1. 배열 이해

I. 배열의 이해

- 배열은 하나씩 사용하던 종이 상자(변수) 를 한 줄로 붙여 놓은 것이다

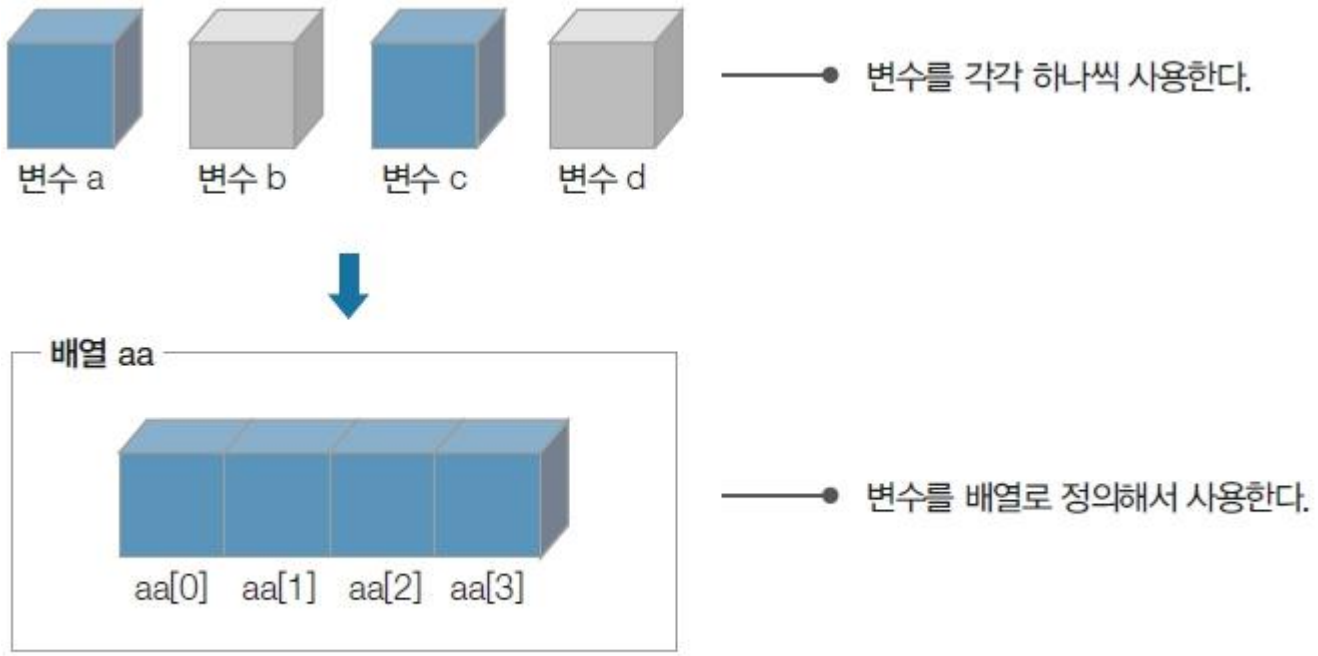

그림 7-3 배열의 개념

1. 배열 선언

배열 생성

- 배열이란 동일한 자료형을 가지는 데이터들의 연속된 집합을 의미
- C#에서는 배열을 참조형으로 제공되므로 new 키워드를 사용하여 다음과 같이 생성

```
자료형[] 배열명 = new 자료형[]{배열 요소};
```

- C#에서 선언하는 3가지 방법

```
자료형[] 배열명 = {요소1, 요소2, ... };
자료형[] 배열명 = new 자료형[]{요소1, 요소2, ...};
자료형[] 배열명 = new 자료형[n]{요소1, 요소2, ..., 요소n};
```

1. 배열 이해

I. 배열의 이해

- 예를 들어 4개의 변수를 담은 정수형 배열을 선언하면 다음과 같다.

```
int[ ] aa = new int[4];
// 또는
int[ ] aa;
aa = new int[4];
```

- 배열을 사용하지 않는다면 ① 각각의 변수를 int a, b, c, d와 같이 선언하여 사용하지만 배열을 사용하면 ② 첨자를 이용해 aa[0], aa[1], aa[2], aa[3]과 같이 나타낼 수 있다.

