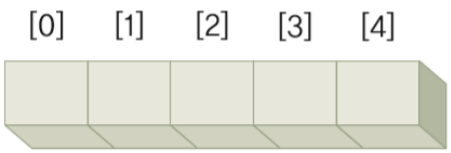
배열

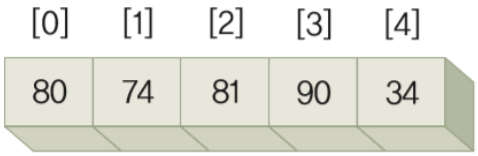
UPDATE 2023

01. 배열

- 배열(Array)이란 이름 하나로 같은 데이터 형식을 여러 개 보관해 놓은 데이터의 집합체이다. 배열은 요소들의 순서 있는 집합이며 각 요소는 인덱스로 접근이 가능하다.
- 배열을 사용하면 데이터를 모아서 관리할 수 있다는 특징이 있으며 배열에서 값 하나는 요소(element) 또는 항목(Item)이라고 한다.




```
int[] scores = new int[5];
scores[0] = 80;
scores[1] = 74;
scores[2] = 81;
scores[3] = 90;
scores[4] = 34;
```



01. 배열

- 배열의 종류는 3가지 종류가 있다.
  - 1차원 배열: 배열의 첨자를 하나만 사용하는 배열이다.
  - 다차원 배열: 첨자 2개 이상을 사용하는 배열(2차원, 3차원, ...)이다.
  - 가변(jagged) 배열: '배열의 배열' 이라고도 하며, 이름 하나로 다양한 차원의 배열을 표현할 때 사용한다.

```
> int[] numbers;
> numbers = new int[3];
```

```
> int[] intArray = new int[3];
```

02. 배열 선언 및 초기화

- 배열 선언 및 초기화 1

  데이터형식[ ] 배열이름 = new 데이터형식[크기숫자n];

  배열이름[인덱스번호] = 값;

```
int[] scores = new int[5];
scores[0] = 80;
scores[1] = 74;
scores[2] = 81;
scores[3] = 90;
scores[4] = 34;
```

02. 배열 선언 및 초기화

```
// 정수형 배열길이 정의
int[] numArr1 = new int[6];
Console.WriteLine(numArr1);

// 배열값 지정
// 배열명[인덱스번호] = 값;
// 인덱스번호는 0번부터 시작
numArr1[0] = 100;
numArr1[1] = 200;
numArr1[2] = 50;
numArr1[3] = 67;
numArr1[4] = 34;
numArr1[5] = 67;
```

```
Console.WriteLine("==============");
Console.WriteLine($"1번째 배열 값은? {numArr1[0]}");
Console.WriteLine($"2번째 배열 값은? {numArr1[1]}");
Console.WriteLine($"3번째 배열 값은? {numArr1[2]}");
Console.WriteLine($"4번째 배열 값은? {numArr1[3]}");
Console.WriteLine($"5번째 배열 값은? {numArr1[4]}");
Console.WriteLine($"6번째 배열 값은? {numArr1[5]}");
```

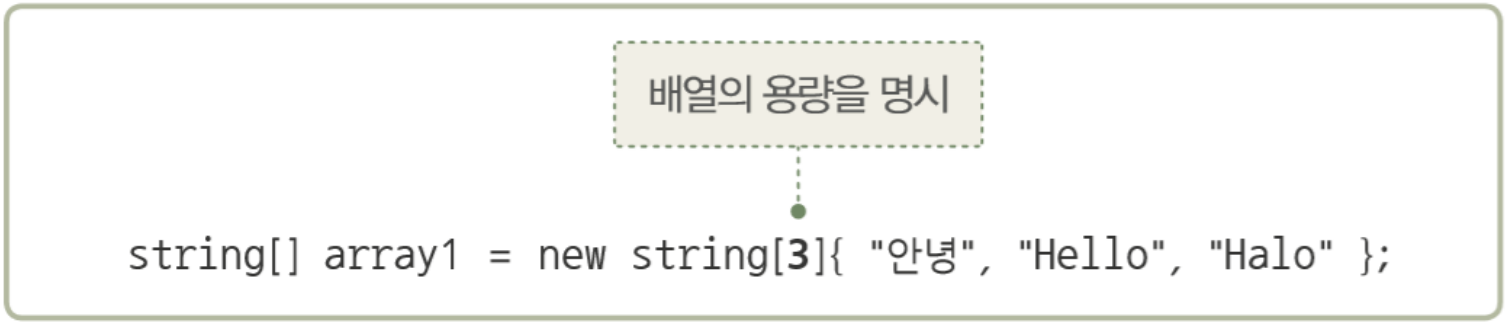
02. 배열 선언 및 초기화

```
// for 문을 이용하여 배열값 전체 가로로 출력하기
// 배열명.Length : 배열의 전체 갯수

for (int i = 0; i < numArr1.Length; i++)
{
    Console.Write($"{i + 1}번째 => {numArr1[i]}    ");
}
```

