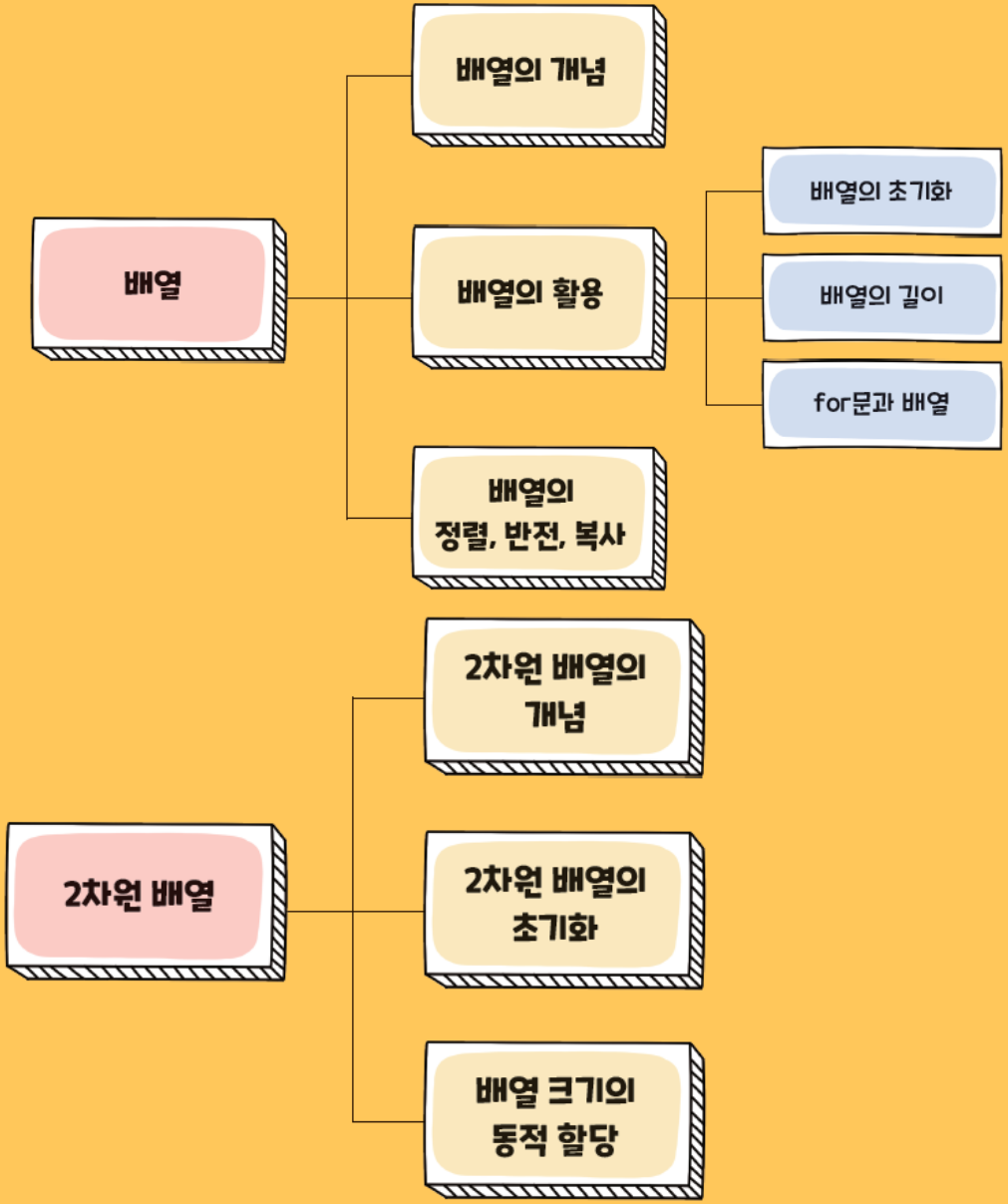
bantaehak

Chapter 09
배열

목차

Preview



학습목표

- 배열을 이해하고 활용법을 익힙니다.
- 배열의 정렬, 반전, 복사 등 고급 처리 방법을 익힙니다.
- 2차원 배열을 이해하고 활용법을 익힙니다.

Section 01
배열

1. 배열의 개념

- 배열을 사용하는 이유
  - 정수형 변수 num1~num4를 선언하고 변수에 값을 입력 받아 합계를 출력하는 프로그램을 작성하는 경우
    - → 변수 네 개를 선언하고 값을 입력 받아 합을 구하는 코드는 간단함
    - → 하지만 변수가 1000개 변수 num1~num1000을 입력하는 것은 현실적으로 어려움
    - → 이럴 때 배열을 이용함

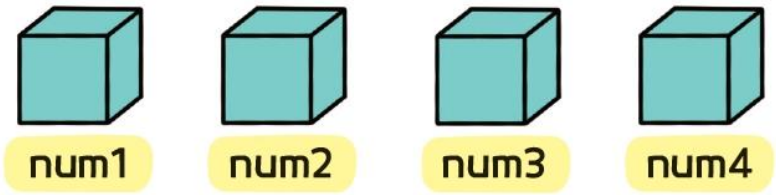


그림 9-1 변수 네 개 준비

1. 배열의 개념

- 변수를 여러 개 사용하는 경우

코드 9-1 Code09_01.java

```
01  import java.util.Scanner;
02  public class Code09_01 {
03      public static void main(String[] args) {
04          Scanner s = new Scanner(System.in);
05          int num1, num2, num3, num4;
06          int hap = 0;
07
08          System.out.print("숫자 : ");
09          num1 = s.nextInt();
10          System.out.print("숫자 : ");
11          num2 = s.nextInt();
12          System.out.print("숫자 : ");
13          num3 = s.nextInt();
14          System.out.print("숫자 : ");
15          num4 = s.nextInt();
16
17          hap = num1 + num2 + num3 + num4;
18          System.out.println("합계 ==> " + hap);
19
20          s.close();
21      }
22  }
```

```
숫자 : 10
숫자 : 20
숫자 : 30
숫자 : 40
합계 ==> 100
```

사용자가 입력

1. 배열의 개념

배열

- 배열에 여러 데이터를 한꺼번에 담을 수 있음
- 여러 개의 값을 하나로 묶어 놓은 꾸러미와 같음
- 배열은 단독으로 사용하기보다는 주로 for문과 함께 사용함

