[연습문제 - 모범답안]

[5-1] 다음은 배열을 선언하거나 초기화 한 것이다. 잘못된 것을 고르고 그 이유를 설명하시오.

a. int[] arr[];
b. int[] arr = {1,2,3,}; // 마지막의 쉼표 ',' 는 있어도 상관없음.
c. int[] arr = new int[5];
d. int[] arr = new int[5]{1,2,3,4,5}; // 두 번째 대괄호[]에 숫자 넣으면 안됨.
e. int arr[5]; // 배열을 선언할 때는 배열의 크기를 지정할 수 없음.
f. int[] arr[] = new int[3][];

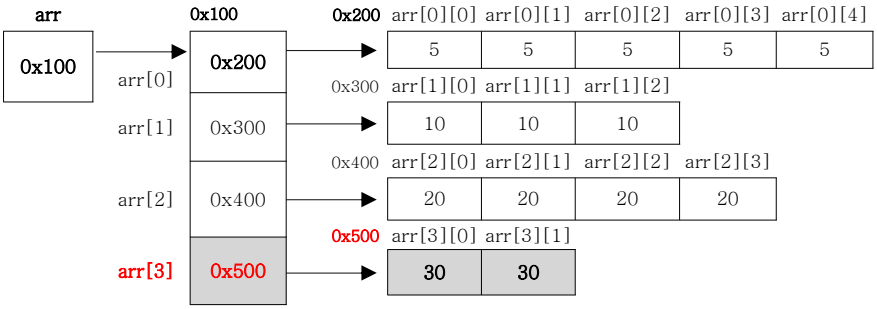
[정답] d, e
[해설] d. int[] arr = new int[]{1,2,3,4,5}에서는 대괄호[]안에 배열의 크기를 지정할 수 없다. 괄호{}안의 데이터의 개수에 따라 자동적으로 결정되기 때문이다.

[5-2] 다음과 같은 배열이 있을 때, arr[3].length의 값은 얼마인가?

```
int[][] arr = {
        {  5, 5, 5, 5, 5},
        { 10, 10, 10},
        { 20, 20, 20, 20},
        { 30, 30}
};
```

[정답] 2

[해설] 위와 같은 코드가 실행되면 다음과 같은 그림의 배열이 생성된다.



arr[3].length는 arr[3]이 가리키는 배열의 크기를 의미한다. 위의 그림에서 arr[3]이 가리키는 배열은 0x500번지에 있는 배열이며 크기는 2이다. 그래서 arr[3].length의 값은 2가 된다. 참고로 arr.length의 값은 4이고, arr[0].length의 값은 5, arr[1].length의 값은 3, arr[2].length의 값은 4이다.

[5-3] 배열 arr에 담긴 모든 값을 더하는 프로그램을 완성하시오.

[연습문제]/ch5/Exercise5_3.java

```
class Exercise5_3
{
     public static void main(String[] args)
     {
       int[] arr = {10, 20, 30, 40, 50};
       int sum = 0;

       for(int i=0;i<arr.length;i++) {
           sum += arr[i];
       }

       System.out.println("sum="+sum);
   }
}
```

[실행결과]

```
sum=150
```

[정답]

```
for(int i=0;i<arr.length;i++) {
    sum += arr[i];
}
```

[해설] 간단한 문제라서 별도의 설명은 생략함.

[5-4] 2차원 배열 arr에 담긴 모든 값의 총합과 평균을 구하는 프로그램을 완성하시오.

[연습문제]/ch5/Exercise5_4.java

```
class Exercise5_4
{
      public static void main(String[] args)
      {
        int[][] arr = {
             { 5, 5, 5, 5, 5},
             {10,10,10,10,10},
             {20,20,20,20,20},
             {30,30,30,30,30}
        };

        int total = 0;
        float average = 0;

        for(int i=0; i < arr.length;i++) {
             for(int j=0; j < arr[i].length;j++) {
                  total += arr[i][j];
             }
        }

        average = total /(float)(arr.length * arr[0].length);
        System.out.println("total="+total);
        System.out.println("average="+average);
    } // end of main
} // end of class
```

[실행결과]

```
total=325
average=16.25
```

[해설] 이번에도 배열과 반복문을 이용하는 문제인데, 2차원 배열이라 2중 for문을 사용해야한다는 것을 제외하고는 이전 문제와 다르지 않다.