02. 배열 선언 및 초기화

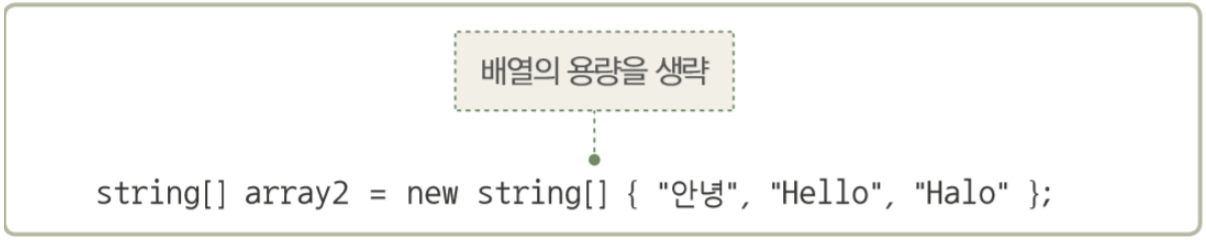
- 배열 선언 및 초기화 2 – 배열 크기 명시 후 컬렉션 초기자 { } 사용
  데이터형식[ ] 배열이름 = new 데이터형식[크기숫자n] { 값1, 값2 ... };




```
// 데이타형[] 배열명 = new 데이타형[길이] { 값1, 값2 ... }
int[] moneyArr = new int[4] { 10000, 1000, 500, 5000 };
for (int i = 0; i < moneyArr.Length; i++)
{
    Console.WriteLine(moneyArr[i]);
}
```

02. 배열 선언 및 초기화

- 배열 선언 및 초기화 3 – 배열 크기를 명시하지 않고 컬렉션 초기자 { } 사용
  데이터형식[ ] 배열이름 = new 데이터형식[] { 값1, 값2 … };



- 배열 선언 및 초기화 4 – 컬렉션 초기자 { } 사용

```
string[] array3 = { "안녕", "Hello", "Halo" };
```

02. 배열 선언 및 초기화

```
// 데이타형[] 배열명 = new 데이타형[] { 값1, 값2 ... }
string[] fruitArr = new string[] { "바나나", "포도", "오렌지" };
for (int i = 0; i < fruitArr.Length; i++)
{
    Console.WriteLine(fruitArr[i]);
}
```

```
// 데이타형[] 배열명 = { 값1, 값2 ... }
string[] nameArr = { "최", "선우", "박", "신", "이" };
for (int i = 0; i < nameArr.Length; i++)
{
    Console.WriteLine(nameArr[i]);
}
```

03. 배열 길이

- 배열명.length : 배열 속성으로 배열의 길이를 반환한다.
- 배열명.GetLength(차원번호n) : 차원에 해당하는 길이를 반환한다. 0은 1차원

```
char[] cArr = { 'a', 'b', 'c', 'd', 'e' };
Console.WriteLine(cArr);
Console.WriteLine(cArr.GetType());
Console.WriteLine($"배열 전체 길이는? {cArr.Length}");
Console.WriteLine($"배열 전체 길이는? {cArr.GetLength(0)}");
for (var i = 0; i < cArr.Length; i++)
{
    Console.WriteLine($"{i}  : {cArr[i]}");
}
```

```
abcde
System.Char[]
배열 전체 길이는? 5
배열 전체 길이는? 5
0  : a
1  : b
2  : c
3  : d
4  : e
```

퀴즈

- 배열로 정의되어 있는 점수의 총점과 평균이 출력되도록 프로그래밍 하여라

```
int[] scoreArr = { 65, 77, 55 };
```

```
총점 = 197
평균 = 65.000
```