배열을 사용하는 경우

→ 배열은 하나씩 사용하던 변수를 한 줄로 붙여 놓은 것

→ 변수를 한 덩어리로 만들면 변수명도 한 개로 충분함

→ 한 줄로 붙인 덩어리 전체의 이름을 numAry와 같이 하나로 지정함

→ 배열을 이루는 각 요소에는 numAry[0] ~numAry[3]과 같이 번호(첨자)를 붙여 접근함

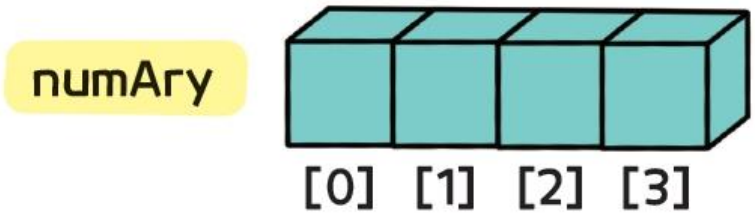


그림 9-2 배열의 개념

1. 배열의 개념

- 배열을 선언하는 형식

```
데이터형[] 배열명 = new 데이터형[개수];
또는
데이터형 배열명[] = new 데이터형[개수]
```

  - 배열 선언을 먼저 하고 new 연산자를 나중에 사용하는 방법

```
데이터형[] 배열명; (또는 데이터형 배열명[];)
배열명 = new [개수];
```

  - [예] 네 개의 변수를 담은 정수형 배열 선언하기

```
int[] numAry = new int[4];
또는
int numAry[] = new int[4];
또는
int[] numAry; (또는 int numAry[];)
numAry = new int[4];
```

1. 배열의 개념

- 변수 선언과 배열 선언의 차이점
  - 변수 선언: 각 변수를 int num1, num2, ... ;와 같이 선언하여 사용함
  - 배열 선언: numAry[0], numAry[1], ... 과 같이 첨자로 표시하여 사용함
    → 첨자는 1이 아니라 0부터 시작됨

코드 9-2 Code09_02.java

```
01  import java.util.Scanner;
02  public class Code09_02 {
03      public static void main(String[] args) {
04          Scanner s = new Scanner(System.in);
05          int[] numAry = new int[4];
06          int hap = 0;
07
08          System.out.print("숫자 : ");
09          numAry[0] = s.nextInt();
10          System.out.print("숫자 : ");
11          numAry[1] = s.nextInt();
12          System.out.print("숫자 : ");
13          numAry[2] = s.nextInt();
14          System.out.print("숫자 : ");
15          numAry[3] = s.nextInt();
16
17          hap = numAry[0] + numAry[1] + numAry[2] + numAry[3];
18          System.out.println("합계 ==> " + hap);
19
20          s.close();
21      }
22  }
```

```
숫자 : 10
숫자 : 20
숫자 : 30
숫자 : 40
합계 ==> 100
```

사용자가 입력

1. 배열의 개념

확인문제

세 숫자를 저장할 배열을 선언하기 위한 다음 코드의 빈칸을 채우시오.

```
int[] numAry = [          ];
```

정답

Click!

2. 배열의 활용

배열의 특성과 활용

- 배열에서 100개의 숫자를 더하는 경우
  - → 배열을 선언하고 값을 입력하는 과정을 100번 반복
  - → numAry[0] + numAry[1] + numAry[2] + …+ numAry[99]
  - → 이렇게 사용하면 배열의 의미가 없음
- 첨자를 반복문(for, while, do~while)과 결합하여 사용하기
  - → 첨자가 순서대로 바뀌도록 반복문과 함께 활용해야 함
- [코드 9-2]의 값 대입 부분에 for문을 활용하기

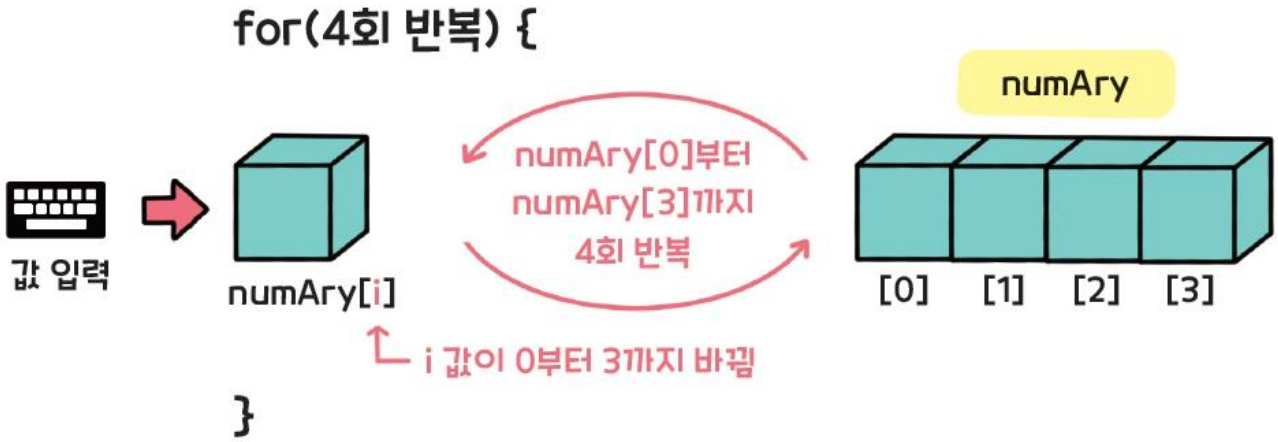


그림 9-3 for문을 활용하여 배열에 값을 대입하는 경우의 동작 원리

2. 배열의 활용

배열의 특성과 활용

- [코드 9-2]의 8~15행(배열에 값을 대입하는 부분)을 반복문으로 수정한 코드

코드 9-3 Code09_03.java

```
01  import java.util.Scanner;
02  public class Code09_03 {
03      public static void main(String[] args) {
04          Scanner s = new Scanner(System.in);
05          int[] numAry = new int[4];
06          int hap = 0;
07
08          for (int i=0; i<=3; i++) {
09              System.out.print("숫자 : ");
10              numAry[i] = s.nextInt();
11          }
12
13          hap = numAry[0] + numAry[1] + numAry[2] + numAry[3];
14          System.out.println("합계 ==> " + hap);
15
16          s.close();
17      }
18  }
```

```
숫자 : 10
숫자 : 20
숫자 : 30
숫자 : 40
합계 ==> 100
```

사용자가 입력

2. 배열의 활용

배열의 초기화

- 배열을 선언하는 동시에 값을 대입하는 것
- 배열을 선언하는 대괄호 안에 개수를 적지 않음
- 선언한 초깃값 개수에 따라서 배열의 요소가 차례대로 초기화됨

```
int numAry[] = {100, 200, 300, 400};
```

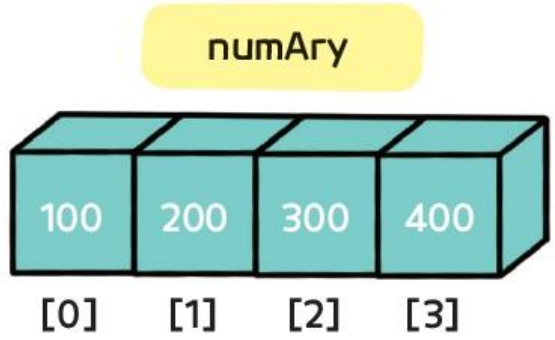


그림 9-4 배열의 초기화 1

2. 배열의 활용

배열의 초기화

- 배열을 먼저 선언하고 초깃값을 대입하는 방식으로 초기화를 할 수도 있음

```
int numAry[];
numAry = new int[] {100, 200, 300, 400};
```

