[강의교안 이용 안내]

학습목표

- C# 문법 구조에 대해 살펴봅니다.
- 문자와 문자열을 출력하는 방법을 알아봅니다.
- 인코딩과 디코딩에 대해 이해합니다.
- 주석문의 선언과 활용 방법에 대해 알아봅니다.

Chapter 02

C# 기본 문법 구조

1. C# 문법 구조

- **기본 소스 코드**
  - C#은 객체지향 프로그래밍 언어
  - 가장 기본적인 문법 구조는 다음과 같음

```
01  static void Main(string[ ] args)
02  {
03  Console.WriteLine("안녕하세요.");
04  }
```

1. C# 문법 구조

기본 문법 구조

static 선언

- 프로그램을 수행할 때 메모리 분배는 정적 방법을 적용하겠다는 의미
- 정적 방법은 프로그램이 컴파일을 수행할 때 메모리를 분배 받음
- 특정 유형의 대상을 생성하지 않고 프로그램을 종료

void 선언

- 메인 함수를 수행 후 리턴값 생략
- Return 문 또한 생략 가능
- 특정 값을 반환하고자 한다면 void 대신 int 또는 float를 선언하며 이 때는 반드시 return 문을 선언해 줘야 함 (선언하지 않으면 문법 오류)

1. C# 문법 구조

- Main( ) 선언
  - 메인 함수를 의미하며 프로그램의 시작 위치를 나타냄
  - 메인 함수는 반드시 1개만 선언해야 함
  - 2개 이상의 메인 함수 선언 시 문법 오류가 발생하게 됨

- string[ ] args 선언
  - Main( ) 함수의 괄호 안에 선언하는 매개 변수가 문자열 그룹임을 의미
  - args는 명령행의 매개 변수를 처리할 때 사용
  - 매개 변수는 프로그램을 실행할 때 인수를 전달하기 위한 매개체 역할을 수행함

1. C# 문법 구조

- Console.WriteLine( ) 선언
  - C#에서 가장 기본적인 출력을 위해 선언하는 소스 코드
  - WriteLine( ) 메서드의 괄호 안에 큰따옴표로 묶여 있는 문자열을 화면에 출력하기 위해 선언
  - Console.WriteLine( )을 선언한다는 것은 Console 클래스의 속성을 지닌 WriteLine( ) 메서드를 사용하겠다는 의미

- 세미콜론 선언
  - 세미콜론(;)은 해당 명령문의 마지막을 의미하는 마침표 역할을 수행
  - 만약 명령문의 끝에 세미콜론을 생략하게 되면 컴파일러는 아직 명령문이 끝나지 않았다는 의미로 해석
  - 그러므로 명령문의 끝에는 반드시 세미콜론을 선언해야 함

출력 메서드

- C#에서 사용하는 출력 메서드는 Console.WriteLine( ) 메서드
- 출력 메서드의 사용 형식과 의미는 다음 표와 같음

표 2-1 출력 메서드

메서드 이름	의미
Console.Write( )	데이터 출력 후 줄을 바꾸지 않음
Console.WriteLine( )	데이터 출력 후 줄을 바꿈

1. C# 문법 구조

신규 문법 구조

- Visual Studio 2022 버전에서는 C# 프로젝트를 생성할 때 아래와 같이 Main( ) 메서드 없이 Top Level Statement를 직접 사용
- Top Level Statement는 닷넷 플랫폼 9.0(2009 v6.9)에서 처음 도입
- 이 책에서는 Top Level Statement가 적용된 다음과 같은 신규 문법 구조를 사용

```
01  // See https://aka.ms/new-console-template for more information
02  Console.Write("문자 출력 : ");
03  Console.WriteLine('A');
04  Console.Write("문자열 출력 : ");
05  Console.WriteLine("반갑습니다.");
```

출력 형식 지정자

- 포맷 문자열에서 출력할 내용을 특정 형식으로 출력하기 위해 사용
- 출력 형식 지정자를 선언하는 문법 구조는 다음과 같음

```
{인덱스[,배치][:포맷문자열]}
```