퀴즈

- 배열로 정의되어 있는 점수의 총점과 평균이 출력되도록 프로그래밍 하여라

```
int[] scoreArr = { 65, 77, 55 };
```

```
총점 = 197
평균 = 65.000
```

퀴즈

- 국가 이름과 관련된 countryArr 배열을 선언하고 입력 값이 저장되도록 프로그래밍 하여라.

```
string[] countryArr = new string[7];
```

```
-----------------
    입력화면
-----------------
0번째 배열값 = > 프랑스
1번째 배열값 = > 브라질
2번째 배열값 = > 싱가폴
3번째 배열값 = > 대한민국
4번째 배열값 = > 영국
5번째 배열값 = > 중국
6번째 배열값 = > 미국
```

```
-----------------
    출력화면
-----------------
=> 프랑스
=> 브라질
=> 싱가폴
=> 대한민국
=> 영국
=> 중국
=> 미국
```

퀴즈

- 국가 이름과 관련된 countryArr 배열을 선언하고 입력 값이 저장되도록 프로그래밍 하여라.

04. foreach 와 배열 출력

- foreach 구문을 이용하면 배열 값 모두를 순차적으로 출력할 수 있다

```
foreach (var 아이템변수 in 배열명)
{
        명령문
}
```

04. foreach 와 배열 출력

```
string[] cityArr = { "서울", "부산", "대전", "대구" };

for (int i = 0; i < cityArr.Length; i++)
{
    Console.WriteLine(cityArr[i]);
}
```

```
foreach (var item in cityArr)
{
    Console.WriteLine(item);
}
```

04. foreach 와 배열 출력

```
// foreach 구문으로 배열합 구하기
float[] fArr = { 10.5f, 20.1f, 30.2f };
float sum = 0.0f;
foreach (float f in fArr)
{
    Console.Write($" {f} ");
    sum += f;
}
Console.WriteLine($"\n 배열의 합은? {sum} ");
```

퀴즈

- 배열을 정의하고 출력화면과 같이 표시되도록 프로그래밍하여라.
- 전체 출력은 foreach 문이나 while, for 반복문을 이용하여야 한다

```
string[] nameArr = { "민호", "수진", "석진", "남준", "태형" };
```

```
----------------------
      출력화면
----------------------
첫번째 배열값 출력 : 민호
세번째 배열값 출력 : 석진
배열 모두 출력하기
=> 민호
=> 수진
=> 석진
=> 남준
=> 태형
```

퀴즈

- 배열의 길이를 입력 받고 입력 받은 길이만큼 배열에 값을 추가하도록 프로그래밍 하여라

```
-------------------------
      입력화면
-------------------------
배열 길이 입력 => 3

0 배열값 => 바나나
1 배열값 => 포도
2 배열값 => 수박
```

```
---------------------
      출력화면
---------------------
바나나
포도
수박
```

퀴즈

- 시험 점수를 입력 받아 합계와 평균을 구한 후 학점을 출력하여라.
- 평균은 소숫점 3째자리까지 출력한다.
- 시험과목은 별도의 배열로 생성하여 사용한다.

```
int[] scoreArr = new int[5];
string[] subTitle = { "국어", "영어", "수학", "과학", "한문" };
```

퀴즈

입력화면

국어 점수 = > 78
영어 점수 = > 90
수학 점수 = > 56
과학 점수 = > 88
한문 점수 = > 90

출력화면

국어 점수 = > 78
영어 점수 = > 90
수학 점수 = > 56
과학 점수 = > 88
한문 점수 = > 90
합계 = > 402
평균 = > 80.000
학점 = > B

05. 다차원 배열

- 2차원 배열 및 3차원 배열처럼 차원이 2 이상인 배열을 다차원 배열이라고 한다.
- 데이터형식[,] 배열이름 = new 데이터형식[2차원길이, 1차원길이];

```
int[,] arr = new int[2, 3]{ {1, 2, 3}, {4, 5, 6} };     // 배열의 형식과 길이를 명시
int[,] arr2 = new int[,] { { 1, 2, 3 }, { 4, 5, 6 } }; // 배열의 길이를 생략
int[,] arr3 = { { 1, 2, 3 }, { 4, 5, 6 } };            // 형식과 길이를 모두 생략
```

- 2차원 배열의 인덱싱

arr[0,0]	arr[0,1]	arr[0,2]
arr[1,0]	arr[1,1]	arr[1,2]

