윈도우프로그래밍

4. C# 기본(2)

2023. 3. 24

심미나

목 차

참고자료: 수업교재(C# 프로그래밍, 2판)

V. 복합 대입 연산자

복합 대입 연산자

복합 대입 연산자

- 자료형에 적용하는 기본 연산자와 = 연산자를 함께 사용
  - a+=10 은 a=a+10 을 뜻함

① 숫자에 적용하는 복합 대입 연산자

연산자	설명
+=	숫자 덧셈 후 대입 연산자
-=	숫자 뺄셈 후 대입 연산자
*=	숫자 곱셈 후 대입 연산자
/=	숫자 나눗셈 후 대입 연산자

② 문자열에 적용하는 복합 대입 연산자

연산자	설명
+=	문자열 연결 후 대입 연산자

복합 대입 연산자

복합 대입 연산자

- (예제 2-17) 숫자와 관련된 복합 대입 연산자(교재 91p)
  - 코드2-37. 숫자와 관련된 복합 대입 연산자

```
01    static void Main(string[] args)
02    {
03        int output = 0;
04        output += 52;
05        output += 273;
06        output += 103;
07
08        Console.WriteLine(output);
09    }
10
11
12
13
14
15
16
17
18
19
20
```

output = output + 52;
output = output + 273;
output = output + 103; 과 동일

실행 결과

```
428
```

복합 대입 연산자

복합 대입 연산자

- (예제 2-18) 문자와 관련된 복합 대입 연산자(교재 92p)
  - 코드 2-39. 문자열과 관련된 복합 대입 연산자

```
01    static void Main(string[] args)
02    {
03        string output = "hello ";
04        output += "world ";
05        output += "!";
06
07        Console.WriteLine(output);
08    }
09
10
11
12
13
14
15
16
17
18
19
20
```

output = output + “world ”;
output = output + “!”; 과 동일

실행 결과

```
hello world !
```

VI. 증감 연산자

증감 연산자

증감 연산자 사용

- 단항 연산자로 변수 앞과 뒤에 ++ 기호와 -- 기호 붙여 만듬

연산자	설명
[변수]++	기존 변수의 값에 1을 더합니다(후위).
++[변수]	기존 변수의 값에 1을 더합니다(전위).
[변수]--	기존 변수의 값에서 1을 뺍니다(후위).
--[변수]	기존 변수의 값에서 1을 뺍니다(전위).

증감 연산자

증감 연산자 사용

- (예제 2-19) 증감 연산자(교재 93p)
  - 코드 2-41. 증감 연산자

```
01    static void Main(string[] args)
02    {
03        int number = 10;
04        number++;
05        Console.WriteLine(number);
06        number--;
07        Console.WriteLine(number);
08    }
09
10
11
12
13
14
15
16
17
18
19
20
```

실행 결과

```
11
10
```

증감 연산자

증감 연산자 사용

- (예제 2-20) 증감 연산자의 전위와 후위 (교재 94p)
  - 코드 2-42(1). 증감 연산자의 후위 형태

```
01    static void Main(string[] args)
02    {
03        // 후위
04        int number = 10;
05        Console.WriteLine(number);
06        Console.WriteLine(number++);
07        Console.WriteLine(number--);
08        Console.WriteLine(number);
09    }
10
11
12
13
14
15
16
17
18
19
20
```

실행 결과

```
10
10
11
10
```

증감 연산자

증감 연산자 사용

- (예제 2-20) 증감 연산자의 전위와 후위 (교재 94p)
  - 코드 2-42(2). 증감 연산자의 전위 형태

```
01    static void Main(string[] args)
02    {
03        // 전위
04        // int number = 10;
05        Console.WriteLine(number);
06        Console.WriteLine(++number);
07        Console.WriteLine(--number);
08        Console.WriteLine(number);
09    }
10
11
12
13
14
15
16
17
18
19
20
```

실행 결과

```
10
11
10
10
```

증감 연산자

증감 연산자 사용

- 전위(Prefix)
  - 해당 문장을 실행하기 전에 값을 변경
  - 예) Console.WriteLine(++number)
    => 변수 number 1을 더한 후, Console.WriteLine(number)를 실행
    => number += 1; Console.WriteLine(number); 과 같음

- 후위(Postfix)
  - 문장을 실행한 이후에 값을 변경
  - 예) Console.WriteLine(number++)
    => Console.WriteLine(number)를 실행한 후, 변수 number 1을 더함
    => Console.WriteLine(number); number += 1; 과 같음

  - 코드2-44, 코드2-45 각자 확인

VII. 자료형 검사

자료형 검사

자료형

- 자료형 검사 방법
  - 방법1 : 마우스 가져다 대기

```
int number = 10;
Console.WriteLine(number);
```

(지역 변수) int number

  - 방법2 : GetType( ) 메서드(프로그램 내부에서 자료형 확인)

메서드	설명
GetType()	해당 변수의 자료형을 추출합니다.