❶ 변수 선언

```
int  a, b, c, d;
a 사용
b 사용
c 사용
d 사용
```

❷ 배열 선언

```
int[ ] aa = new int[4];
aa[0] 사용
aa[1] 사용
aa[2] 사용
aa[3] 사용
```

I. 배열의 이해

배열에 값을 대입하고 출력하는 예 (프로젝트명 : array_01.cs)

```
1   static void Main(string[] args)
2   {
3       int[] aa = new int[4];
4       int hap;
5
6       Console.Write("1번째 숫자를 입력하세요 : ");
7       aa[0] = int.Parse(Console.ReadLine());
8       Console.Write("2번째 숫자를 입력하세요 : ");
9       aa[1] = int.Parse(Console.ReadLine());
10      Console.Write("3번째 숫자를 입력하세요 : ");
11      aa[2] = int.Parse(Console.ReadLine());
12      Console.Write("4번째 숫자를 입력하세요 : ");
13      aa[3] = int.Parse(Console.ReadLine());
14
15      hap = aa[0] + aa[1] + aa[2] + aa[3];
16
17      Console.WriteLine("합계 ==> {0}", hap);
18  }
```

1. 배열 이해

■ 배열에 값을 대입하고 출력하는 예 (프로젝트명 : array_02.cs)

```
int[] aa = new int[4];
int hap = 0;

for (int i = 0; i < 4; i++)
{
   Console.Write("{0}번째 숫자를 입력하세요 : ", i);
   aa[i] = int.Parse(Console.ReadLine());
}

   hap = aa[0] + aa[1] + aa[2] + aa[3]

Console.WriteLine();
Console.WriteLine("합계 => {0}", hap);
```

II. 배열의 활용

배열의 실제 활용

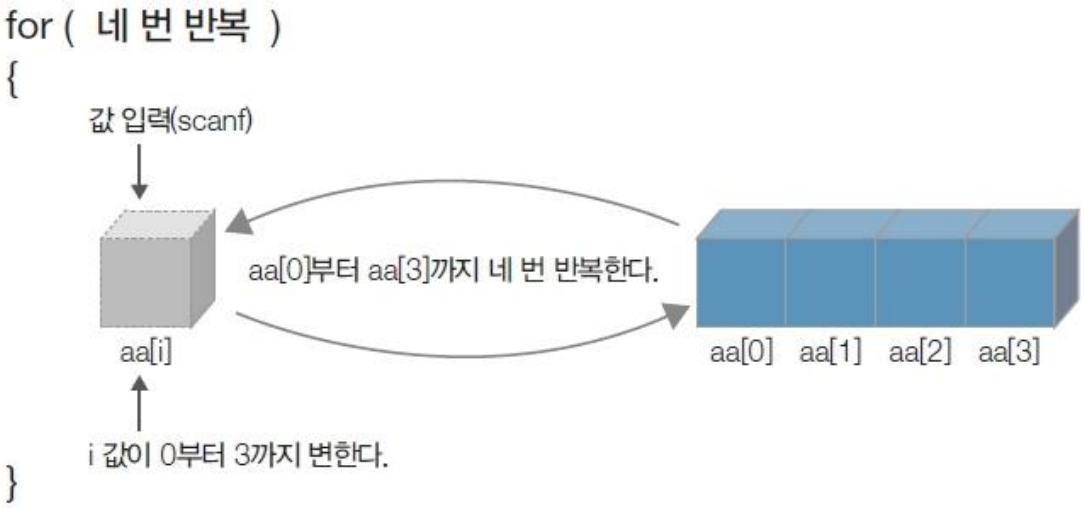

그림 7-4 for 문을 활용하여 배열 값 입력

- 그림을 해석하면 for 문을 네 번 반복하면서 aa[i]의 첨자를 aa[0]~aa[3]까지 변하게 한다는 뜻이다. 그림대로 실행되면 4개의 변수에 자동으로 값이 입력된다

1. 배열 선언

foreach 반복문

- 배열의 요소를 반복하는 단순하고 깔끔한 반복문
- 인덱스 0으로 시작하고 인덱스 n-1로 끝나는 인덱스 순서로 배열의 요소를 전부 처리

```
int[] numbers = {-1, 0, 1, 2, 3};
foreach(int cnt in numbers)
{
    Console.Write(cnt + " ");
}
// 출력결과 : -1, 0, 1, 2, 3
```