05. 다차원 배열

- 3차원 배열의 인덱싱

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

05. 다차원 배열

```
// 2행 3열짜리 2차원 배열 만들기
int[,] arrInt = { { 10, 20, 30 }, { 400, 500, 100 } };

Console.WriteLine("1행 1열 배열값은? {0}", arrInt[0, 0]);
Console.WriteLine("1행 3열 배열값은? {0}", arrInt[0, 2]);
Console.WriteLine("2행 2열 배열값은? {0}", arrInt[1, 1]);
Console.WriteLine("2행 3열 배열값은? {0}", arrInt[1, 2]);
```

```
for (int i = 0; i < 2; i++)
{
    for (int j = 0; j < 3; j++)
    {
        Console.WriteLine($" {i} , {j} => {arrInt[i, j]} ");
    }
}
```

```
1행 1열 배열값은? 10
1행 3열 배열값은? 30
2행 2열 배열값은? 500
2행 3열 배열값은? 100
 0 , 0 => 10
 0 , 1 => 20
 0 , 2 => 30
 1 , 0 => 400
 1 , 1 => 500
 1 , 2 => 100
```

05. 다차원 배열

```
// 3행 4열 배열 생성 = 음식메뉴로 배열 만들기
string[,] foodArr = {
                        {"라면","짜장면","순대","짬뽕" },
                        {"우동","초밥","샐러드","볶음밥"},
                        {"돈까스","짜장면","삼겹살","짬뽕" }
                    };

Console.WriteLine($"{foodArr[0, 1]} / {foodArr[2, 1]}");
Console.WriteLine(foodArr[2, 2]);
Console.WriteLine(foodArr[0, 2]);
```

05. 다차원 배열

```
for (int i = 0; i < 3; i++)
{
    for (int j = 0; j < 4; j++)
    {
        Console.WriteLine($"{i}, {j} => {foodArr[i, j]}");
    }
    Console.WriteLine("=================");
}
```

```
=================
0, 0 => 라면
0, 1 => 짜장면
0, 2 => 순대
0, 3 => 짬뽕
=================
1, 0 => 우동
1, 1 => 초밥
1, 2 => 샐러드
1, 3 => 볶음밥
=================
2, 0 => 돈까스
2, 1 => 짜장면
2, 2 => 삼겹살
2, 3 => 짬뽕
=================
```

05. 다차원 배열

```
// 전체 배열값 출력하기 - foreach구문 이용
Console.WriteLine("\n\n\n==================");
int cnt = 1;
foreach (var item in foodArr)
{
    Console.WriteLine($"{cnt} : {item}");
    // 4개마다 줄 삽입
    if (cnt % 4 == 0)
    {
        Console.WriteLine("==================");
    }
    cnt++;
}
```

```
==================
1 : 라면
2 : 짜장면
3 : 순대
4 : 짬뽕
==================
5 : 우동
6 : 초밥
7 : 샐러드
8 : 볶음밥
==================
9 : 돈까스
10 : 짜장면
11 : 삼겹살
12 : 짬뽕
==================
```

05. 다차원 배열

```
// 3행3열 배열 만들기
int[,] int2Darr =  {
                        { 1, 2, 3 },
                        { 4, 5, 6 },
                        { 7, 8, 9 }
                   };

for (int i = 0; i < 3; i++)
{
    for (int j = 0; j < 3; j++)
    {
        Console.Write($"\t{int2Darr[i, j]}");
    }
    Console.WriteLine();
}
```

```
1       2       3
4       5       6
7       8       9
```

05. 다차원 배열

```
foreach (var item in int2Darr)
{
    Console.Write($"\t{item}");
}
```

```
1	2	3	4	5	6	7	8	9
```

```
// 3의 배수라면 줄바꿈 명령어 삽입
int cnt2 = 1;
foreach (var item in int2Darr)
{
    Console.Write($"\t{item}");
    if (cnt2 % 3 == 0)
    {
        Console.WriteLine();
    }
    cnt2++;
}
```

```
1	2	3
4	5	6
7	8	9
```

퀴즈

- 아래의 코드를 이용하여 2차원 배열을 생성하고 출력화면과 같이 출력하도록 프로그래밍 하여라.

```
string[,] userInfo = { {"홍길동","gildong"," 남"},
                       {"신은주","eunju67"," 여"},
                       {"김수홍","su7836"," 남"},
                       {"최민호","minho6"," 남"},
                       {"이수진","sujun77"," 여"}
                     };
```

```
--------------------
     출력화면
--------------------
이 름   아이디   성 별

홍길동  gildong   남
신은주  eunju67   여
김수홍  su7836    남
최민호  minho6    남
이수진  sujun77   여
```