    - 변수뿐만 아니라 숫자 또는 문자열에 직접 적용 가능

자료형 검사

자료형

- (예제 2-21) GetType( ) 메서드 활용(교재 98p)
  - 코드2-46. GetType()메서드 활용

```
01  namespace TypeCheck
02  {
03      class Program
04      {
05          static void Main(string[] args)
06          {
07              // 변수를 선언합니다.
08              int _int = 273;
09              long _long = 522731033265;
10              float _float = 52.273F;
11              double _double = 52.273;
12              char _char = '글';
13              string _string = "문자열";
14
15              // 출력합니다.
16              Console.WriteLine(_int.GetType());
17              Console.WriteLine(_long.GetType());
18              Console.WriteLine(_float.GetType());
19              Console.WriteLine(_double.GetType());
20              Console.WriteLine(_char.GetType());
21              Console.WriteLine(_string.GetType());
22          }
23      }
24  }
```

변수.GetType() 형태

실행 결과

```
System.Int32
System.Int64
System.Single
System.Double
System.Char
System.String
```

자료형 검사

자료형

- (예제 2-22) 직접적인 GetType( ) 메서드 활용(교재 93p)
  - 코드2-47. 직접적인 GetType()메서드 활용

```
01  namespace DirectTypeCheck
02  {
03      class Program
04      {
05          static void Main(string[] args)
06          {
07              Console.WriteLine((273).GetType());
08              Console.WriteLine((522731033265L).GetType());
09              Console.WriteLine((52.273F).GetType());
10              Console.WriteLine((52.273).GetType());
11              Console.WriteLine(('자').GetType());
12              Console.WriteLine(("문자열").GetType());
13          }
14      }
15  }
16
17
18
19
20
```

(특정값).GetType() 형태

실행 결과

```
System.Int32
System.Int64
System.Single
System.Double
System.Char
System.String
```

VIII. var 키워드

var 키워드

var 키워드 사용

- C#에서는 var키워드로 변수의 자료형을 자동으로 지정할 수 있음

var 키워드	설명
var	자료형을 자동으로 지정합니다.

```
01  static void Main(string[] args)
02  {
03      var number = 100;
04  }
```

- var키워드의 제약
  - 한 번 지정된 자료형은 계속 유지
  - int 자료형으로 선언된 변수를 string 자료형으로 바꾸는 것은 불가능

```
01  var number = 10.2;
02  number = "변경";
```

var 키워드

var 키워드 추가 사용 조건

- 지역 변수로 선언
  - 지역 변수 : 메서드 내부에 선언되어 있는 변수

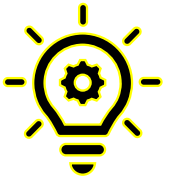
```
01 class Program
02 {
03     var global = 52;          ← 인스턴스 변수(var 키워드 사용 불가)
04
05     static void Main(string[] args)
06     {
07         var local = 273;      ← 지역 변수(var 키워드 사용 가능)
08     }
09 }
```

- 변수를 선언과 동시에 초기화

```
01 static void Main(string[] args)
02 {
03     var number = 20;
04 }
```

var 키워드

(참고) var 키워드 선언



- 정수 선언 시
  - var number = 100 입력, int 자료형으로 선언
- 실수 선언 시
  - var number = 10.0 입력, double 자료형으로 선언
- long 자료형, float 자료형 선언 시
  - 숫자 뒤에 L, F 등 기호 붙여야 함

  - 예) (코드2-54) var키워드를 사용한 다양한 자료형 선언

```
static void Main(string[] args)
{
    var numberA = 100L;   // long 자료형
    var numberB = 100.0;  // double 자료형
    var numberC = 100.0F; // float 자료형
}
```

IX. 입력

입력

- C#에서 사용자의 입력을 받을 때 Console.ReadLine() 메서드를 사용
  - 입력 메서드

메서드 이름	설명
Console.ReadLine()	사용자로부터 한 줄의 문자열을 입력 받습니다.