1. 배열 선언

- 다차원 배열의 경우 왼쪽 차원의 인덱스는 오른쪽 차원의 인덱스가 모두 증가한 다음 증가하는 방식으로 배열의 요소를 출력
- 3행 2열의 2차원 배열 요소를 foreach 문으로 출력할 때는 다음과 같이 선언

```
int[,] numbers2D[3, 2] = new int{{1, 3}, {5, 7}, {9, 11}};
foreach(int cnt in numbers2D)
{
    Console.Write(cnt + “ ”);
}
// 출력결과 : 1 3 5 7 9 11
```

1. 배열 선언

■ 배열 요소 출력

- 1차원 배열 요소를 foreach 반복문으로 출력하기 위해 다음 예제를 수행

예제 06-01 foreach 반복문으로 1차원 배열 요소 출력하기

• **Step 01** | **프로젝트 생성** : 프로젝트명은 'Array1D'로 입력합니다. 소스 파일명은 그대로 둡니다.

• **Step 02** | **소스 코드 입력** : 1차원 배열을 생성합니다. 그리고 foreach 반복문으로 배열의 요소를 출력하기 위해 다음과 같이 소스 코드를 입력합니다. 단축키 F5를 눌러 실행 결과를 확인합니다.

ch06/Array1D/Array1D/Program.cs

```
01  // See https://aka.ms/new-console-template for more information
02  int[] a = { -1, 0, 1, 2, 3 };
03  foreach (int cnt in a)
04  {
05      Console.Write(cnt + " ");
06  }
```

실행 결과

```
-1 0 1 2 3
```

1. 배열 선언

■ 배열 요소 출력

- 2차원 배열 요소를 foreach 반복문으로 출력하기 위해 다음 예제를 수행

예제 06-02 foreach 반복문으로 2차원 배열 요소 출력하기

• **Step 01 | 프로젝트 생성** : 프로젝트명은 'Array2D'로 입력합니다. 소스 파일명은 그대로 둡니다.

• **Step 02 | 소스 코드 입력** : 2차원 배열을 생성합니다. 그리고 foreach 반복문으로 배열의 요소를 출력하기 위해 다음과 같이 소스 코드를 입력합니다. 단축키 F5를 눌러 실행 결과를 확인합니다.

ch06/Array2D/Array2D/Program.cs

```
01  // See https://aka.ms/new-console-template for more information
02  int[,] b = new int[3, 2] { { 1, 3 }, { 5, 7 }, { 9, 11 } };
03  foreach(int cnt in b)
04  {
05      Console.Write(cnt + " ");
06  }
```

실행 결과

```
1 3 5 7 9 11
```

2. Array 클래스

C#의 모든 배열

- System.Array 클래스를 상속받아 구현해야 함
- 배열을 선언하게 되면 System.Array 클래스의 속성과 메서드를 사용할 수 있음

표 6-1 System.Array 클래스의 속성 및 메서드

속성 및 메서드	의미
Rank	배열의 차원 수를 반환
Length	배열 요소의 전체 개수를 반환
LongLength	배열 요소의 전체 개수를 64비트 정수값으로 반환
GetLength(n)	n 차원의 요소 개수를 반환 (n은 0부터 시작)
IsFixedSize	배열의 크기가 고정되어 있는지 여/부를 나타내는 값 반환
GetValue( )	인덱스 위치의 요소값을 반환
SetValue( )	인덱스 위치에 지정한 값을 저장
Array.Sort(배열명)	배열의 요소를 오름차순으로 정렬
Array.Clear(배열명)	배열을 초기화
Array.Reverse(배열명)	배열 요소의 순서를 역순으로 정렬
Clone( )	동일한 내용을 갖는 배열을 복사
Array.Copy.(a,n1,b,n2,len)	a 배열의 n1부터 len 길이만큼 b 배열의 n2에 복사
CopyTo(b,n)	현재 1차원 배열의 모든 요소를 b 배열의 n 인덱스 위치에 복사

예제 06-03 Array 클래스의 속성으로 배열의 차원 수 출력하기

• **Step 01** | **프로젝트 생성** : 프로젝트명은 'ArrayClass'로 입력합니다. 소스 파일명은 그대로 둡니다.