표 2-2 출력 형식 지정자의 종류

형식 지정자	의미	소스 코드 선언	출력 결과
D	10진수	Console.WriteLine("{0:D}", 123)	123
E	지수	Console.WriteLine("{0:E}", 12345.6789)	1.234568E+004
F	고정소수점	Console.WriteLine("{0:F}", 123.456)	123.46
G	일반	Console.WriteLine("{0:G}", 123.456)	123.456
N	숫자	Console.WriteLine("{0:N}", 123456)	123,456.00
P	백분율	Console.WriteLine("{0:P}", 0.38)	38%
X	16진수	Console.WriteLine("0x{0:X}", 12345)	0x3039

2. 문자와 문자열

문자

- 유니코드 1개를 의미하며 작은따옴표('A')로 묶어 표현
- 세계 모든 언어와 기호에 값을 부여한 것을 유니코드라고 함
- 유니코드의 종류는 UTF-8, UTF-16, EUC-KR 등이 있음
- 우리나라에서는 UTF-8과 EUC-KR 유니코드를 주로 사용
- UTF-8은 유니코드를 인코딩하는 방식으로 8비트(1byte) 단위로 가변 인코딩을 수행하는 형식
- EUC-KR은 한글을 표현하기 위해 16비트(2byte) 단위로 인코딩을 수행하는 형식

2. 문자와 문자열

- 인코딩과 디코딩
  - 인코딩 : 문자를 2진 숫자 코드로 변환하는 과정
  - 디코딩 : 2진 숫자 코드를 문자로 변환하는 과정

그림 1-1 문자 처리를 위한 인코딩과 디코딩

- 문자로 인식하게 되는 경우

```
Console.WriteLine('A'); // 알파벳 A를 문자로 처리
Console.WriteLine('5'); // 5는 숫자가 아닌 문자로 처리
Console.WriteLine(7);   // 7은 숫자로 처리
```

2. 문자와 문자열

문자열

- 연속된 문자열을 “대한민국”과 같이 큰따옴표로 묶어 표현
- 문자열 또한 WriteLine( ) 메서드를 사용하여 출력
- 숫자에도 “8”과 같이 큰따옴표로 묶어 선언하게 되면 숫자가 아닌 문자열로 처리됨

```
Console.WriteLine("A"); // 알파벳 A를 문자열로 처리
Console.WriteLine("8"); // 8은 숫자가 아닌 문자로 처리
Console.WriteLine(9);  // 9는 숫자로 처리
```

예제 02-01 문자와 문자열 출력하기

- **Step 01 | 비주얼 스튜디오 실행** : Visual Studio 프로그램을 실행합니다.

- **Step 02 | 프로젝트 생성** : 프로젝트명은 'CSProgram'으로 입력합니다. 소스 파일명은 CSprint.cs로 변경합니다.

- **Step 03 | 소스 코드 입력** : 문자와 문자열을 출력하기 위해 다음과 같이 소스 코드를 입력합니다. 소스 코드 입력 후 단축키 F5를 눌러 실행 결과를 확인합니다.

ch02/CSProgram/CSProgram/CSprint.cs

```
01  // See https://aka.ms/new-console-template for more informatio
02  Console.Write("문자 출력 : ");
03  Console.WriteLine('A');
04  Console.Write("문자열 출력 : ");
05  Console.WriteLine("반갑습니다.");
```

실행 결과

문자 출력 : A

문자열 출력 : 반갑습니다.

• **Step 02** | **기존 프로젝트 소스 코드 확인** : 기존에 작성했던 프로젝트의 소스 코드를 확인합니다.