- 배열을 선언하기만 하고 초기화를 하지 않을 경우 모든 요소에 0이 들어감

```
int[] numAry = new int[4];
```

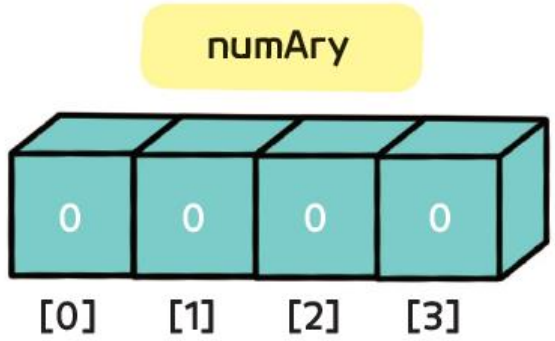


그림 9-5 배열의 초기화 2

2. 배열의 활용

- 배열의 초기화

코드 9-4 Code09_04.java

```
01  public class Code09_04 {
02      public static void main(String[] args) {
03          int ary1[] = {100, 200, 300, 400};
04          int ary2[] = new int[] {100, 200, 300};
05          int ary3[];
06          ary3 = new int[] {100, 200};
07          int[] ary4 = new int[1];
08          ary4[0] = 100;
09
10          for (int i = 0; i < 4; i++)
11              System.out.printf("ary1[%d]==>%d\t", i, ary1[i]);
12          System.out.println();
13
14          for (int i = 0; i < 3; i++)
15              System.out.printf("ary2[%d]==>%d\t", i, ary2[i]);
16          System.out.println();
17
18          for (int i = 0; i < 2; i++)
19              System.out.printf("ary3[%d]==>%d\t", i, ary3[i]);
20          System.out.println();
21
22          for (int i = 0; i < 1; i++)
23              System.out.printf("ary4[%d]==>%d\t", i, ary4[i]);
24          System.out.println();
25      }
26  }
```

```
ary1[0]==>100   ary1[1]==>200   ary1[2]==>300   ary1[3]==>400
ary2[0]==>100   ary2[1]==>200   ary2[2]==>300
ary3[0]==>100   ary3[1]==>200
ary4[0]==>100
```

\t는 Tab을 의미합니다.

2. 배열의 활용

■ 배열의 요소 개수

- 배열의 길이(요소 개수)를 알아내는 형식

```
배열요소개수 = 배열명.length;
```

- [예] int ary[] = new int[4];일 때 ary의 길이 알아내기

```
int count = ary.length;
```

코드 9-5 Code09_05.java

```
01  public class Code09_05 {
02      public static void main(String[] args) {
03          int ary[] = {10, 20, 30, 40, 50};
04          int count, size;
05
06          count = ary.length;
07          size = count * Integer.BYTES;
08
09          System.out.println("배열 ary[]의 요소 개수 : " + count);
10          System.out.println("배열 ary[]의 요소 전체 크기 : " + size);
11      }
12  }
```

```
배열 ary[]의 요소 개수 : 5
배열 ary[]의 요소 전체 크기 : 20
```

2. 배열의 활용

■ 배열의 길이 변경

- 배열의 길이를 늘려야 하는 경우

  → 이때 기존 데이터를 그대로 유지하면서 배열의 길이 변경하기

- 배열의 길이를 변경하는 형식

```
import java.util.Arrays;
배열명 = Arrays.copyOf(배열명, 새로운_길이);
```

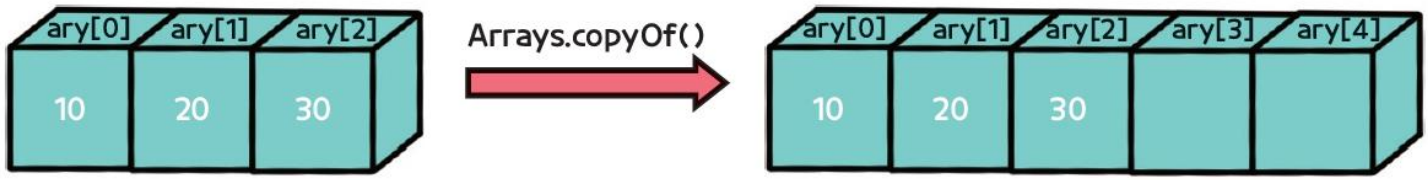


그림 9-6 배열의 길이 변경

2. 배열의 활용

■ 배열의 길이 변경

- [예] ary 배열의 길이를 3에서 5로 변경하기

코드 9-6 Code09_06.java

```
01  import java.util.Arrays;
02  public class Code09_06 {
03      public static void main(String[] args) {
04          int ary[] = {10, 20, 30};
05
06          System.out.println("현재 배열 길이 : " + ary.length);
07          System.out.print("현재 배열의 내용 : ");
08          for (int i=0; i < ary.length; i++)
09              System.out.print(ary[i] + " ");
10          System.out.println();
11
12          ary = Arrays.copyOf(ary, ary.length + 2);
13
14          System.out.println("새 배열 길이 : " + ary.length);
15          System.out.print("새 배열의 내용 : ");
16          for (int i=0; i < ary.length; i++)
17              System.out.print(ary[i] + " ");
18          System.out.println();
19      }
20  }
```

```
현재 배열 길이 : 3
현재 배열의 내용 : 10 20 30
새 배열 길이 : 5
새 배열의 내용 : 10 20 30 0 0
```

2. 배열의 활용

[하나 더 알기] 인덱스 오류

배열을 사용할 때 흔히 저지르는 실수는 첨자의 범위를 벗어나 접근하는 것이며, 이로 인해 발생하는 오류를 인덱스 오류(index error)라고 합니다. 다음 코드의 오류를 살펴봅시다.

```
int numAry[] = {10, 20, 30};
System.out.println(numAry[3]);
```

```
Exception in thread "main" java.lang.ArrayIndexOutOfBoundsException:
        Index 3 out of bounds for length 3
```

오류 메시지의 'Index 3 out of bounds'는 범위를 벗어났다는 의미입니다. 현재 numAry에는 세 개의 값이 들어 있습니다. 즉 numAry[0]이 10, numAry[1]이 20, numAry[2]가 30인데, 여기에 속하지 않는 numAry[3]을 출력하려고 시도했기 때문에 이러한 오류가 발생한 것입니다. 이처럼 인덱스 번호와 요소 개수가 동일하다고 착각하기 쉬운데, 인덱스는 0부터 시작된다는 것을 잊지 마세요.