• **Step 02** | **소스 코드 입력** : 1차원 배열과 2차원 배열을 생성합니다. 그리고 Rank 속성을 지정하여 각 배열의 차원 수를 출력하기 위해 다음과 같이 소스 코드를 입력 합니다. 단축키 F5를 눌러 실행 결과를 확인합니다.

ch06/ArrayClass/ArrayClass/Program.cs

```
01  // See https://aka.ms/new-console-template for more information
02  int[] a = { 10, 50, 30 };
03  int[,] b = { { 1, 2, 3 }, { 4, 5, 6 } };
04
05  int d1 = a.Rank;
06  int d2 = b.Rank;
07
08  Console.WriteLine(" > 차원 수 : " + d1);
09  Console.WriteLine(" > 차원 수 : " + d2);
```

실행 결과

```
> 차원 수 : 1
> 차원 수 : 2
```

예제 06-04 Array 클래스의 메서드를 선언하여 배열 요소 정렬하기

• **Step 01 | 프로젝트 생성** : 프로젝트명은 'ArrayClassAD'로 입력합니다. 소스 파일명은 그대로 둡니다.

• **Step 02 | 소스 코드 입력** : 1차원 배열을 생성하고 Sort( ) 메서드를 선언하여 배열 요소를 오름차순으로 정렬합니다. 그리고 Reverse( ) 메서드를 선언하여 내림차순으로 정렬합니다. 오름차순과 내림차순으로 정렬된 배열 요소를 출력하기 위해 다음과 같이 소스 코드를 입력합니다. 단축키 F5를 눌러 실행 결과를 확인합니다.

ch06/ArrayClassAD/ArrayClassAD/Program.cs

```
01  // See https://aka.ms/new-console-template for more information
02  int[] a = { 10, 50, 30 };
03  Array.Sort(a);
04  Console.Write("> 오름차순 정렬 : ");
05
06  foreach (int cnt in a)
07  {
08      Console.Write(cnt + " ");
09  }
10
11  Console.WriteLine();
12  Array.Reverse(a);
13  Console.Write("> 내림차순 정렬 : ");
14
15  foreach (int cnt in a)
16  {
17      Console.Write(cnt + " ");
18  }
```

실행 결과

```
> 오름차순 정렬 : 10 30 50
> 내림차순 정렬 : 50 30 10
```

3. 다차원 배열

2차원 배열

- [행]과 [열]로 구성된 배열
- 2차원 배열부터는 가변 배열을 사용
- 가변 배열
  - 일반 다차원 배열로 선언했을 경우 메모리 공간의 낭비가 심해지거나
  - 다른 배열의 크기를 가져야만 수행이 가능한 경우
  - 사용할 수 있는 유용한 자료 구조

- 2차원 배열을 다음과 같이 생성할 수 있음

```
자료형[,] 배열명 = new 자료형[X, Y];     // 2차원 배열
자료형[][] 배열명 = new 자료형[X][Y];    // 2차원 배열 (가변 배열)
```

3. 다차원 배열

3차원 배열

- [면], [행], [열]로 구분되는 자료구조
- 3차원 배열을 다음과 같이 생성할 수 있음

```
자료형[,,] 배열명 = new 자료형[X,Y,Z];          // 3차원 배열
자료형[][][] 배열명 = new 자료형[X][Y][Z];      // 3차원 배열 (가변 배열)
```

예제 06-05 중첩 for 문으로 2차원 배열 요소 출력하기

• **Step 01** | **프로젝트 생성** : 프로젝트명은 'ArrayMultiD'로 입력합니다. 소스 파일명은 그대로 둡니다.

• **Step 02** | **소스 코드 입력** : 2차원 배열을 생성합니다. 그리고 중첩 for 문으로 배열의 요소를 출력하기 위해 다음과 같이 소스 코드를 입력합니다. 단축키 F5를 눌러 실행 결과를 확인합니다.

ch06/ArrayMultiD/ArrayMultiD/Program.cs

```
01  // See https://aka.ms/new-console-template for more information
02  Console.WriteLine(" > 2차원 배열 : arrA[2, 3]");
03  int[,] arrA = new int[2,3] {{10, 20, 30}, {40, 50, 60 }};
04
05  for(int i = 0; i < 2; i++)
06  {
07      for(int j = 0; j < 3; j++)
08      {
09          Console.Write("{0,5}", arrA[i,j]);    // 5자리로 출력
10      }
11      Console.WriteLine();
12  }
```

실행 결과

```
> 2차원 배열 : arrA[2,3]
   10   20   30
   40   50   60
```

예제 06-06 중첩 for 문으로 2차원 가변 배열 요소 출력하기

• **Step 01 | 프로젝트 생성** : 프로젝트명은 'ArrayMultiJD'로 입력합니다. 소스 파일명은 그대로 둡니다.