평균을 구할 때는 배열의 모든 요소의 총합을 개수로 나누면 되는데, int로 나누면 int나누기 int이기 때문에 결과를 int로 얻으므로 소수점 이하의 값을 얻을 수 없다. 그래서 나누는 값을 float로 형변환 해주었다. 만일 float로 형변환을 해주지 않으면 avaerage는 16.25가 아닌 16.00이 될 것이다.(average의 타입이 float이므로 16을 저장하면 16.00이 된다.)

1. int형(4 byte)보다 크기가 작은 자료형은 int형으로 형변환 후에 연산을 수행한다.
   byte / short → int / int → int
2. 두 개의 피연산자 중 자료형의 표현범위가 큰 쪽에 맞춰서 형변환 된 후 연산을 수행한다.
   int / float → float / float → float
3. 정수형 간의 나눗셈에서 0 으로 나누는 것은 금지되어 있다.

[5-5] 다음은 1과 9사이의 중복되지 않은 숫자로 이루어진 3자리 숫자를 만들어내는 프로그램이다. (1)~(2)에 알맞은 코드를 넣어서 프로그램을 완성하시오.

[참고] Math.random()을 사용했기 때문에 실행결과와 다를 수 있다.

[연습문제]/ch5/Exercise5_5.java

```
class Exercise5_5 {
  public static void main(String[] args) {
      int[] ballArr = {1,2,3,4,5,6,7,8,9};
      int[] ball3   = new int[3];

      // 배열 ballArr의 임의의 요소를 골라서 위치를 바꾼다.
      for(int i=0; i< ballArr.length;i++) {
          int j = (int)(Math.random() * ballArr.length);
          int tmp = 0;

          tmp = ballArr[i];
          ballArr[i] = ballArr[j];
          ballArr[j] = tmp;
      }

      // 배열 ballArr의 앞에서 3개의 수를 배열 ball3로 복사한다.
      System.arraycopy(ballArr,0, ball3,0,3);

      for(int i=0;i<ball3.length;i++) {
          System.out.print(ball3[i]);
      }
      System.out.println();
  } // end of main
} // end of class
```

[실행결과]

```
486
```

[정답]

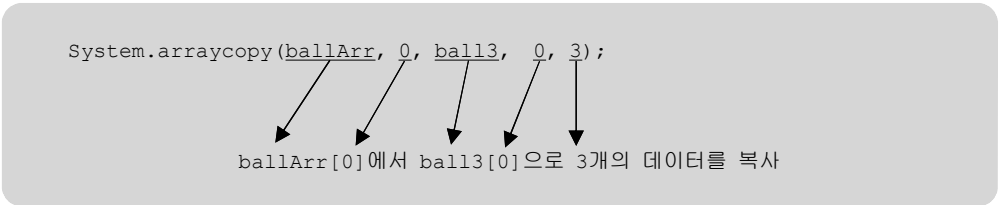
(1)

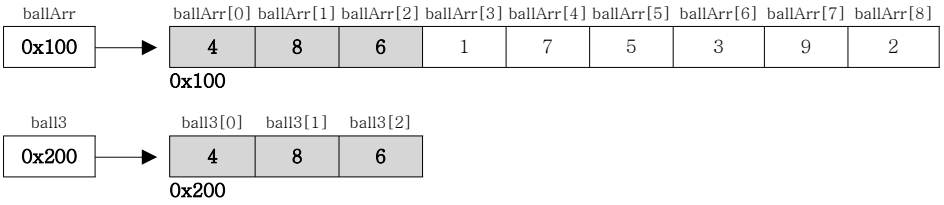
```
tmp = ballArr[i];
ballArr[i] = ballArr[j];
ballArr[j] = tmp;
```