2. 배열의 활용

for문의 다른 용법

- 배열의 요소에 하나씩 접근하여 출력하는 for문

  → 배열의 요소를 추출하기 위해 첨자 i를 사용함

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

for (int i=0; i<ary.length; i++) {
    int data = ary[i];
    System.out.println(data);
}
```

- 편리하게 배열에 접근할 수 있도록 **for(변수 : 배열)** 형식의 구문을 사용할 수 있음

코드 9-7 Code09_07.java

```
01  public class Code09_07 {
02      public static void main(String[] args) {
03          int[] ary = {10, 20, 30};
04          for (int data : ary) {
05              System.out.println(data);
06          }
07      }
08  }
```

```
10
20
30
```

2. 배열의 활용

확인문제

1. 다음 빈칸에 알맞은 말을 넣으시오.

1000개의 배열에 값을 대입하거나 계산하기 위해 [ ]문을 한다. 이렇게 하지 않으면 현실적으로 코드를 작성하기가 어렵다.

2. 아래 코드를 보고 다음 빈칸에 알맞은 말을 넣으시오.

다음과 같이 배열을 초기화하면 배열의 요소 개수는 [㉠]개이고, 마지막 요소의 첨자는 [㉡]이다.

```
int[] ary = new int[3];
ary[0] = 5;
```

3. array 배열의 길이를 5만큼 늘리기 위한 다음 코드의 빈칸을 채우시오.

```
array = Arrays.copyOf(array, [ ] + 5);
```

정답

Click!

3. 배열의 정렬, 반전, 복사

- 배열의 정렬
  - 정렬: 어떤 값을 순서대로 나열하는 것
  - 오름차순 정렬: 작은 것부터 큰 것 순으로 나열함
  - 내림차순 정렬: 큰 것부터 작은 것 순으로 나열함
  - Arrays.sort() 메서드
    - → 배열의 값을 정렬할 때는 사용하는 메서드
    - → 배열의 내용이 정수든 실수든 문자열이든 정렬할 수 있음

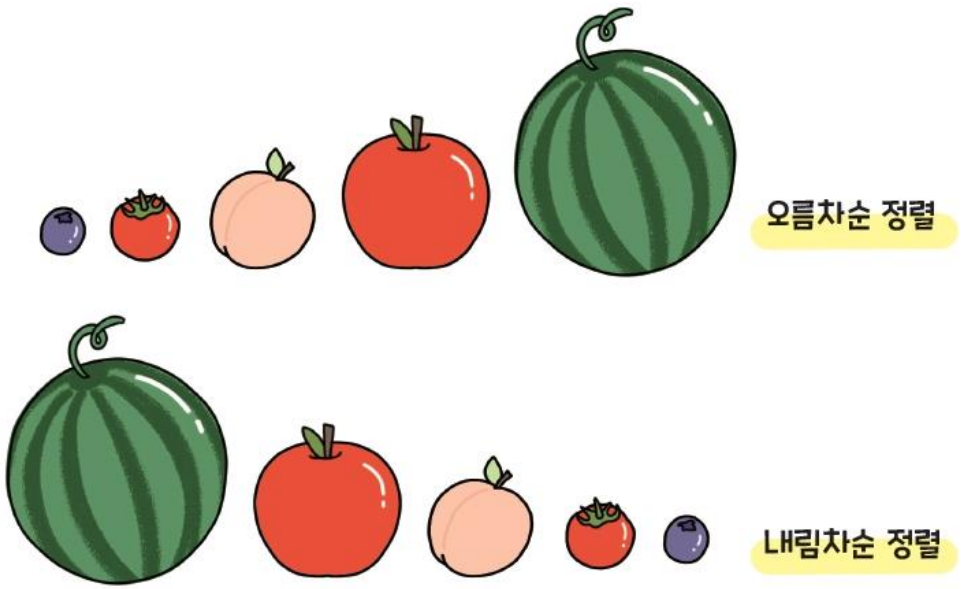


그림 9-7 오름차순 정렬과 내림차순 정렬

3. 배열의 정렬, 반전, 복사

■ 배열의 정렬

코드 9-8 Code09_08.java

```
01  import java.util.Arrays;
02  import java.util.Collections;
03  public class Code09_08 {
04      public static void main(String[] args) {
05          int[] numAry = {33, 99, 11, 77, 22, 88 , 66, 44};
06          Arrays.sort(numAry);
07          for (int data : numAry) {
08              System.out.print(data + " ");
09          }
10          System.out.println();
11
12          String[] strAry = {"한빛", "아카데미", "난생", "자바", "열공"};
13          Arrays.sort(strAry, Collections.reverseOrder());
14          for (String data : strAry) {
15              System.out.print(data + " ");
16          }
17      }
18  }
```

```
11 22 33 44 66 77 88 99
한빛 자바 열공 아카데미 난생
```

3. 배열의 정렬, 반전, 복사

배열의 반전

- 배열의 순서를 반대로 하는 것
- Arrays.asList(배열명)으로 배열을 리스트(list)라는 자료형으로 변환한 후 Collections.reverse(리스트) 메서드를 사용해야 함

코드 9-9 Code09_09.java

```
01  import java.util.Arrays;
02  import java.util.Collections;
03  public class Code09_09 {
04      public static void main(String[] args) {
05          String[] strAry = {"해린", "혜인", "하니", "민지", "다니엘"};
06          System.out.println("원본 : " + Arrays.toString(strAry));
07
08          Collections.reverse(Arrays.asList(strAry));
09
10          System.out.println("역순 : " + Arrays.toString(strAry));
11      }
12  }
```

```
원본 : [해린, 혜인, 하니, 민지, 다니엘]
역순 : [다니엘, 민지, 하니, 혜인, 해린]
```

3. 배열의 정렬, 반전, 복사

배열의 복사

- 배열을 복사할 때는 주의할 점
- [예] 배열 oldAry를 newAry로 복사한 후 newAry를 수정하여 출력해보기
  → 예상과 실행 결과가 다름

코드 9-10 Code09_10.java

```
01  import java.util.Arrays;
02  public class Code09_10 {
03      public static void main(String[] args) {
04          String[] oldAry = {"짜장", "탕수육", "군만두"};
05          String[] newAry;
06
07          newAry = oldAry;
08
09          oldAry[0] = "쟁반짜장";
10          newAry[1] = "짬뽕";
11
12          System.out.println("원본 배열 : " + Arrays.toString(oldAry));
13          System.out.println("복사 배열 : " + Arrays.toString(newAry));
14      }
15  }
```