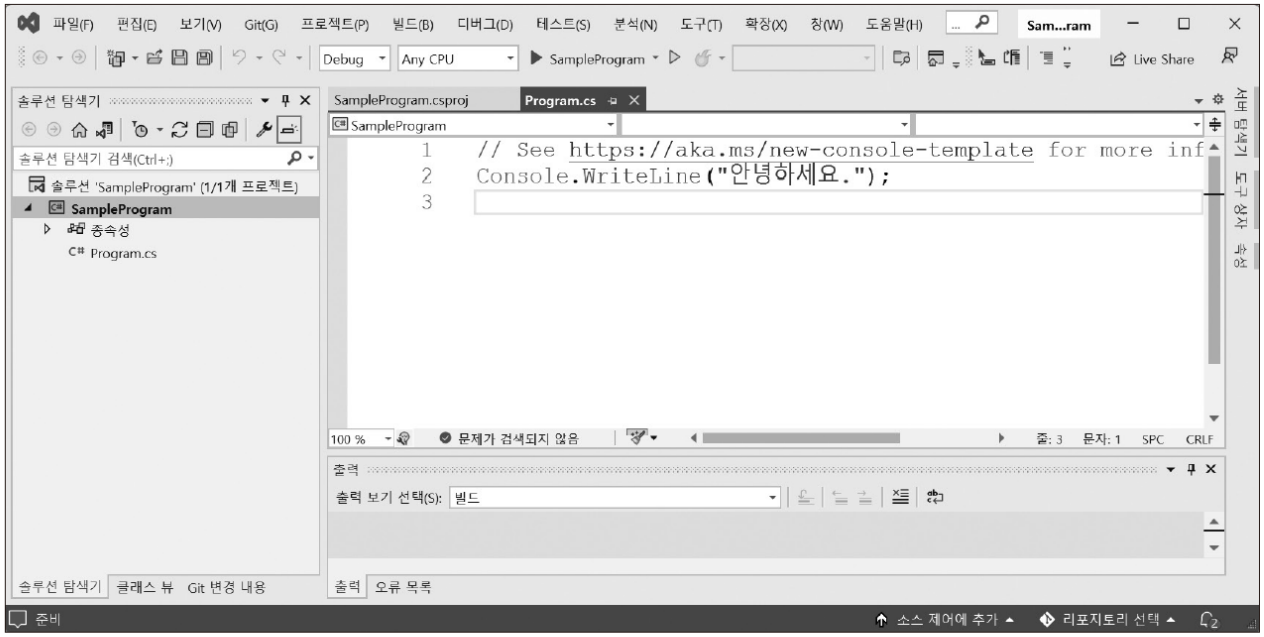

그림 1-39 기존 프로젝트 파일의 소스 코드

2. 문자와 문자열

주석문

- 소스 코드 안에 기록하는 메모와 같은 역할을 수행
- 주석문은 컴파일 과정에서 소스 코드에 전혀 지장을 주지 않음
- 소스 코드의 유지/보수 효율화를 위해 주로 사용

- 한 줄 주석문
  - 겹슬래시(//)를 사용하여 다음과 같이 선언

```
Console.WriteLine("반갑습니다."); // 문자열을 출력 후 강제 개행
```

2. 문자와 문자열

- 여러 줄 주석문
  - /*로 시작하여 */로 끝내도록 다음과 같이 선언

```
/* 프로그램 작성자 : 홍길동
   소속 : 기획실
   업데이트 일자 : 2038. 12. 25  */
Console.Write("반갑습니다.");
```

이스케이프 문자

- 이스케이프 시퀀스를 따르는 문자들로서 다음 문자가 특수 문자임을 의미하는 백슬래시(\ 또는 ₩)를 사용
- 일부 제어 시퀀스인 이스케이프 문자들은 미리 예약어로 등록되어 있음
- C#에서 자주 사용하는 이스케이프 문자는 다음 표와 같음

표 2-3 이스케이프 문자의 종류

이스케이프 문자	의미	참고
\	시퀀스 문자 시작	다음 문자가 특수 문자임을 선언
\n	줄 바꿈	개행문자(캐리지 리턴)
\t	수평 탭	문자열을 출력할 때 일정 간격 띄움
\‘	작은따옴표	문자 또는 문자열 앞/뒤에 선언
\“	큰따옴표	특정 단어를 강조할 때 선언

예제 02-02 이스케이프 문자를 사용하여 문자열 출력하기

• **Step 01 | 프로젝트 생성** : 프로젝트명은 'ESCProgram'으로 입력합니다. 소스 파일명은 ESCprint.cs로 변경합니다.