(2)

```
System.arraycopy(ballArr,0, ball3,0,3);
```

[해설] 1~9의 숫자를 배열에 순서대로 담고, 반복해서 위치를 서로 바꿈으로써 숫자를 섞는다. 그 다음에 배열의 세 요소를 차례대로 가져오면 중복되지 않은 세 개의 정수를 얻을 수 있다. 그 다음엔 배열 ballArr에서 세 개의 값을 배열 ball3으로 복사한다. 편의상 맨 앞의 세 값을 ball3로 복사하기로 하자.




ballArr
0x100
ballArr[0] ballArr[1] ballArr[2] ballArr[3] ballArr[4] ballArr[5] ballArr[6] ballArr[7] ballArr[8]
4 8 6 1 7 5 3 9 2
0x100
ball3
0x200
ball3[0] ball3[1] ball3[2]
4 8 6
0x200

[5-6] 다음은 거스름돈을 몇 개의 동전으로 지불할 수 있는지를 계산하는 문제이다. 변수 money의 금액을 동전으로 바꾸었을 때 각각 몇 개의 동전이 필요한지 계산해서 출력하라. 단, 가능한 한 적은 수의 동전으로 거슬러 주어야한다. (1)에 알맞은 코드를 넣어서 프로그램을 완성하시오.

[Hint] 나눗셈 연산자와 나머지 연산자를 사용해야 한다.

[연습문제]/ch5/Exercise5_6.java

```
class Exercise5_6 {
   public static void main(String args[]) {
        // 큰 금액의 동전을 우선적으로 거슬러 줘야한다.
        int[] coinUnit = {500, 100, 50, 10};

        int money = 2680;
        System.out.println("money="+money);

        for(int i=0;i<coinUnit.length;i++) {
            System.out.println(coinUnit[i]+"원: "+money/coinUnit[i]);
            money = money%coinUnit[i];
        }
   } // main
}
```

[실행결과]

```
money=2680
500원: 5
100원: 1
50원: 1
10원: 3
```

[정답]

```
System.out.println(coinUnit[i]+"원: "+money/coinUnit[i]);
money = money%coinUnit[i];
```

[해설] 동전의 단위를 내림차순으로 배열에 초기화한다. 금액이 큰 동전을 우선적으로 지불해야 가장 적은 동전의 개수로 거스름돈을 줄 수 있기 때문이다. 그렇지 않으면, 모든 거스름돈을 10원짜리로만 주게 될 수도 있다.

변수 money를 coinUnit[i]로 나누면 지불할 동전의 개수가 되고, 나머지 연산을 하면 coinUnit[i]로 지불하고 남은 금액이 된다. 동전단위(coinUnit배열)의 개수만큼 이 과정을 반복하면 된다.

money	coinUnit[i]	money/coinUnit[i]	money%coinUnit[i]
2680	500	5	180
180	100	1	80
80	50	1	30
30	10	3	0

[5-7] 문제 5-6에 동전의 개수를 추가한 프로그램이다. 커맨드라인으로부터 거슬러 줄 금액을 입력받아 계산한다. 보유한 동전의 개수로 거스름돈을 지불할 수 없으면, '거스름돈이 부족합니다.'라고 출력하고 종료한다. 지불할 돈이 충분히 있으면, 거스름돈을 지불한 만큼 가진 돈에서 빼고 남은 동전의 개수를 화면에 출력한다. (1)에 알맞은 코드를 넣어서 프로그램을 완성하시오.