```
원본 배열 : [쟁반짜장, 짬뽕, 군만두]
복사 배열 : [쟁반짜장, 짬뽕, 군만두]
```

3. 배열의 정렬, 반전, 복사

- 배열의 복사
  - 얕은 복사(shallow copy)
    - → 동일한 메모리를 공유하는 복사
    - → newAry = oldAry;
  - 깊은 복사(deep copy)
    - → 동일한 메모리를 사용하지 않는 복사
    - → newAry = oldAry.clone();

```
newAry = oldAry.clone();
```

```
원본 배열 : [쟁반짜장, 탕수육, 군만두]
복사 배열 : [짜장, 짬뽕, 군만두]
```

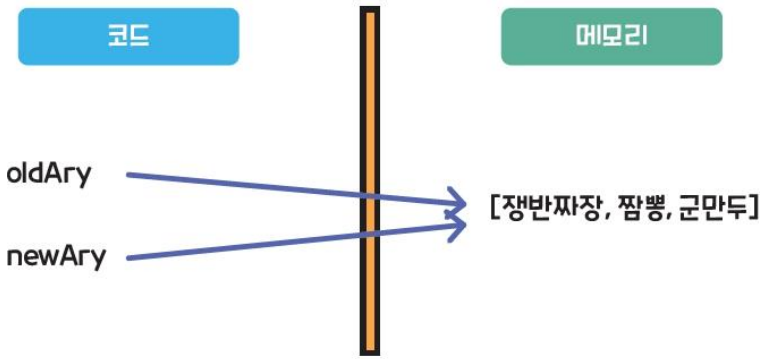


그림 9-8 얕은 복사(같은 메모리 공간을 공유하는 방식의 복사)

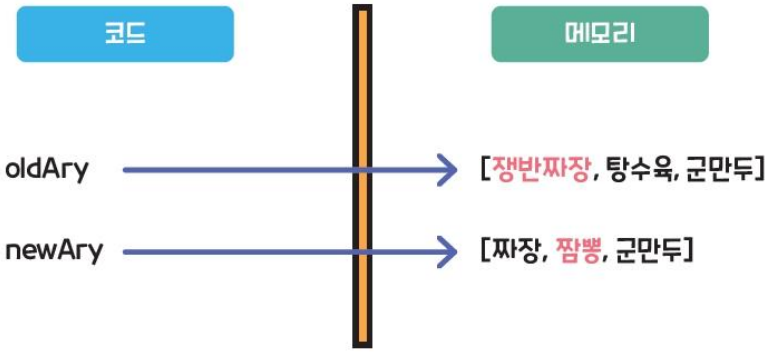


그림 9-9 깊은 복사(메모리 공간을 복사하여 새로 만드는 복사)

3. 배열의 정렬, 반전, 복사

확인문제

다음 빈칸에 알맞은 말을 넣으시오.

- 배열을 정렬하는 메서드는 Arrays.[㉠]()이고, 내림차순으로 정렬하려면 매개변수로 Collections.[㉡]()를 사용한다.
- 배열을 복사할 때 '배열명1=배열명2'의 경우 동일한 메모리 공간을 공유하는 [㉢] 복사라고 하며, '배열명1=배열명2.clone()'의 경우 새로운 메모리 공간을 할당하는 [㉣] 복사라고 한다.

정답

Click!

[LAB] 오늘의 명언 출력하기

다양한 명언을 배열에 저장해놓고 이를 무작위로 출력하는 프로그램을 만들어 봅시다.

실행 결과

```
오늘의 명언 ==> 종종 작은 기회로부터 위대한 업적이 시작된다.
```

[LAB] 오늘의 명언 출력하기

1. Lab09_01.java 파일을 만들고 빈 문자열 배열을 준비하기

```
String[] wiseSay = {};
```

2. 문자열 배열 안에 명언을 쉼표로 구분하여 입력하기
   - 마지막 명언에는 쉼표를 넣지 않음

```
"삶이 있는 한 희망은 있다.",
"언제나 현재에 집중할 수 있다면 행복할 것이다.",
"신은 용기 있는 자를 결코 버리지 않는다.",
"피할 수 없으면 즐겨라.",
"행복한 삶을 살기 위해 필요한 것은 거의 없다.",
"내일은 내일의 태양이 뜬다.",
"계단을 밟아야 계단 위에 올라설 수 있다.",
"행복은 습관이다. 그것을 몸에 지녀라.",
"1퍼센트의 가능성, 그것이 나의 길이다.",
"종종 작은 기회로부터 위대한 업적이 시작된다."
```

3. '오늘의 명언'을 무작위로 추출하기 위해 배열의 첨자 중 하나를 무작위로 뽑고 해당 첨자의 명언을 출력하기

```
int today = (int)(Math.random()*wiseSay.length);
System.out.println("오늘의 명언 ==> " + wiseSay[today]);
```

[LAB] 심사위원 점수 결과 구하기

피겨스케이팅 대회에서 김연아 선수가 연기를 펼치면 심사위원 다섯 명이 각각 점수를 매기고, 이 점수를 배열에 저장하여 평균 점수를 구하는 프로그램을 구현해봅시다.

이때 각 심사위원이 매기는 점수는 10점이 만점입니다.

실행 결과

```
김연아 선수 경기 끝났습니다~~ 짝짝짝
평가 점수==>10
평가 점수==>9
평가 점수==>8
평가 점수==>9
평가 점수==>10
심사위원 평균 점수 : 9.20
```

사용자가 입력

[LAB] 심사위원 점수 결과 구하기

1. Lab09_02.java 파일을 만들고, 키보드로 값을 입력받기 위해 Scanner 클래스를 사용할 수 있도록 준비하기

```
import java.util.Scanner;
Scanner s = new Scanner(System.in);

s.close();
```

2. 심사위원의 점수를 입력받을 배열과 합계 및 평균을 저장할 변수를 선언하기

```
int[] score = new int[5];
int hap = 0;
double avg;
```

3. 경기가 끝난 것을 표시하고, 심사위원 다섯 명에게 입력받은 점수를 배열에 저장하기

```
System.out.println("김연아 선수 경기 끝났습니다~~ 짝짝짝");

for (int i=0; i<5; i++) {
    System.out.print("평가 점수==>" );
    score[i] = s.nextInt();
}
```

[LAB] 심사위원 점수 결과 구하기

4. 배열의 점수를 합한 뒤 평균을 구하여 출력하기

```
for (int i=0; i<5; i++)
    hap += score[i];
avg = (double) hap / 5;

System.out.printf("심사위원 평균 점수 : %5.2f", avg);
```

Section 02
2차원 배열

1. 2차원 배열의 개념

2차원 배열의 개념

- 1차원 배열을 여러 개 연결한 것으로, 두 개의 첨자를 사용하는 배열
- 1차원 ary 배열: ary[0], ary[1], ary[2]라는 세 요소가 생성됨