05. 다차원 배열

- 배열의 크기를 할당하고 배열 값은 인덱스를 각각 호출하여 지정하는 방식이다.
- 자료형[,] 배열명 = new 자료형[행수,열수];
- 배열명[행인덱스, 열인덱스] = 값;

```
// 2행2열 배열의 크기 생성
string[,] firstNameArr = new string[2, 2];
firstNameArr[0, 0] = "박씨";
firstNameArr[0, 1] = "고씨";
firstNameArr[1, 0] = "옹씨";
firstNameArr[1, 1] = "최씨";
for (int i = 0; i < 2; i++)
{
    for (int j = 0; j < 2; j++)
    {
        Console.Write($"{firstNameArr[i, j]}\t");
    }
    Console.WriteLine();
}
```

05. 다차원 배열

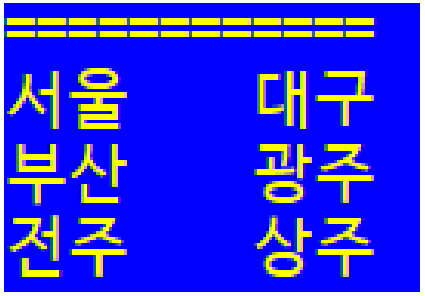
```
//3행 2열로 구성된 배열을 선언후 값 입력값으로 지정하기
string[,] city2d = new string[3, 2];

int number = 1;
for (int i = 0; i < 3; i++)
{
    for (int j = 0; j < 2; j++)
    {
        // Console.Write($"입력값{j+1} => ");
        Console.Write($"입력값{number} => ");
        number++;
        city2d[i, j] = Console.ReadLine();
        Console.WriteLine();
    }
}
```

```
입력값1 => 서울
입력값2 => 대구
입력값3 => 부산
입력값4 => 광주
입력값5 => 전주
입력값6 => 상주
```

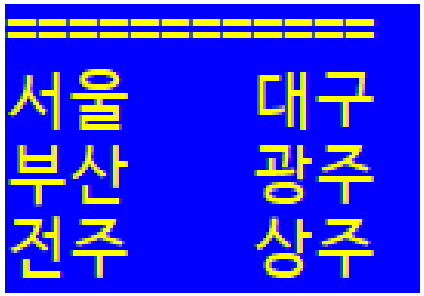
05. 다차원 배열

```
for (int i = 0; i < 3; i++)
{
    for (int j = 0; j < 2; j++)
    {
        Console.Write($"{city2d[i, j]}\t");
    }
    Console.WriteLine();
}
```



05. 다차원 배열

```
int n = 1;
foreach (var item in city2d)
{
    Console.Write($"{item}\t");
    if (n % 2 == 0)
    {
        Console.WriteLine();
    }
    n++;
}
```



퀴즈

- 5의 배수로 구성된 5행 5열의 2차원 배열을 생성하고 출력하여라

```
int[,] num2DArr = new int[5, 5];
```

```
출력화면
  5   10   15   20   25
 30   35   40   45   50
 55   60   65   70   75
 80   85   90   95  100
105  110  115  120  125
```

퀴즈

- 3행 3열의 2차원 배열을 생성하고 행과 열이 같으면 1, 다르면 0이 배열값으로 삽입되도록 프로그래밍하여라

```
int[,] num2DArr = new int[5, 5];
```

퀴즈

- 학생 3명의 국어와 영어 점수로 구성되어 있는 3행 2열의 배열에 합계와 평균 값을 추가시켜 3행 4열 배열로 변경되도록 프로그래밍 하여라.