• **Step 02 | 소스 코드 입력** : 이스케이프 문자를 사용합니다. 주어진 문자열을 특정 형식에 맞게 설정합니다. 그리고 출력하기 위해 다음과 같이 소스 코드를 입력합니다. 단축키 F5를 눌러 실행 결과를 확인합니다.

ch02/ESCProgram/ESCProgram/ESCprint.cs

```
01  // See https://aka.ms/new-console-template for more informatio
02  Console.WriteLine("대한\t민국");
03  Console.WriteLine("용기는 \" 진실\'에서 우러나온다.");
04  Console.WriteLine("오늘보다\n내일은 희망차다.");
```

실행 결과

```
대한    민국
용기는 "진실'에서 우러나온다.
오늘보다
내일은 희망차다.
```

문자열 연결 연산자

- 문자열은 연결 연산자 '+'를 사용하여 2개 또는 3개 이상의 문자열을 연결할 수 있음
- 문자열 연결 연산자의 의미는 다음 표와 같음

표 2-4 문자열 연결 연산자

연산자	의미	참고
+	문자열 연결	문자열1 + 문자열2 + 문자열3

예제 02-03 여러 개의 문자열을 연결하여 출력하기

• **Step 01 | 프로젝트 생성** : 프로젝트명은 'SCProgram'으로 입력합니다. 소스 파일명은 SCprint.cs로 변경합니다.

• **Step 02 | 소스 코드 입력** : 연결 연산자를 사용합니다. 주어진 여러 개의 문자열을 하나의 문자열로 연결합니다. 그리고 출력하기 위해 다음과 같이 소스 코드를 입력합니다. 단축키 F5 를 눌러 실행 결과를 확인합니다.

ch02/SCprogram/SCprogram/SCprint.cs

```
01  // See https://aka.ms/new-console-template for more informatio
02  Console.WriteLine("대" + "한" + "민국");
03  Console.WriteLine("자축인묘" + "진사오" + "미신유술해");
```

실행 결과

대한민국
자축인묘진사오미신유술해

문자 선택 괄호

- 0부터 시작하는 인덱스 번호를 대괄호 안에 선언하여 문자열 중 특정 위치에 존재하는 문자만 선택할 수 있음
- 문자열에서 특정 문자를 선택할 때 사용하는 괄호와 사용 형식은 다음 표와 같음

표 2-5 문자 선택 괄호

연산자	의미	참고
"문자열"[인덱스]	문자열 중 특정 위치의 문자 선택	"대한민국"[2] → 민

예제 02-04 주어진 문자열에서 특정 문자를 선택하여 출력하기

• **Step 01 | 프로젝트 생성** : 프로젝트명은 'SSprogram'으로 입력합니다. 소스 파일명은 SSprint.cs로 변경합니다.

• **Step 02 | 소스 코드 입력** : 주어진 문자열에서 특정 위치에 존재하는 문자를 선택합니다. 그리고 출력하기 위해 다음과 같이 소스 코드를 입력합니다. 단축키 F5를 눌러 실행 결과를 확인합니다.

ch02/SSprogram/SSprogram/SSprint.cs

```
01 // See https://aka.ms/new-console-template for more informatio
02 Console.WriteLine("대한민국"[0]);
03 Console.WriteLine("대한민국"[1]);
04 Console.WriteLine("대한민국"[2]);
05 Console.WriteLine("대한민국"[3]);
```

실행 결과

```
대
한
민
국
```

3. 비교 논리 연산자

■ 부울 연산자

- 참(True) 또는 거짓(False)을 표현할 때 사용
- 부울 연산자로 표현할 수 있는 결과값은 True와 False의 두 가지 형태로만 출력

예제 02-05 부울 연산자의 기본값 출력하기

• **Step 01 | 프로젝트 생성** : 프로젝트명은 'DDprogram'으로 입력합니다. 소스 파일명은 DDprint.cs로 변경합니다.