```
int[] ary = new int[3];
```

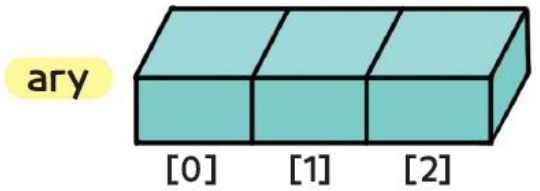


그림 9-10 1차원 배열의 개념

- 1차원 배열 ary를 2차원 배열 ary[3][4]로 확장하기
- ary[3][4]는 3행 4열의 배열로서 앞의 3은 가로줄 수, 뒤의 4는 세로줄 수를 의미함
  → ary[행][열]
- ary[3][4] 배열의 요소: 3×4=12개

```
int[][] ary = new int[3][4];
```

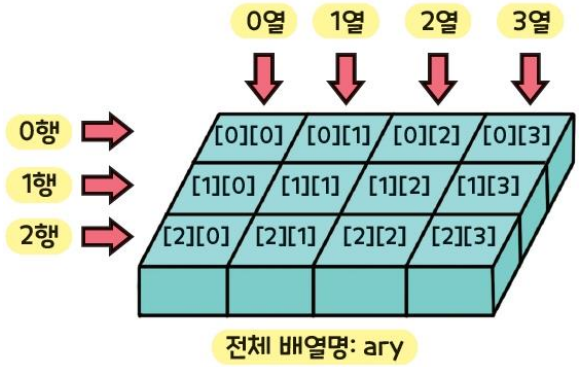


그림 9-11 2차원 배열의 개념

1. 2차원 배열의 개념

2차원 배열의 개념

코드 9-11 Code09_11.java

```
01  public class Code09_11 {
02      public static void main(String[] args) {
03          int[][] ary = new int[3][4];
04
05          ary[0][0] = 1; ary[0][1] = 2;  ary[0][2] = 3;  ary[0][3] = 4;
06          ary[1][0] = 5; ary[1][1] = 6;  ary[1][2] = 7;  ary[1][3] = 8;
07          ary[2][0] = 9; ary[2][1] = 10; ary[2][2] = 11; ary[2][3] = 12;
08
09          System.out.printf("ary[0][0]부터 ary[2][3]까지 출력 \n");
10
11          System.out.printf("%3d%3d%3d%3d\n", ary[0][0], ary[0][1], ary[0][2], ary[0][3]);
12          System.out.printf("%3d%3d%3d%3d\n", ary[1][0], ary[1][1], ary[1][2], ary[1][3]);
13          System.out.printf("%3d%3d%3d%3d\n", ary[2][0], ary[2][1], ary[2][2], ary[2][3]);
14      }
15  }
```

```
ary[0][0]부터 ary[2][3]까지 출력
  1  2  3  4
  5  6  7  8
  9 10 11 12
```

1. 2차원 배열의 개념

2차원 배열과 반복문

- 2차원 배열은 첨자가 두 개이므로 데이터의 입력 및 출력에 중첩 for문을 사용함

코드 9-12 Code09_12.java

```
01  public class Code09_12 {
02      public static void main(String[] args) {
03          int[][] ary = new int[3][4];
04          int value = 1;
05
06          for (int i = 0; i < 3; i++) {
07              for (int k = 0; k < 4; k++) {
08                  ary[i][k] = value;
09                  value++;
10              }
11          }
12
13          System.out.printf("ary[0][0]부터 ary[2][3]까지 출력 \n");
14          for (int i = 0; i < 3; i++) {
15              for (int k = 0; k < 4; k++) {
16                  System.out.printf("%3d ", ary[i][k]);
17              }
18              System.out.printf("\n");
19          }
20      }
21  }
```

```
ary[0][0]부터 ary[2][3]까지 출력
  1  2  3  4
  5  6  7  8
  9 10 11 12
```

2. 2차원 배열의 초기화

2차원 배열의 초기화

- 2차원 배열도 배열을 선언하는 동시에 값을 초기화할 수 있음

코드 9-13 Code09_13.java

```
01  public class Code09_13 {
02      public static void main(String[] args) {
03          int[][] ary = { {1, 2, 3, 4},
04                          {5, 6, 7, 8},
05                          {9, 10, 11, 12} };
06
07          System.out.printf("ary[0][0]부터 ary[2][3]까지 출력 \n");
08          for (int i = 0; i < 3; i++) {
09              for (int k = 0; k < 4; k++) {
10                  System.out.printf("%3d", ary[i][k]);
11              }
12              System.out.printf("\n");
13          }
14      }
15  }
```

```
ary[0][0]부터 ary[2][3]까지 출력
  1  2  3  4
  5  6  7  8
  9 10 11 12
```

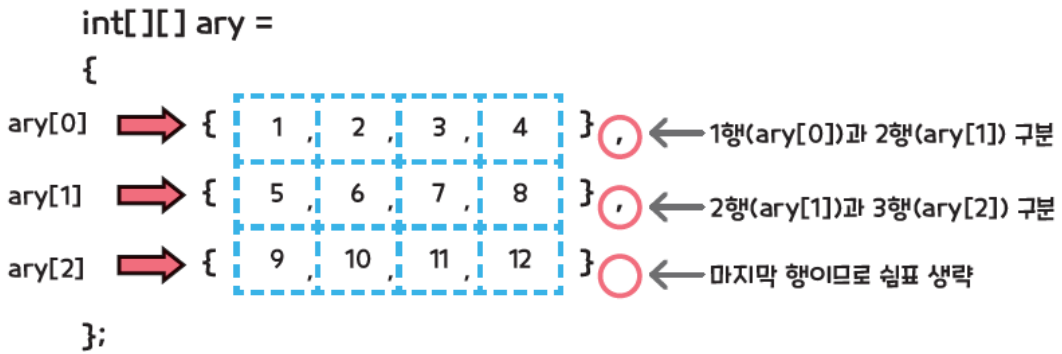


그림 9-12 2차원 배열의 초기화

3. 배열 크기의 동적 할당

배열 크기의 동적 할당

- 배열을 생성할 때 크기를 지정하지 않고 사용자의 입력에 따라 배열의 크기를 지정할 수 있음
- 필요에 따라 배열의 크기를 바꿀 수 있어 유용함

코드 9-14 Code09_14.java

```
01  import java.util.Scanner;
02  public class Code09_14 {
03      public static void main(String[] args) {
04          Scanner s = new Scanner(System.in);
05          int row, col;
06
07          System.out.print("행 개수 입력 : ");
08          row = s.nextInt();
09          System.out.print("열 개수 입력 : ");
10          col = s.nextInt();
11
12          int[][] ary = new int[row][col];
13          int value = 1;
14
15          for (int i = 0; i < row; i++) {
16              for (int k = 0; k < col; k++) {
17                  ary[i][k] = value;
18                  value++;
19              }
20          }
```

3. 배열 크기의 동적 할당

■ 배열 크기의 동적 할당

