, 텍스트 모드에서 이 파일을 fgetc()나 fgets()로 읽다가 이 문자를 만나면, **파일의 끝인 줄 알고 읽기를 멈춘다.** 즉, 이진 파일을 텍스트 모드로 열어서 복사하면, 파일의 일부만 복사될 수 있다는 뜻이다.

마지막으로 **0x0D와 0x0A가 연속해서 나오면₩r₩n로 인식하여 ₩n으로 바꿔서 출력한다**는 문제가 있다.(윈도우즈에서만 발생.)