[연습문제]/ch5/Exercise5_7.java

```
class Exercise5_7
{
   public static void main(String args[])
  {
       if(args.length!=1) {
           System.out.println("USAGE: java Exercise5_7 3120");
           System.exit(0);
       }

      // 문자열을 숫자로 변환한다. 입력한 값이 숫자가 아닐 경우 예외가 발생한다.
       int money = Integer.parseInt(args[0]);

       System.out.println("money="+money);

       int[] coinUnit = {500, 100, 50, 10 }; // 동전의 단위
       int[] coin     = {5, 5, 5, 5};        // 단위별 동전의 개수


       for(int i=0;i<coinUnit.length;i++) {
           int coinNum = 0;

           // 1. 금액(money)을 동전단위로 나눠서 필요한 동전의 개수(coinNum)를 구한다.
           coinNum = money/coinUnit[i];

           // 2. 배열 coin에서 coinNum만큼의 동전을 뺀다.
           //   (만일 충분한 동전이 없다면 배열 coin에 있는 만큼만 뺀다.)
           if(coin[i] >= coinNum) {
               coin[i] -= coinNum;
           } else {
               coinNum = coin[i];
               coin[i] = 0;
           }

           // 3. 금액에서 동전의 개수(coinNum)와 동전단위를 곱한 값을 뺀다.
           money -= coinNum*coinUnit[i];

           System.out.println(coinUnit[i]+"원: "+coinNum);
       }

       if(money > 0) {
           System.out.println("거스름돈이 부족합니다.");
           System.exit(0);  // 프로그램을 종료한다.
       }

       System.out.println("=남은 동전의 개수 =");

       for(int i=0;i<coinUnit.length;i++) {
```

```
            System.out.println(coinUnit[i]+"원:"+coin[i]);
        }
    } // main
}
```

[실행결과]

```
C:\jdk1.8\work\ch5>java Exercise5_7
USAGE: java Exercise5_7 3120

C:\jdk1.8\work\ch5>java Exercise5_7 3170
money=3170
500원: 5
100원: 5
50원: 3
10원: 2
=남은 동전의 개수 =
500원:0
100원:0
50원:2
10원:3

C:\jdk1.8\work\ch5>java Exercise5_7 3510
money=3510
500원: 5
100원: 5
50원: 5
10원: 5
거스름돈이 부족합니다.
```

[정답]

```
// 1. 금액(money)을 동전단위로 나눠서 필요한 동전의 개수(coinNum)를 구한다.
coinNum = money/coinUnit[i];

// 2. 배열 coin에서 coinNum만큼의 동전을 뺀다.
//   (만일 충분한 동전이 없다면 배열 coin에 있는 만큼만 뺀다.)
if(coin[i] >= coinNum) {
    coin[i] -= coinNum;
} else {
    coinNum = coin[i];
    coin[i] = 0;
}

// 3. 금액에서 동전의 개수(coinNum)와 동전단위를 곱한 값을 뺀다.
money -= coinNum*coinUnit[i];
```

[해설] 주어진 로직대로만 작성하면 별 어려움 없이 풀 수 있었을 것이라 생각한다. 문제 5-6을 이해했다면 이 문제도 쉽게 이해될 것이므로 자세한 설명은 생략한다.

이 예제를 발전시켜 자판기 프로그램을 작성해보면 좋은 공부가 될 것이다.

[5-8] 다음은 배열 answer에 담긴 데이터를 읽고 각 숫자의 개수를 세어서 개수만큼 '*'을 찍어서 그래프를 그리는 프로그램이다. (1)~(2)에 알맞은 코드를 넣어서 완성하시오.

[연습문제]/ch5/Exercise5_8.java

```
class Exercise5_8 {
     public static void main(String[] args) {
           int[] answer = { 1,4,4,3,1,4,4,2,1,3,2 };
           int[] counter = new int[4];

           for(int i=0; i < answer.length;i++) {
               counter[answer[i]-1]++;
           }

           for(int i=0; i < counter.length;i++) {
               System.out.print(counter[i]);

               for(int j=0; j < counter[i];j++) {
                   System.out.print("*"); // counter[i]의 값 만큼 '*'을 찍는다.
               }

               System.out.println();
           }
     }      // end of main
}      // end of class
```

[실행결과]

```
3***
2**
2**
4****
```

[정답]

(1) `counter[answer[i]-1]++;`

(2)

```
System.out.print(counter[i]);

for(int j=0; j < counter[i];j++) {
    System.out.print("*");
}
```

[해설] 배열을 이용해서 데이터의 개수를 세는 문제이다. 1~4범위의 데이터를 집계할 것이기 때문에 크기가 4인 배열(counter)을 생성하였다. 크기가 4이지만 배열의 index는 0~3이기 때문에 answer[i]에서 1을 빼주어야 한다.

```
counter[answer[i]-1]++;
```

Java의 정석 3판의 예제5-11과 거의 동일하므로 자세한 설명은 생략한다. 다만, Java의 정석 소스 압축파일에 있는 플래시 동영상'/flash/Array.swf'을 보면 단계별로 자세히 설명되어 있으니 참고하기 바란다.