- 문자열 입력과 출력
  - Console.ReadLine( ) 메서드는 문자열만 입력 가능
    - 입력 메서드의 반환형: 자동완성 기능을 통해 확인하면 string? 으로 표시
  - 숫자를 입력 받아 더하는 등의 코드를 만들려면 문자열을 숫자로 바꾸는 방법 필요

입력

- (예제 2-23) 문자열 입력과 출력 (교재 105p)
  - 코드 2-55. 문자열 입력과 출력

```
01   namespace Input
02   {
03       class Program
04       {
05           static void Main(string[] args)
06           {
07               string input = Console.ReadLine();
08               Console.WriteLine("input: " + input);
09           }
10       }
11   }
12
13
14
15
16
17
18
19
20
```

ReadLine()메서드를 사용한 결과는 string 자료형이므로 string 변수에 저장

실행 결과

```
아무 것이나 입력
input: 아무 것이나 입력
```

X. 자료형 변환

자료형 변환

자료형 변환

- 자료형 변환이란, 한 자료형을 다른 자료형으로 바꾸는 것
  - 예)

```
01 static void Main(string[] args)
02 {
03     // long 자료형을 int 자료형으로 변환합니다.
04     long longNumber = 2147483647L + 2147483647L;
05     int intNumber = longNumber;
06     Console.WriteLine(intNumber);
07 }
```

자료형 변환

강제 자료형 변환

- 자료형 변환 실패

```
static void Main(string[] args)
{
    // long 자료형을 int 자료형으로 변환합니다.
    long longNumber = 2147483647L + 2147483647L;
    int intNumber = longNumber;
    Console.WriteL(
}
```

(지역 변수) long longNumber

CS0266: 암시적으로 'long' 형식을 'int' 형식으로 변환할 수 없습니다. 명시적 변환이 있습니다. 캐스트가 있는지 확인하세요.

잠재적 수정 사항 표시 (Alt+Enter 또는 Ctrl+.)

  - 예) 강제 자료형 변환

```
01  var a = (int)10.0;
02  var b = (float)10;
03  var c = (double)10;
```

자료형 변환

강제 자료형 변환

- (예제 2-24) 강제 자료형 변환 (교재 107p)
  - 코드 2-58. 강제 자료형 변환

```
01   namespace ExplicitConversion
02   {
03       class Program
04       {
05           static void Main(string[] args)
06           {
07               // long 자료형을 int 자료형으로 변환합니다.
08               long longNumber = 2147483647L + 2147483647L;
09               int intNumber = (int)longNumber;
10               Console.WriteLine(intNumber);
11           }
12       }
13   }
14
15
16
17
18
19
20
```

long자료형을 int자료형으로 명시적 형변환

실행 결과

```
-2
```

자료형 변환

강제 자료형 변환

- 강제 자료형 변환 데이터 손실 발생하지 않는 예

```
01 static void Main(string[] args)
02 {
03     // long 자료형을 int 자료형으로 변환합니다.
04     long longNumber = 52273;
05     int intNumber = (int)longNumber;
06     Console.WriteLine(intNumber);
07 }
```

자료형 변환

자동 자료형 변환

- (예제 2-25) 숫자 손상(교재 108p)
  - 코드2-60. 숫자 손상

```
01    namespace NumberLost
02    {
03        class Program
04        {
05            static void Main(string[] args)
06            {
07                // long 자료형을 int 자료형으로 변환
08                long longNumber = 2147483647L + 2147483647L;
09                int longToInt = (int)longNumber;
10                Console.WriteLine(longToInt);
11
12                // double 자료형을 int 자료형으로 변환
13                double doubleNumber = 52.27310332;
14                int doubleToInt = (int)doubleNumber;
15                Console.WriteLine(doubleToInt);
16            }
17        }
18    }
19
20
```

Long형을 int형으로 형변환
데이터 손실이 일어날 수 있음

double형을 int형으로 형변환
데이터 손실이 일어날 수 있음

실행 결과

```
-2
52
```

자료형 변환

자동 자료형 변환

- 자동 자료형 변환

기존 자료형	자동 변환되는 자료형
int	long, float, double
long	float, double
char	int, long, float, double
float	double