• **Step 02 | 소스 코드 입력** : 2차원 가변 배열을 생성합니다. 중첩 for 문으로 가변 배열의 요소를 출력하기 위해 다음과 같이 소스 코드를 입력합니다. 단축키 F5를 눌러 실행 결과를 확인합니다.

```
01  // See https://aka.ms/new-console-template for more information
02  Console.WriteLine(" > 2□□ □□ □□ : arrB[2][3]");
03
04  int[][] arrB = new int[2][];
05  arrB[0] = new int[] { 100, 200 };
06  arrB[1] = new int[] { 300, 400, 500 };
07
08  for (int i = 0; i < 2; i++)
09  {
10      Console.Write(" arrB[{0}] : ", i);
11
12      for (int j = 0; j < arrB[i].Length; j++)
13      {
14          Console.Write("{0}  ", arrB[i][j]);
15      }
16      Console.WriteLine();
17  }
```

2차원 가변 배열

실행 결과

2차원 가변 배열

```
> 2차원 배열 : arrB[2][3]
 arrB[0] :  100  200
 arrB[1] :  300  400  500
```

✔ Check Point | 합성 서식 문자열

합성 서식 문자열과 개체 목록은 합성 서식 지정 기능을 지원하는 메서드의 인수로 사용됩니다. 합성 서식 문자열은 0개 이상의 고정 텍스트가 하나 이상의 서식 항목과 결합된 형태로 구성됩니다. Console.Write(" arrB[{0}] : ", i);과 같이 선언한 출력문에서 중괄호로 표기한 {0}은 인덱스 0을 의미하고 {1}은 인덱스 1, {2}는 인덱스 2를 의미하여 문자열을 출력하게 됩니다.

4. Random 클래스

객체 생성

- C#에서 랜덤값을 사용하려면 Random 클래스를 사용하여 객체를 생성해야 함
- new 키워드를 사용하여 Random 클래스의 새로운 객체를 다음과 같이 생성

```
Random r = new Random( );    // Random 클래스의 객체 r 생성
```

4. Random 클래스

- 랜덤값은 숫자가 차례대로 정렬되어 있지 않고 무작위로 추출하고자 할 때 사용하며
- 예를 들면 로또 복권은 1~45까지의 숫자를 사용하고 주사위는 1~6까지의 숫자를 무작위로 추출할 때 랜덤값을 사용함
- Random 클래스에서 제공하는 Next( ) 메서드를 선언하게 되면 다음과 같이 피라미터의 개수를 지정할 수 있음

```
int x = r.Next( );              // 0~2,147,483,647 사이의 랜덤 정수값을 반환
int y = r.Next(100);            // 0~99 사이의 랜덤 정수값을 반환
int z = r.Next(1, 7);           // 1~6 사이의 랜덤 정수값을 반환
double d = r.NextDouble( );     // 0.0 이상 1.0 미만의 랜덤 실수값을 반환
Byte[] b = r.NextBytes( );      // 배열 전체를 0~255까지의 랜덤 숫자로 채움
```

예제 06-07 Random 클래스의 객체를 생성하여 6개의 랜덤 숫자 출력하기

• **Step 01 | 프로젝트 생성** : 프로젝트명은 'RandomClass'로 입력합니다. 소스 파일명은 그대로 둡니다.

• **Step 02 | 소스 코드 입력** : Random 클래스의 객체를 생성합니다. 그리고 Next( ) 메서드를 사용하여 1~45까지 숫자 중 랜덤 숫자 중 6개를 출력하기 위해 다음과 같이 소스 코드를 입력합니다. 단축키 F5를 눌러 실행 결과를 확인합니다.

ch06/RandomClass/RandomClass/Program.cs

```
01  // See https://aka.ms/new-console-template for more information
02  Random r = new Random();
03  int[] a = new int[6];
04
05  // 6개의 랜덤값 생성하여 저장
06  for (int i = 0; i < 6; i++)
07  {
08      a[i] = r.Next(1, 46);
09  }
10
11  Console.WriteLine(" > 생성된 6개의 랜덤 숫자 출력 ");
12  foreach (int value in a)
13  {
14      Console.Write("{0, 8}", value);  // 8자리로 출력
15  }
```

실행 결과

```
> 생성된  6개의 랜덤 숫자 출력
      31      16      17      6      11      34
```

실행 결과

```
> 생성된  6개의 랜덤 숫자 출력
      3      44      34      13      35      36
```

예제 06-08 랜덤값에 대한 최대값과 최소값 출력하기

• **Step 01 | 프로젝트 생성** : 프로젝트명은 'RandomMaxMin'로 입력합니다. 소스 파일명은 그대로 둡니다.