Java의 정석 소스압축파일은 http://cafe.naver.com/javachobostudy 에서 얻을 수 있다.

[5-9] 주어진 배열을 시계방향으로 90도 회전시켜서 출력하는 프로그램을 완성하시오.

[연습문제]/ch5/Exercise5_9.java

```
class Exercise5_9 {
     public static void main(String[] args) {
       char[][] star = {
           {'*','*',' ',' ',' '},
           {'*','*',' ',' ',' '},
           {'*','*','*','*','*'},
           {'*','*','*','*','*'}
       };

       char[][] result = new char[star[0].length][star.length];

       for(int i=0; i < star.length;i++) {
           for(int j=0; j < star[i].length;j++) {
               System.out.print(star[i][j]);
           }
           System.out.println();
       }

       System.out.println();

       for(int i=0; i < star.length;i++) {
           for(int j=0; j < star[i].length;j++) {
               int x = j;
               int y = star.length-1-i;

               result[x][y]=star[i][j];
           }
       }

       for(int i=0; i < result.length;i++) {
           for(int j=0; j < result[i].length;j++) {
               System.out.print(result[i][j]);
           }
           System.out.println();
       }
     } // end of main
} // end of class
```

[실행결과]

```
**
**
*****
*****

****
****
**
**
**
```

[정답]

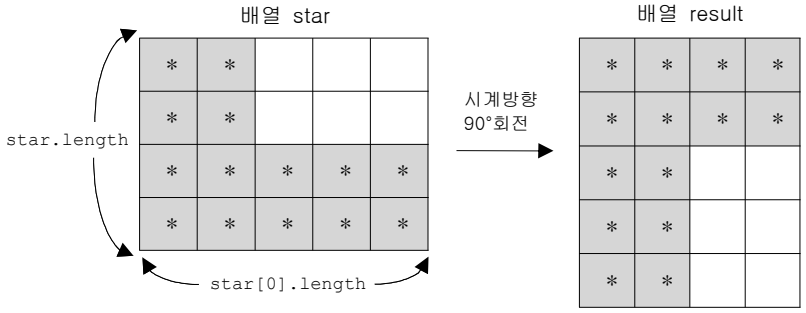
```
int x = j;
int y = star.length-1-i;

result[x][y]=star[i][j];
```

[해설] 테트리스의 도형을 회전시키듯이 배열을 회전시키는 문제이다. 배열 star가 4×5의 2차원 배열이므로 이 배열을 90도 회전시키면 5×4의 2차원 배열이 되어야한다.

```
char[][] result = new char[star[0].length][star.length];
```

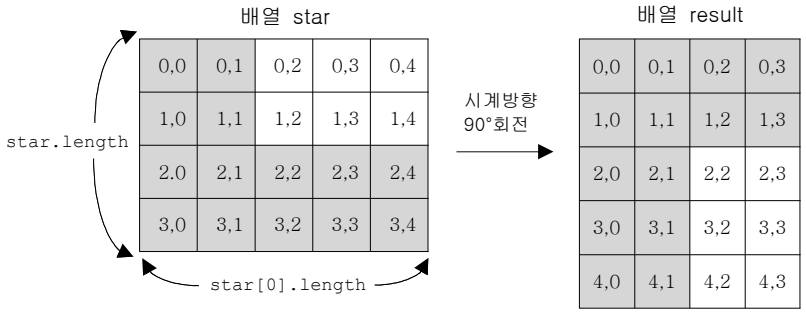
배열 star를 시계방향으로 회전시켜서 만든 배열 result를 그림으로 그려보면 다음과 같다.



result배열을 생성했으니, 이제 배열 star의 요소들을 배열 result의 적절한 위치로 옮기면 된다. 원래의 위치를 (i,j), 옮길 위치를 (x, y)라고 할 때 i와 j의 값을 이용해서 x와 y의 값을 어떻게 계산해 낼 것인지를 고민해보자.

```
for(int i=0; i < star.length;i++) {
    for(int j=0; j < star[i].length;j++) {
        int x = ???;
        int y = ???;

        result[x][y]=star[i][j]; // (i,j) → (x,y)
    }
}
```