자료형 변환

자동 자료형 변환

- (예제 2-26) 자동 자료형 변환(교재 109p)
  - 코드2-61. 자동 자료형 변환

```
01   namespace ImplicitConversion
02   {
03       class Program
04       {
05           static void Main(string[] args)
06           {
07               // int 자료형의 숫자를 생성합니다.
08               int intNumber = 2147483647;
09
10               // int 자료형을 long 자료형으로 자동 변환
11               long intToLong = intNumber;
12               Console.WriteLine(intToLong);
13
14               // int 자료형을 double 자료형으로 자동 변환
15               double intToDouble = intNumber;
16               Console.WriteLine(intToDouble);
17           }
18       }
19   }
20
```

자동 형변환

실행 결과

```
2147483647
2147483647
```

자료형 변환

다른 자료형을 숫자로 변환

- 문자열을 숫자로 변환하는 메서드
  - 다른 자료형을 int, long, float, double 자료형으로 변환

메서드 이름	설명
int.Parse()	다른 자료형을 int 자료형으로 변경합니다.
long.Parse()	다른 자료형을 long 자료형으로 변경합니다.
float.Parse()	다른 자료형을 float 자료형으로 변경합니다.
double.Parse()	다른 자료형을 double 자료형으로 변경합니다.

자료형 변환

다른 자료형을 숫자로 변환

- (예제 2-27) 문자열을 숫자로 변환(교재 111p)
  - 코드2-63. 문자열을 숫자로 변환

```
01  namespace StringTo
02  {
03      class Program
04      {
05          static void Main(string[] args)
06          {
07              Console.WriteLine(int.Parse("52"));
08              Console.WriteLine(long.Parse("273"));
09              Console.WriteLine(float.Parse("52.273"));
10              Console.WriteLine(double.Parse("103.32"));
11
12              Console.WriteLine(int.Parse("52").GetType());
13              Console.WriteLine(long.Parse("273").GetType());
14              Console.WriteLine(float.Parse("52.273").GetType());
15              Console.WriteLine(double.Parse("103.32").GetType());
16          }
17      }
18  }
19
20
```

문자열을 숫자로 변환

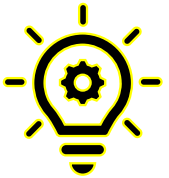
문자열을 숫자로 변환한 후 자료형의 타입을 확인

실행 결과

```
52
273
52.273
103.32
System.Int32
System.Int64
System.Single
System.Double
```

자료형 변환

(참고) FormatException 예외



- Parse()메서드를 사용할 때 해당 자료형으로 변환 불가능한 문자
  - 예) 숫자로 변환할 수 없는 문자열을 변환하는 경우

```
Console.WriteLine(int.Parse("52.273"));
Console.WriteLine(int.Parse("abc"));
```

자료형 변환

다른 자료형을 문자열로 변환

- ToString() 메서드
  - 다른 자료형을 문자열로 변환할 때 사용하는 메서드

메서드	설명
ToString()	문자열로 변환합니다.

  - 예) (10).ToString() : int자료형을 문자열로 변환
    (52.273).ToString() : double자료형을 문자열로 변환

자료형 변환

다른 자료형을 문자열로 변환

- (예제 2-28) 기본 자료형을 문자열로 변환(교재 112p)
  - 코드2-65. 기본 자료형을 문자열로 변환

```
01  namespace ToStringBasic
02  {
03      class Program
04      {
05          static void Main(string[] args)
06          {
07              Console.WriteLine((52).ToString());
08              Console.WriteLine((52.273).ToString());
09              Console.WriteLine(('a').ToString());
10              Console.WriteLine((true).ToString());
11              Console.WriteLine((false).ToString());
12
13              Console.WriteLine((52).ToString().GetType());
14              Console.WriteLine((52.273).ToString().GetType());
15              Console.WriteLine(('a').ToString().GetType());
16              Console.WriteLine((true).ToString().GetType());
17              Console.WriteLine((false).ToString().GetType());
18          }
19      }
20  }
```

정수, 실수, 문자, 불 자료형을 문자열로 변환

변환 후 자료형 타입 확인

실행 결과

```
52
52.273
a
True
False
System.String
System.String
System.String
System.String
System.String
```

자료형 변환

다른 자료형을 문자열로 변환

- (예제 2-29) 소숫점 제거(교재 113p)
  - 코드 2-66. 소수점 제거

```
01  namespace DoubleToString
02  {
03      class Program
04      {
05          static void Main(string[] args)
06          {
07              double number = 52.273103;
08              Console.WriteLine(number.ToString("0.0"));
09              Console.WriteLine(number.ToString("0.00"));
10              Console.WriteLine(number.ToString("0.000"));
11              Console.WriteLine(number.ToString("0.0000"));
12          }
13      }
14  }
15
16
17
18
19
20
```