• **Step 02 | 소스 코드 입력** : 랜덤값을 배열에 저장합니다. 최대값과 최소값을 출력하기 위해 다음과 같이 소스 코드를 입력합니다. 단축키 F5를 눌러 실행 결과를 확인합니다.

```
01  // See https://aka.ms/new-console-template for more information
02  Random r = new Random();
03  int[] v = new int[10];
04
05  // 10개의 랜덤값 저장
06  for (int i = 0; i < v.Length; i++)
07      v[i] = r.Next(100);     // 0~99까지 랜덤값
08
09  Console.WriteLine(" > 랜덤값 출력 ");
10  foreach (int a in v)
11      Console.Write("{0, 5}", a);
12  Console.WriteLine();
13
14  // 랜덤값 중 최대값
15  int max = v[0];
16  for(int i = 1; i < v.Length; i++)
17      if(v[i] > max)
18          max = v[i];
```

```
19  Console.Write(" > 최대값 : ");
20  Console.WriteLine("{0, 3}", max);
21
22  // 랜덤값 중 최소값
23  int min = v[0];
24  for (int i = 1; i < v.Length; i++)
25      if (v[i] < min)
26          min = v[i];
27  Console.Write(" > 최소값 : ");
28  Console.WriteLine("{0, 3}", min);
```

실행 결과

```
 > 랜덤값 출력
   18    8   79    5   75   94   51   59   25   48
 > 최대값 :  94
 > 최소값 :   5
```

실행 결과

```
> 생성된  6개의 랜덤 숫자 출력
      31      16      17      6      11      34
```

실행 결과

```
> 생성된  6개의 랜덤 숫자 출력
      3      44      34      13      35      36
```

예제 06-09 랜덤값에 대한 총점과 평균 출력하기

• **Step 01 | 프로젝트 생성** : 프로젝트명은 'RandomSumAverage'로 입력합니다. 소스 파일명은 그대로 둡니다.

• **Step 02 | 소스 코드 입력** : 랜덤값을 배열에 저장합니다. 그리고 총점과 평균을 산출하여 출력하기 위해 다음과 같이 소스 코드를 입력합니다. 단축키 F5 를 눌러 실행 결과를 확인합니다.

ch06/RandomSumAverage/RandomSumAverage/Program.cs

```
01  // See https://aka.ms/new-console-template for more information
02  Random r = new Random();
03  int[] v = new int[13];
04
05  // 13개의 랜덤값 저장
06  for (int i = 0; i < v.Length; i++)
07      v[i] = r.Next(100);     // 0~99까지 랜덤값
08
09  Console.WriteLine(" > 랜덤값 출력 ");
10  foreach (int a in v)
11      Console.Write("{0, 5}", a);
12  Console.WriteLine();
13
14  // 랜덤값 총점
15  int sum = 0;
16  for (int i = 0; i < v.Length; i++)
17      sum += v[i];
18  Console.Write(" > 총점 : ");
```

```
19  Console.WriteLine("{0, 3}", sum);
20
21  // 랜덤값 평균
22  double avg = 0;
23  avg = (double)sum / v.Length;
24
25  Console.Write(" > 평균 : ");
26  Console.WriteLine("{0, 3}", avg);
```

실행 결과

```
> 랜덤값 출력
  83   55    3   84   77   83   67   27   74   50   91   64   16
> 총점 : 774
> 평균 : 59.53846153846154
```

예제 06-10 NextDouble( ) 메서드로 랜덤 실수값 출력하기

• **Step 01 | 프로젝트 생성** : 프로젝트명은 'RandomDouble'로 입력합니다. 소스 파일명은 그대로 둡니다.

• **Step 02 | 소스 코드 입력** : NextDouble( ) 메서드를 사용하여 랜덤 실수값을 출력하기 위해 다음과 같이 소스 코드를 입력합니다. 단축키 F5를 눌러 실행 결과를 확인합니다.

ch06/RandomDouble/RandomDouble/Program.cs

```
01  // See https://aka.ms/new-console-template for more information
02  Random r = new Random();
03
04  Console.WriteLine(r.NextDouble());
05  Console.WriteLine(r.NextDouble() * 10);
06  Console.WriteLine(r.NextDouble() * 20);
```

실행 결과

```
0.3786097386536993
2.458708122357253
13.198773775026858
```

■ 순차탐색

- 배열 요소의 처음부터 끝까지 모든 요소를 찾고자 하는 키값과 비교하여 그 값의 인덱스를 반환

예제 06-11 랜덤값에 대한 순차탐색 수행하기

• **Step 01 | 프로젝트 생성** : 프로젝트명은 'RandomSequentialSearch'로 입력합니다. 소스 파일명은 그대로 둡니다.