위 그림은 배열의 인덱스를 표시한 것인데, star[0][0]은 result[0][3]으로, star[0][1]은 result[1][3]으로 이동해야 함을 알 수 있다. 이 것을 표로 정리하면 다음과 같다.

i	j
0	0
0	1
0	2
0	3
1	0
1	1
…	…
3	2
3	3
3	4

→

x	y
0	3
1	3
2	3
3	3
0	2
1	2
…	…
2	0
3	0
4	0

위의 표를 보면, x의 값은 j의 값과 정확히 일치함을 알 수 있다. 그래서 x의 값은 j의 값을 그대로 가져다 쓰면 된다. y의 값을 보면, i와 y의 합이 일정함을 알 수 있다. i와 y의 합은 항상 3이다. 3은 star.length-1과 동일한 값이다.

'i+y = star.length-1'이니까, 'y = star.length-1-i'라고 할 수 있다.

```
for(int i=0; i < star.length;i++) {
    for(int j=0; j < star[i].length;j++) {
        int x = j;
        int y = star.length-1-i;

        result[x][y]=star[i][j]; // (i,j) → (x,y)
    }
}
```

익숙하지 않겠지만 이렇게 그림을 그리고 표를 그리면 쉽게 해결된다. 코딩을 하기 전에 종이에 그림을 그리고 로직을 정리하는 것은 모니터를 보는 시간을 줄여서 눈의 피로를 적게 하고, 논리적 오류를 줄여줄 수 있다는 장점이 있다.

[5-10] 다음은 알파벳과 숫자를 아래에 주어진 암호표로 암호화하는 프로그램이다. (1)에 알맞은 코드를 넣어서 완성하시오.

a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t
`	~	!	@	#	$	%	^	&	*	(	)	-	_	+	=	|	[	]	{

u	v	w	x	y	z
}	;	:	,	.	/

0	1	2	3	4	5	6	7	8	9
q	w	e	r	t	y	u	i	o	p

[연습문제]/ch5/Exercise5_10.java

```
class Exercise5_10 {
   public static void main(String[] args)     {
        char[] abcCode =
           { '`','~','!','@','#','$','%','^','&','*',
             '(',')','-','_','+','=','|','[',']','{',
             '}',';',':',',','.','/'};
                        // 0   1   2   3   4   5   6   7   8   9
        char[] numCode = {'q','w','e','r','t','y','u','i','o','p'};

        String src = "abc123";
        String result = "";

        // 문자열 src의 문자를 charAt()으로 하나씩 읽어서 변환 후 result에 저장
        for(int i=0; i < src.length();i++) {
            char ch = src.charAt(i);

            if('a' <= ch && ch <='z') {
                result += abcCode[ch-'a'];
            } else if('0' <= ch && ch <='9') {
                result += numCode[ch-'0'];
            }
        }

        System.out.println("src:"+src);
        System.out.println("result:"+result);

   } // end of main
} // end of class
```

[실행결과]

```
src:abc123
result:`~!wer
```

[정답]

```
if('a' <= ch && ch <='z') {
    result += abcCode[ch-'a'];
} else if('0' <= ch && ch <='9') {
    result += numCode[ch-'0'];
}
```

[해설] 문자열을 반복문과 chatAt(int i)을 이용해서, 한 문자씩 배열에 있는 암호코드로 변경해서 암호화하는 문제이다.

암호코드는 영어소문자와 숫자로 나뉘어져 있는데, 영어소문자인 경우 배열 abcCode에서 해당 암호코드를 얻고, 숫자인 경우에는 배열 numCode에서 암호코드를 얻도록 되어 있다.