- 배열의 정의는 아래를 이용한다.

```
int[,] middle_scores =
{
    {55, 88}, {80,78}, {95, 77}
};
string[] student_names = { "최은우", "마동탁", "박지민" };
float[,] middle_scores2 = new float[3, 4];
```

퀴즈

출력화면1

	국어	영어
최은우	55	88
마동탁	80	78
박지민	95	77

출력화면2

학생명	국어	영어	총점	평균
최은우	55.00	88.00	143.00	71.50
마동탁	80.00	78.00	158.00	79.00
박지민	95.00	77.00	172.00	86.00

열거형(enum)

UPDATE 2023

01. 열거형(enum) 형식

- 열거형(enumeration) 형식은 기억하기 어려운 상수들을 기억하기 쉬운 이름 하나로 묶어 관리하는 표현 방식이다
- enum 키워드를 사용하며 이늄 또는 이넘으로 읽는다.
- 열거형을 사용하면 값 리스트 여러 개를 이름 하나로 관리할 수 있다는 특징이 있다.
- 같은 범주에 속하는 여러 개의 상수를 선언할 때 사용하기도 한다.

```
enum 열거형이름
{
    열거형변수1,
    열거형변수2,
    열거형변수3
}
```

```
enum 열거형이름
{
    열거형변수1 = 기본값1,
    열거형변수2 = 기본값2,
    열거형변수3 = 기본값3
}
```

01. 열거형(enum) 형식

- 기본 자료형은 int 이며 기반 자료형은 byte, sbyte, short, ushort, int, unint, long, ulong, char 을 사용할 수 있다.
- 열거형이름.열거형변수로 호출할 수 있다.

열거형이름 변수 = 열거형이름.열거형변수

- 요일 정보에 대한 열거형 선언

```
enum Weekday
{
    Sunday,     Monday,    Tuesday,    Wednesday,    Thursday,    Friday,    Saturday
}
```

01. 열거형(enum) 형식

- 열거형을 선언하면 기본적으로 Align.Top은 0, Align.Bottom은 1, Align.Left는 2, Align. Right는 3의 상수 값을 설정할 수 있다.

```
enum Align
{
    Top = 0,
    Bottom = 2,
    Left = 4,
    Right = 8
}
```

- 커피 사이즈에 대한 열거형 선언

```
enum CoffeeSize
{
    Small,    Medium,    Large
}
```

02. 열거형(enum) 생성 및 할당

```
namespace day4_3
{
    // namespace 안, class 위에 선언한다.
    // 열거형으로 선언
    // 성별 => 남자, 여자
    참조 6개
    enum Gender
    {
        Male,
        Female
    }
```

```
static void Main(string[] args)
{
    //==================================
    // 열거형 값을 출력
    Console.WriteLine(Gender.Male);   // Male
    Console.WriteLine(Gender.Female); // Female
    Console.WriteLine("====================");

    // 열거형 변수 선언하기
    // 열거형이름 열거형변수 =  열거형이름.열거형상수;
    Gender m = Gender.Male;
    Gender f = Gender.Female;

    // 출력문등에서 열거형변수로 사용가능
    Console.WriteLine(m);   // Male
    Console.WriteLine(f); // Female
```

퀴즈

- 사계절과 관련된 데이타를 열거형 형태로 선언하고 출력화면과 같이 출력하여라.

```
enum Season
{
    Spring,
    Summer,
    Autumn,
    Winter
}
```

출력화면

```
------------------
01. Spring
02. Summer
03. Autumn
04. Winter
```

03. 열거형 항목에 상수 값 주기

```
// 커피 사이즈를 열거형으로 선언
// Tall, Medium , Short
참조 13개
enum CoffeeSize
{
    Tall = 100,
    Medium = 200,
    Short = 300
}
```

```
CoffeeSize c_t = CoffeeSize.Tall;
CoffeeSize c_m = CoffeeSize.Medium;
CoffeeSize c_s = CoffeeSize.Short;
Console.WriteLine(c_t);
Console.WriteLine(c_m);
Console.WriteLine(c_s);
Console.WriteLine($"커피사이즈 {c_t} , {c_m}, {c_s}");
Console.WriteLine("커피사이즈 {0},{1},{2}", c_t, c_m, c_s);
```

03. 열거형 항목에 상수 값 주기

```
..
// 열거형 상수 문자열값을 숫자로 변경하기
Console.WriteLine("\n\n==============");
Console.WriteLine($"{CoffeeSize.Tall} => {(int)CoffeeSize.Tall}");
Console.WriteLine($"{c_t} => {(int)c_t}");
Console.WriteLine($"{CoffeeSize.Short} => {Convert.ToInt32(CoffeeSize.Short)}");
Console.WriteLine($"{c_s} => {Convert.ToInt32(c_s)}");
```

```
Tall => 100
Tall => 100
Short => 300
Short => 300
```