• **Step 02 | 소스 코드 입력** : 랜덤 숫자를 배열에 저장합니다. 그리고 순차탐색을 수행하기 위해 다음과 같이 소스 코드를 입력합니다. 단축키 F5를 눌러 실행 결과를 확인합니다.

ch06/RandomSequentialSearch/RandomSequentialSearch/Program.cs

```
01  // See https://aka.ms/new-console-template for more information
02  Random r = new Random();
03  int[] v = new int[10];
04
05  // 10개의 랜덤값 저장
06  for (int i = 0; i < v.Length; i++)
07      v[i] = r.Next(100);     // 0~99까지 랜덤값
08
09  Console.WriteLine(" > 정렬 전 랜덤값 ");
10  foreach (int a in v)
11      Console.Write("{0, 5}", a);
12
13  Console.WriteLine();
14  Console.WriteLine();
15
16  Console.WriteLine(" > 정렬 후 랜덤값 ");
17  Array.Sort(v);
18  foreach (int a in v)
19      Console.Write("{0, 5}", a);
```

```
20
21  Console.WriteLine();
22  Console.WriteLine();
23
24  Console.Write(" > 순차탐색할 숫자 입력 : ");
25  int choice = int.Parse(Console.ReadLine());
26  int count = 0;    // 탐색 횟수
27
28  for (int i = 0; i < v.Length - 1; i++)
29  {
30      count++;
31      if(v[i] == choice)
32      {
33          Console.WriteLine(" > 인덱스 : [{0}] = {1}", i, choice);
34          Console.WriteLine(" > 탐색 횟수 : {0}회", count);
35          break;
36      }
37  }
```

실행 결과

```
> 정렬 전 랜덤값
  46    88    69    77    80    68    74    98    22    35

> 정렬 후 랜덤값
  22    35    46    68    69    74    77    80    88    98

> 순차탐색할 숫자 입력 : 74
> 인덱스 : [5] = 74
> 탐색 횟수 : 6회
```

5. 탐색과 정렬

이진탐색

- 이진탐색을 수행하기 위해서는 반드시 배열 요소가 정렬되어 있어야 함
- 만약 배열 요소가 정렬되어 있지 않다면 이진탐색을 수행할 수 없으므로 주의해야 함
- 배열의 중간값과 키값을 비교하여 중간값보다 키값이 크다면 아래쪽 자료는 검색하지 않음
- 한 번 탐색을 수행할 때마다 탐색할 배열 요소가 $\frac{1}{2}$씩 줄어들기 때문에 탐색속도가 순차탐색에 비해 빠르게 수행됨

예제 06-12 랜덤값에 대한 이진탐색 수행하기

• **Step 01 | 프로젝트 생성** : 프로젝트명은 'RandomBinarySearch'로 입력합니다. 소스 파일명은 그대로 둡니다.

• **Step 02 | 소스 코드 입력** : 랜덤값을 배열에 저장합니다. 그리고 이진탐색을 수행하기 위해 다음과 같이 소스 코드를 입력합니다. 단축키 F5를 눌러 실행 결과를 확인합니다.

```
01  // See https://aka.ms/new-console-template for more information
02  Random r = new Random();
03  int[] v = new int[10];
04
05  // 10개의 랜덤값 저장
06  for (int i = 0; i < v.Length; i++)
07      v[i] = r.Next(100);     // 0~99까지 랜덤값
08
09  Console.WriteLine(" > 정렬 전 랜덤값 ");
10  foreach (int a in v)
11      Console.Write("{0, 5}", a);
12
13  Console.WriteLine();
14  Console.WriteLine();
15
16  Console.WriteLine(" > 정렬 후 랜덤값 ");
17  Array.Sort(v);
18  foreach (int a in v)
19      Console.Write("{0, 5}", a);
```

```
20
21  Console.WriteLine();
22  Console.WriteLine();
23
24  Console.Write(" > 이진탐색할 숫자 입력 : ");
25  int choice = int.Parse(Console.ReadLine());
26
27  int count = 0;     // 탐색 횟수
28  int low = 0;
29  int high = v.Length - 1;
30
31  while(low <= high)
32  {
33      count++;
34      int mid = (low + high) / 2;    // 중간 위치 산출
35
36      if (choice == v[mid])
```

```
37      {
38          Console.WriteLine(" > 인덱스 : [{0}] = {1}", mid, choice);
39          Console.WriteLine(" > 탐색 횟수 : {0}회", count);
40          break;
41      }
42      else if (choice > v[mid])  // 중간 배열 요소값보다 클 경우
43          low = mid + 1;
44      else    // 값을 찾지 못하거나 중간 배열 요소값보다 크지 않을 경우
45          high = mid - 1;  // high 변수값을 새로운 값으로 설정
46  }
```