```
21
22          System.out.printf("ary[0][0]부터 ary[%d][%d]까지 출력 \n",row,col);
23          for (int i = 0; i < row; i++) {
24              for (int k = 0; k < col; k++) {
25                  System.out.printf("%3d", ary[i][k]);
26              }
27              System.out.println();
28          }
29      }
30  }
```

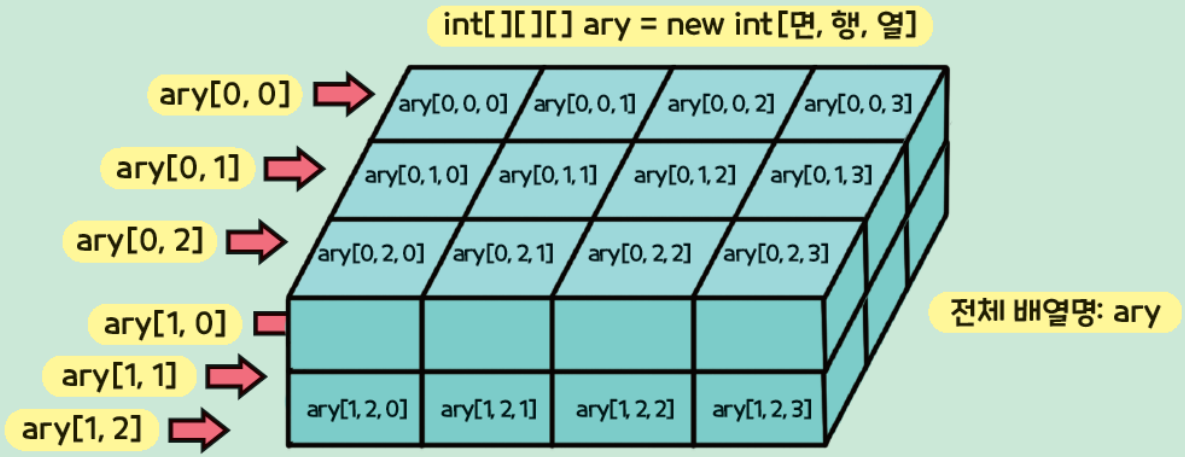
```
행 개수 입력 : 3
열 개수 입력 : 5
ary[0][0]부터 ary[3][5]까지 출력
  1  2  3  4  5
  6  7  8  9 10
 11 12 13 14 15
```

사용자가 입력

3. 배열 크기의 동적 할당

[하나 더 알기] 3차원 이상의 배열

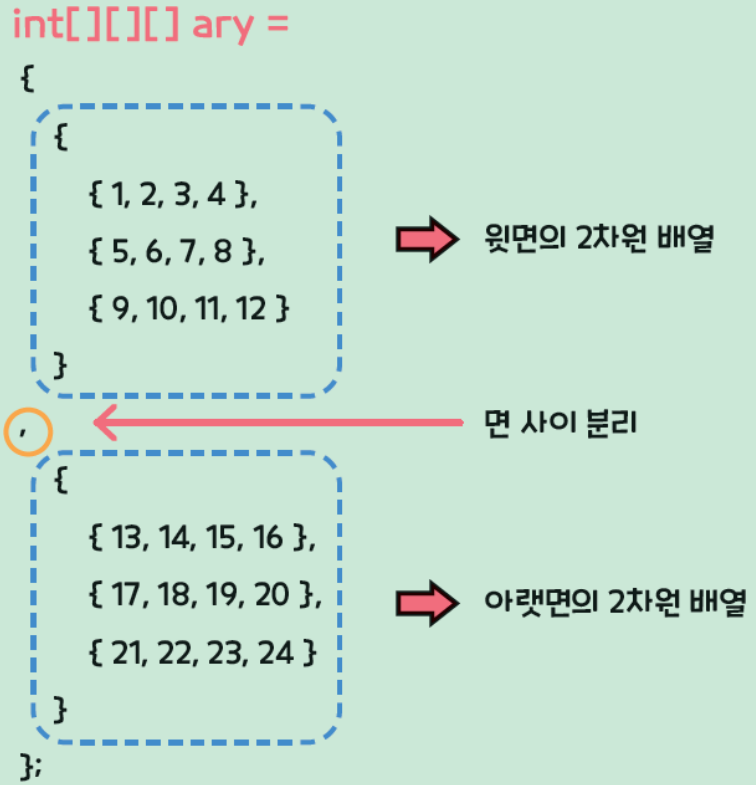
자바에서는 3차원, 4차원, 그 이상의 배열도 사용할 수 있습니다. 사실 3차원 이상의 배열을 사용하는 경우가 드물지만 개념은 알아두는 것이 좋습니다. 3차원 배열은 다음 그림과 같이 2차원 배열 위에 2차원 배열을 쌓아놓은 것으로, 2차원 배열 하나를 한 면으로 취급하여 첨자를 추가합니다.



3. 배열 크기의 동적 할당

[하나 더 알기] 3차원 이상의 배열

이러한 2면 3행 4열의 3차원 배열은 다음과 같이 초기화할 수 있습니다. 2차원 배열의 초기화를 한 번 더 한다고 생각하고 각 2차원을 쉼표로 분리했다가 전체를 다시 중괄호로 묶습니다.



3. 배열 크기의 동적 할당

확인문제

1. 다음과 같은 배열의 요소는 총 몇 개인가요?

```
int[][] ary = new int[5][4];
```

2. 2차원 문자열 배열을 동적 할당하기 위한 다음 코드의 빈칸을 채우시오.

```
r = s.nextInt();
c = s.nextInt();

String[][] strAry = new String[ ㉠ ][ ㉡ ];
```

정답

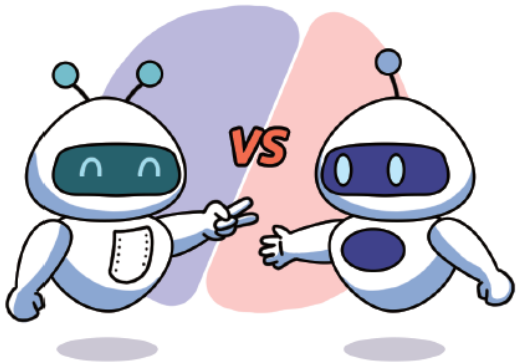
Click!

[LAB] 가위바위보 결과를 배열로 만들기

두 대의 컴퓨터가 가위바위보를 10000번 한 결과를 배열에 저장하고 누가 더 많이 이겼는지 확인하는 프로그램을 만들어봅시다.

이미 잘 알고 있겠지만 가위바위보의 규칙은 다음과 같습니다.

컴퓨터A	컴퓨터B	이긴 컴퓨터
가위	가위	없음
	바위	B
	보	A
바위	가위	A
	바위	없음
	보	B
보	가위	B
	바위	A
	보	없음



실행 결과