퀴즈

- 프로그래밍 언어를 열거형 상수로 선언한 후 숫자 값과 함께 다음과 같이 출력하여라.

```
------------------------ -
      출력화면
--------------------------
 Num     Enum
==================
 10      C_language
 20      C_Plus
 30      C_Sharp
 40      Python
 50      JAVA
```

04. 열거형 인덱스 순서 변경하기

- 열거형 인덱스의 기본값은 0, 1, 2이지만 새로운 값으로 설정할 수 있다.
- 만약 첫번째 요소에는 직접 인덱스의 값을 할당하고 두번째 요소부터는 값을 할당하지 않지 않는다면 첫번째 요소의 값에 이어 자동으로 인덱스 값이 할당된다.

```
enum DialogResult
{
 YES = 10,
 NO,
 CANCEL,
 CONFIRM = 50,
 OK
}
```

04. 열거형 인덱스 순서 변경하기

```
// 카페 메뉴 열거형 자료로 선언
참조 8개
enum Cafemenu
{
    // 첫번째 열거형 상수의 고유의 숫자 값을 지정하면
    // 뒤의 열거형 상수의 고유의 숫자 값은
    // 1씩 자동증가
    Coffee = 1,
    IceCoffee, // 2
    CafeLatte,  // 3
    Icecream    // 4
}
```

```
==========
 카페 메뉴
==========
1 .. Coffee
2 .. IceCoffee
3 .. CafeLatte
4 .. Icecream
```

```
Console.WriteLine("\t카페 메뉴");
Console.WriteLine("==========");
Console.WriteLine($"{(int)Cafemenu.Coffee} .. {Cafemenu.Coffee}");
Console.WriteLine($"{(int)Cafemenu.IceCoffee} .. {Cafemenu.IceCoffee}");
Console.WriteLine($"{(int)Cafemenu.CafeLatte} .. {Cafemenu.CafeLatte}");
Console.WriteLine($"{(int)Cafemenu.Icecream} .. {Cafemenu.Icecream}");
```

04. 열거형 인덱스 순서 변경하기

```
// 인덱스 상수값이 자동으로 1씩 증가됨
참조 6개
enum DialogResult
{
    YES = 10,
    NO,
    CANCEL,
    CONFIRM = 50,
    OK
}
```

```
DialogResult.YES => 10
DialogResult.NO => 11
DialogResult.CANCEL => 12
DialogResult.CONFIRM => 50
DialogResult.OK => 51
```

```
Console.WriteLine($"DialogResult.YES => {(int)DialogResult.YES}");
Console.WriteLine($"DialogResult.NO => {(int)DialogResult.NO}");
Console.WriteLine($"DialogResult.CANCEL => {(int)DialogResult.CANCEL}");
Console.WriteLine($"DialogResult.CONFIRM => {(int)DialogResult.CONFIRM}");
Console.WriteLine($"DialogResult.OK => {(int)DialogResult.OK}");
```

05. 열거형과 switch 문

- 열거형 값을 switch 문에 대입하면 자동으로 열거형 항목에 해당하는 case 문을 만드는 기능을 제공한다.

```
enum Animal { Chicken, Dog, Pig }
```

\ 실행 결과 /

```
개
```

```
Animal animal = Animal.Dog;
switch (animal)
{
    case Animal.Chicken:
        Console.WriteLine("닭");
        break;
    case Animal.Dog:
        Console.WriteLine("개");
        break;
    case Animal.Pig:
        Console.WriteLine("돼지");
        break;
    default:
        Console.WriteLine("기본값 설정 영역");
        break;
}
```

05. 열거형과 switch 문

```
// 카페 메뉴
참조 6개
enum Item
{
    Coffee = 1,
    Tea,
    Icecream,
    Bread
}
```

```
// 열거형 변수로 지정
Item item;
item = Item.Coffee;
int qty = 10;
switch (item)
{
    case Item.Coffee:
        Console.WriteLine($"커피 {qty}잔 주문합니다.");
        break;
    case Item.Tea:
        Console.WriteLine($"홍차 {qty}잔 주문합니다.");
        break;
    case Item.Icecream:
        Console.WriteLine($"아이스크림 {qty}개 주문합니다.");
        break;
    case Item.Bread:
        Console.WriteLine($"토스트 {qty}인분 주문합니다.");
        break;
    default:
        break;
}
```