실행 결과

```
> 정렬 전 랜덤값
  94   51   42   16   69   40    7   57   31   46

> 정렬 후 랜덤값
   7   16   31   40   42   46   51   57   69   94

> 이진탐색할 숫자 입력 : 94
> 인덱스 : [9] = 94
> 탐색 횟수 : 4회
```

버블정렬

- 인접한 2개의 배열 요소를 비교하여 더 큰 수를 뒤로 보내는 과정으로 배열 요소를 정렬

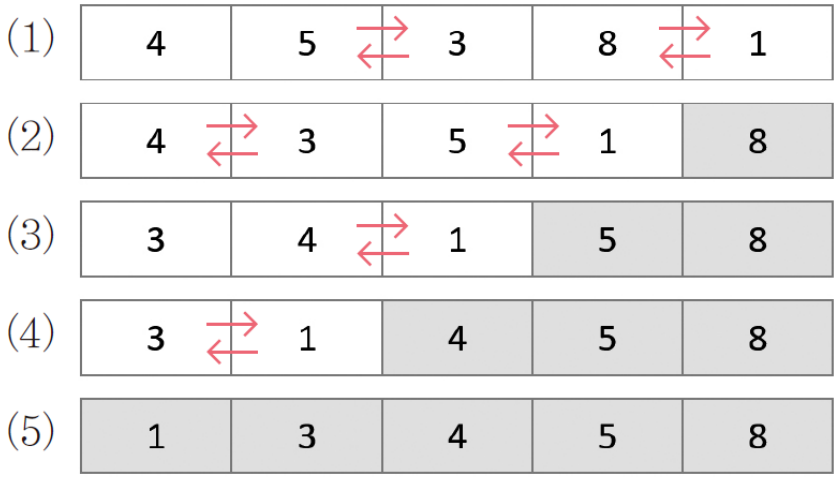

그림 6-1 버블정렬 수행 과정

예제 06-13 배열 요소에 대한 버블정렬 수행하기

• **Step 01 | 프로젝트 생성** : 프로젝트명은 'ArrayBubbleSort'로 입력합니다. 소스 파일명은 그대로 둡니다.

• **Step 02 | 소스 코드 입력** : 배열 요소에 대한 버블정렬을 수행하기 위해 다음과 같이 소스 코드를 입력합니다. 단축키 F5를 눌러 실행 결과를 확인합니다.

ch06/ArrayBubbleSort/ArrayBubbleSort/Program.cs

```
01 // See https://aka.ms/new-console-template for more information
02 int[] v = { 4, 5, 3, 8, 1 };
03
04 Console.WriteLine(" > 주어진 배열 요소 ");
05 foreach (var i in v)
06     Console.Write("{0, 5}", i);
07
08 Console.WriteLine();
09 Console.WriteLine();
10
11 Console.WriteLine(" > 버블정렬 수행 과정 ");
12 for(int i = 4; i > 0; i--)
13 {
14     for(int j = 0; j < i; j++)
15         if(v[j] > v[j + 1])
16         {
17             int t = v[j];
```

```
18                v[j] = v[j + 1];
19                v[j + 1] = t;
20            }
21
22        foreach (var s in v)
23            Console.Write("{0, 5}", s);
24        Console.WriteLine();
25    }
```

실행 결과

> 주어진 배열 요소

```
    4    5    3    8    1
```

> 버블정렬 수행 과정

```
    4    3    5    1    8
    3    4    1    5    8
    3    1    4    5    8
    1    3    4    5    8
```

자신감 뿜뿜!

C# 프로그래밍 Hard Carry

Thank You