그래서 조건문으로 문자가 영어소문자인 경우와 숫자인 경우를 나누어서 처리했다.

```
if('a' <= ch && ch <='z') {
    result += abcCode[ch-'a'];
} else if('0' <= ch && ch <='9') {
    result += numCode[ch-'0'];
}
```

암호코드배열 abcCode에는 문자 'a'시작해서 문자 'z'까지의 암호코드가 순서대로 저장되어 있기 때문에 문자 'a'의 암호코드는 abcCode[0]이고, 문자 'c'의 암호코드는 abcCode[2]이다. 즉, 영어소문자 ch의 암호코드는 abcCode[ch-'a']로 표현할 수 있는 것이다.

만일 문자 ch가 'c'였다면, 조건식 'a' <= ch && ch <='z'가 true가 되어 result+= abcCode[ch-'a';가 수행된다. 이 문장은 아래와 같은 순서로 연산이 진행된다.

```
result+= abcCode[ch-'a']; → result+= abcCode['c'-'a'];
→ result+= abcCode[2]; → result+= '!';
```

알파벳이나 숫자는 문자코드가 연속적으로 할당되어 있기 때문에, 'c'에서 'a'를 빼면 2를 결과로 얻는다.

```
'c'-'a' → 99 - 97 → 2
```

뺄셈과 같은 이항연산자는 int타입보다 작은 타입은 피연산자(byte, short, char)은 int로 변환한다. 그래서 'c'-'a'은 99 - 97로 변환되고 그 결과는 숫자 2가 된다. 참고로 문자 'a'와 'c'의 코드는 아래와 같다.

문자	문자코드
...	...
'a'	97
'b'	98
'c'	99
'd'	100
...	...

[5-11] 주어진 2차원 배열의 데이터보다 가로와 세로로 1이 더 큰 배열을 생성해서 배열의 행과 열의 마지막 요소에 각 열과 행의 총합을 저장하고 출력하는 프로그램이다. (1)에 알맞은 코드를 넣어서 완성하시오.

[연습문제]/ch5/Exercise5_11.java

```
class Exercise5_11
{
   public static void main(String[] args)
   {
       int[][] score = {
                 {100, 100, 100}
               , {20, 20, 20}
               , {30, 30, 30}
               , {40, 40, 40}
               , {50, 50, 50}
            };

       int[][] result = new int[score.length+1][score[0].length+1];

       for(int i=0; i < score.length;i++) {
           for(int j=0; j < score[i].length;j++) {
               result[i][j] = score[i][j];
               result[i][score[0].length] += result[i][j];
               result[score.length][j] += result[i][j];
               result[score.length][score[0].length] += result[i][j];
           }
       }

       for(int i=0; i < result.length;i++) {
           for(int j=0; j < result[i].length;j++) {
               System.out.printf("%4d",result[i][j]);
           }
           System.out.println();
       }
   } // main
}
```

[실행결과]

```
 100 100 100 300
  20  20  20  60
  30  30  30  90
  40  40  40 120
  50  50  50 150
 240 240 240 720
```

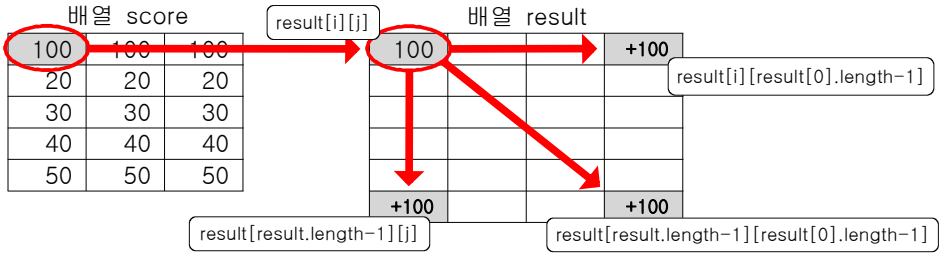
[정답]

```
result[i][j] = score[i][j];
result[i][score[0].length] += result[i][j];
result[score.length][j] += result[i][j];
result[score.length][score[0].length] += result[i][j];
```

[해설] 2차원 배열의 복사를 조금 응용한 문제이다. 2차원 배열 score에 담긴 값들을 2차원 배열 result에 복사하되 배열 result의 맨 오른쪽 열에는 각 행의 총합이, 그리고 맨

마지막 행에는 각 열의 총합이, 마지막으로 맨 오른쪽 행의 마지막 열에는 전체 총합이 저장되어야 한다. 언뜻 복잡해 보이지만 그림을 그려보면 간단명료해 진다.