```
10000번 중 컴퓨터A의 승리 : 3349
10000번 중 컴퓨터B의 승리 : 3409
10000번 중 비긴 경기 : 3242
```

[LAB] 가위바위보 결과를 배열로 만들기

1. Lab09_03.java 파일을 만들고 Arrays, Collections를 임포트하기. 그리고 가위바위보 결과를 저장할 배열(toss), 컴퓨터A와 컴퓨터B가 내는 가위바위보를 저장할 변수(comA, comB), 컴퓨터A와 컴퓨터B의 승리 횟수 및 비긴 횟수를 저장할 변수(aWin, bWin, noWin)를 준비하기. '가위, 바위, 보'는 문자열 배열 strAry에 저장하기

```
import java.util.Arrays; // 코드 시작 부분에 입력
import java.util.Collections; // 코드 시작 부분에 입력

String[] toss = new String[10000];
String comA, comB;
int aWin, bWin, noWin;

String[] strAry = {"가위", "바위", "보"};
```

2. 두 컴퓨터가 내는 가위바위보를 무작위로 10000번 추출함

```
for (int i=0; i<10000; i++) {
    comA = strAry[(int)(Math.random()*strAry.length)];
    comB = strAry[(int)(Math.random()*strAry.length)];
    ~~~생략~~~
}
```

[LAB] 가위바위보 결과를 배열로 만들기

3. 두 컴퓨터가 가위바위보를 한 결과를 배열에 저장하기
   - 2번의 for문 안에 작성함

```
if (comA == "가위") {
    if (comB == "가위") {
        toss[i] = "없음";
    } else if (comB == "바위") {
        toss[i] = "B";
    } else if (comB == "보") {
        toss[i] = "A";
    }
} else if (comA == "바위") {
    ~~~생략~~~
} else if (comA == "보") {
    ~~~생략~~~
}
```

[LAB] 가위바위보 결과를 배열로 만들기

4. 가위바위보를 10000번 한 뒤 배열에서 두 컴퓨터의 승리 횟수와 비긴 횟수를 세어 출력하기.
   - Colloction.frequency(리스트, 값) 메서드: 리스트에서 해당 값의 개수를 계산함
   - Arrays.asList(배열) 메서드: 배열을 리스트로 변환함

```
aWin = Collections.frequency(Arrays.asList(toss), "A");
bWin = Collections.frequency(Arrays.asList(toss), "B");
noWin = Collections.frequency(Arrays.asList(toss), "없음");

System.out.println("10000번 중 컴퓨터A의 승리 : " + aWin);
System.out.println("10000번 중 컴퓨터B의 승리 : " + bWin);
System.out.println("10000번 중 비긴 경기 : " + noWin);
```

[실전 예제]
거북이들이
펼치는 쇼

[실전 예제] 거북이들이 펼치는 쇼

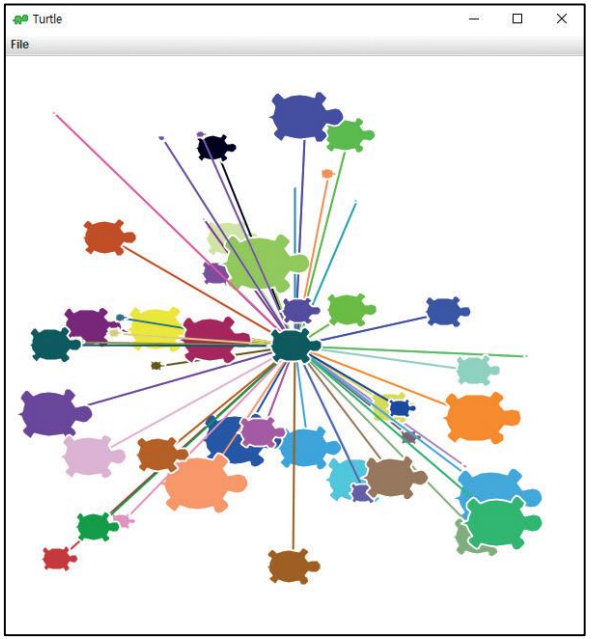
[문제]

거북이 50마리가 펼치는 쇼 프로그램을 만들어봅시다.

거북이 50마리를 2차원 배열로 생성하고, 이 거북이들이 화면 중앙에서 무작위 위치로 이동하게 합니다.

거북이 한 마리는 'X 위치, Y 위치, 크기, Red 색상, Green 색상, Blue 색상'의 1차원 배열로 구성하며, 50마리는 다음과 같은 2차원 배열이 됩니다.

	X 위치	Y 위치	크기	Red	Green	Blue
0	77	99	80	35	55	99
1	-232	184	3	120	57	3
2	333	255	10	135	78	203
3	7	176	3	120	57	3
4	84	1	22	23	166	199
⋮	⋮	⋮	⋮	⋮	⋮	⋮



[실전 예제] 거북이들이 펼치는 쇼

[해결]

Ex09_01.java

```
01  import java.awt.Color;
02  public class Ex09_01 {
03      public static void main(String[] args) {
04          int TURTLE_NUMBER = 50;
05          int WIDTH = 500, HEIGHT = 500;
06
07          Turtle turtle = new Turtle();
08          turtle.speed(100);
09          turtle.outlineColor("white");
10          turtle.setCanvasSize(WIDTH+100, HEIGHT+100);
11
12          // X 위치, Y 위치, 크기, Red, Green, Blue
13          int[][] turtleAry = new int[TURTLE_NUMBER][6];
14
15          for (int i=0; i<TURTLE_NUMBER; i++) {
16              turtleAry[i][0] = (int)(Math.random() * WIDTH - WIDTH/2);    // X 위치
17              turtleAry[i][1] = (int)(Math.random() * HEIGHT - HEIGHT/2);  // Y 위치
18              turtleAry[i][2] = (int)(Math.random() * 100);  // 크기
19              turtleAry[i][3] = (int)(Math.random() * 256);  // Red
20              turtleAry[i][4] = (int)(Math.random() * 256);  // Green
21              turtleAry[i][5] = (int)(Math.random() * 256);  // Blue
22          }
23
```

[실전 예제] 거북이들이 펼치는 쇼

[해결]

```
24          int x, y, size, r, g, b;
25          for (int i=0; i<TURTLE_NUMBER; i++) {
26              x =  turtleAry[i][0];
27              y = turtleAry[i][1];
28              size = turtleAry[i][2];
29              r = turtleAry[i][3];
30              g = turtleAry[i][4];
31              b = turtleAry[i][5];
32
33              turtle.fillColor(new Color(r, g, b));
34              turtle.penColor(new Color(r, g, b));
35
36              turtle.shapeSize(size,size);
37              turtle.down();
38              turtle.setPosition(x, y);
39              turtle.stamp();
40
41              turtle.up();
42              turtle.setPosition(0,0);
43          }
44      }
45  }
```

Thank you!