잘려진 소수점 부분은 반올림됨

실행 결과

```
52.3
52.27
52.273
52.2731
```

자료형 변환

다른 자료형을 문자열로 변환

- (예제 2-30) 숫자와 문자열 덧셈(교재 114p)
  - 코드 2-67. 숫자와 문자열 덧셈

```
01  namespace StringPlusNumber
02  {
03      class Program
04      {
05          static void Main(string[] args)
06          {
07              Console.WriteLine(52 + 273);
08              Console.WriteLine("52" + 273);
09              Console.WriteLine(52 + "273");
10              Console.WriteLine("52" + "273");
11          }
12      }
13  }
14
15
16
17
18
19
20
```

숫자끼리 덧셈연산 수행

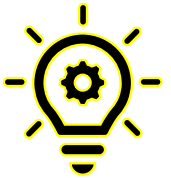
문자끼리, 또는 문자와 숫자간 문자열 연결 연산 수행

실행 결과

```
325
52273
52273
52273
```

자료형 변환

(참고) 간단한 문자열 변환



- 빈 문자열을 더해주어 문자열로 쉽게 변환 가능
  - 예)

```
int number = 52273;
string outputA = number + "";
Console.WriteLine(outputA);

char character = 'a';
string outputB = character + "";
Console.WriteLine(outputB);
```

- 2번째 코드의 경우, “”가 없으면 문자 문자열 변환 오류 발생

```
char character = 'a';
string output = character;
```

(지역 변수) char character

CS0029: 암시적으로 'char' 형식을 'string' 형식으로 변환할 수 없습니다.

자료형 변환

다른 자료형을 불로 변환

- Bool.Parse() 메서드
  - 다른 자료형을 불 자료형으로 변환할 때 사용하는 메서드
  - 예) 문자열을 불로 변환하는 메서드

메서드	설명
bool.Parse()	문자열을 불 자료형으로 변환합니다.

자료형 변환

다른 자료형을 불로 변환

- (예제 2-31) 문자열을 불로 전환(교재 116p)
  - 코드 2-69. 문자열을 불로 변환

```
01   namespace StringToBool
02   {
03       class Program
04       {
05           static void Main(string[] args)
06           {
07               Console.WriteLine(bool.Parse("True"));
08               Console.WriteLine(bool.Parse("true"));
09               Console.WriteLine(bool.Parse("False"));
10               Console.WriteLine(bool.Parse("false"));
11           }
12       }
13   }
14
15
16
17
18
19
20
```

문자열을 불 자료형으로 변환

실행 결과

```
True
True
False
False
```

XI. 실습 및 과제

과제

과제2

- 세부 과제
  - (교재 90~117) 예제 2-17, 2-20, 2-22, 2-25, 2-27, 2-29, 2-30
    - 각 예제의 코드를 작성하여 실행한 후, 실행결과 화면을 캡처한 이미지 파일을 제출
    - 각 예제의 세부 코드가 1개 이상일 경우, 교안의 코드로 제출하고 나머지는 자율 수행

- 제출 시 주의사항
  - 실행결과 이미지 캡처 시, 소스코드와 이미지가 겹치지 않게 모두 보이도록 할 것
  - 각 소스코드의 마지막줄에 “학과, 학년, 분반, 학번, 이름” 출력문 삽입
    - Console.WriteLine(“학과, 학번, 학년, 분반, 이름”); 추가하여 본인임을 증빙
  - 실행결과 이미지(.jpg 등)를 모두 압축하여 1개의 압축파일(.zip)로 제출
    - 파일명 “과제2_분반_학번_이름.zip”으로 제출 (세부 파일명은 예제번호 등 구분되게 임의로 지정)
  - 제출기한 : 2023년 3월 30일 자정까지(기한 외 추가제출은 기본적으로 불인정)
    - 사이버캠퍼스 오류로 마감일까지 제출 불가시에만 메일제출(시스템오류 입증할 화면 캡처 함께 제출)
    - mnshim@sungkyul.ac.kr (메일제목: 과제번호(분반,성명), 메일내용: 오류상황 설명 및 증빙)

감사합니다

mnshim@sungkyul.ac.kr