먼저 score[i][j]를 result[i][j]에 저장하고, result[i][j]는 아래 그림에 표시한 세 곳에 누적해서 더해주면 된다.



result.length-1과 result[0].length-1은 score.length와 score[0].length와 각각 동일하므로 보다 식을 간단히 하기 위해 대체해서 사용했는데, 반드시 그렇게 해야 하는 것은 아니다.

```
result.length-1    → score.length
result[0].length-1 → score[0].length
```

score[i][j]의 값을 result[i][j]에 저장하기 때문에, result[i][j] 대신 score[i][j]를 사용해도 같은 결과를 얻는다.

```
result[i][score[0].length] += score[i][j];
result[score.length][j] += score[i][j];
result[score.length][score[0].length] += score[i][j];
```

[5-12] 예제5-23을 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.
```

[정답]

[연습문제]/ch5/Exercise5_12.java

```
import java.util.*;

class  Exercise5_12 {
   public static void main(String[] args) {
        String[][] words = {
            {"chair","의자"},          // words[0][0], words[0][1]
            {"computer","컴퓨터"},     // words[1][0], words[1][1]
            {"integer","정수"}         // words[2][0], words[2][1]
        };

        int score = 0; // 맞춘 문제의 수를 저장하기 위한 변수

        Scanner scanner = new Scanner(System.in);

        for(int i=0;i<words.length;i++) {
            System.out.printf("Q%d. %s의 뜻은?", i+1, words[i][0]);

            String tmp = scanner.nextLine();

            if(tmp.equals(words[i][1])) {
                System.out.printf("정답입니다.%n%n");
                score++;
            } else {
              System.out.printf("틀렸습니다. 정답은 %s입니다.%n%n",words[i][1]);
            }
        } // for

        System.out.printf("전체 %d문제 중 %d문제 맞추셨습니다.%n",
                                                    words.length, score);
   } // main의 끝
}
```

[해설] 맞춘 문제의 수를 저장하기 위한 변수 하나를 추가하고, 정답을 맞추면 이 변수의 값을 증가시키기만 하면 된다.

[5-13] 단어의 글자위치를 섞어서 보여주고 원래의 단어를 맞추는 예제이다. 실행결과와 같이 동작하도록 예제의 빈 곳을 채우시오.

[연습문제5-13]/ch5/Excercise5_13.java

```
import java.util.Scanner;

class Exercise5_13 {
   public static void main(String args[]) {
        String[] words = { "television", "computer", "mouse", "phone" };

        Scanner scanner = new Scanner(System.in);

        for(int i=0;i<words.length;i++) {
            char[] question = words[i].toCharArray(); // String을 char[]로 변환

            for(int j=0;j<question.length;j++) {
                int idx = (int)(Math.random() * question.length);

                char tmp = question[i];
                question[i] = question[idx];
                question[idx] = tmp;
            }

            System.out.printf("Q%d. %s의 정답을 입력하세요.>",
                                                    i+1, new String(question));
            String answer = scanner.nextLine();

            // trim()으로 answer의 좌우 공백을 제거한 후, equals로 word[i]와 비교
            if(words[i].equals(answer.trim()))
                System.out.printf("맞았습니다.%n%n");
            else
                System.out.printf("틀렸습니다.%n%n");
        }
   } // main의 끝
}
```

[실행결과]

```
Q1. lvtsieeoin의 정답을 입력하세요.>television
맞았습니다.

Q2. otepcumr의 정답을 입력하세요.>computer
맞았습니다.

Q3. usemo의 정답을 입력하세요.>asdf
틀렸습니다.

Q4. ohpne의 정답을 입력하세요.>phone
맞았습니다.
```

[정답]

```
for(int j=0;j<question.length;j++) {
    int idx = (int)(Math.random() * question.length);
```

```
            char tmp = question[i];
            question[i] = question[idx];
            question[idx] = tmp;
        }
```

[해설] 예제5-8을 응용한 문제이다. 간단한 문제이므로 설명은 생략한다.