• **Step 02 | 소스 코드 입력** : 부울 연산자의 기본값을 출력하기 위해 다음과 같이 소스 코드를 입력합니다. 단축키 F5를 눌러 실행 결과를 확인합니다.

ch02/DDprogram/DDprogram/DDprint.cs

```
01  // See https://aka.ms/new-console-template for more informatio
02  Console.Write("참 : ");
03  Console.WriteLine(true);
04  Console.Write("거짓 : ");
05  Console.WriteLine(false);
```

실행 결과

```
참 : True
거짓 : False
```

3. 비교 논리 연산자

비교 연산자

- 2개 이상의 피연산자를 비교하여 True와 False의 결과값을 반환할 때 사용

표 2-6 연산자의 종류

연산자	의미	참고
==	왼쪽과 오른쪽 값이 같음	88 == 99 → False
!=	왼쪽과 오른쪽 값이 같지 않음	88 != 99 → True
>	왼쪽 값이 오른쪽 값보다 큼	88 > 99 → False
<	왼쪽 값이 오른쪽 값보다 작음	88 < 99 → True
>=	왼쪽 값이 오른쪽 값보다 크거나 같음	88 >= 99 → False
<=	왼쪽 값이 오른쪽 값보다 작거나 같음	88 <= 99 → True

예제 02-06 비교 연산자를 선언하여 부울 결과값 출력하기

• **Step 01 | 프로젝트 생성** : 프로젝트명은 'COprogram'으로 입력합니다. 소스 파일명은 COprint.cs로 변경합니다.

• **Step 02 | 소스 코드 입력** : 비교 연산자를 사용합니다. 그리고 결과값을 출력하기 위해 다음과 같이 소스 코드를 입력합니다. 단축키 F5를 눌러 실행 결과를 확인합니다.

ch02/COprogram/COprogram/COprint.cs

```
01  // See https://aka.ms/new-console-template for more informatio
02  Console.Write("88 > 99의 비교 연산 결과 : ");
03  Console.WriteLine(88 > 99);
04  Console.Write("88 <= 99의 비교 연산 결과 : ");
05  Console.WriteLine(88 <= 99);
```

실행 결과

```
88 > 99의 비교 연산 결과 : False
88 <= 99의 비교 연산 결과 : True
```

3. 비교 논리 연산자

논리 연산자

- 2개 이상의 피연산자를 대상으로 NOT, OR, AND 연산을 수행
- 단항 연산자 : NOT 연산자인 '!'
- 이항 연산자 : '||' 연산자(OR)와 '&&' 연산자(AND)

표 2-7 논리 연산자의 종류

연산자	의미	참고
!	논리 부정	!true → False
||	논리합	true || false → True
&&	논리곱	true && false → False

예제 02-07 논리 연산자를 사용하여 결과값 출력하기

• **Step 01 | 프로젝트 생성** : 프로젝트명은 'LOprogram'으로 입력합니다. 소스 파일명은 LOprint.cs로 변경합니다.

• **Step 02 | 소스 코드 입력** : 비교 연산자를 사용합니다. 그리고 결과값을 출력하기 위해 다음과 같이 소스 코드를 입력합니다. 단축키 F5를 눌러 실행 결과를 확인합니다.

ch02/LOprogram/LOprogram/LOprint.cs

```
01  // See https://aka.ms/new-console-template for more informatio
02  Console.Write("!false의 논리 연산 결과 : ");
03  Console.WriteLine(!false);
04  Console.Write("true || false의 논리 연산 결과 : ");
05  Console.WriteLine(true || false);
06  Console.Write("true && false의 논리 연산 결과 : ");
07  Console.WriteLine(true && false);
```

실행 결과

```
!false의 논리 연산 결과 : True
true || false의 논리 연산 결과 : True
true && false의 논리 연산 결과 : False
```

3. 비교 논리 연산자

- A와 B에 대한 논리 연산 결과는 다음 표와 같음

표 2-8 OR 및 AND 논리 연산 결과

A	B	A || B	A && B
true	true	true	true
true	false	true	false
false	true	true	false
false	false	false	false

자신감 뿜뿜!

C# 프로그래밍 Hard Carry

Thank You