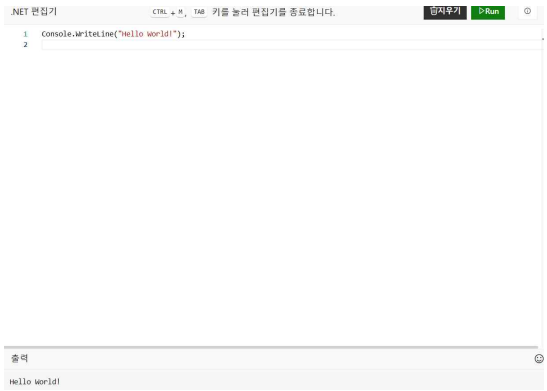
2376261 정서진 C# 공부기록

Ⅰ. 첫 번째 C# 코드 작성

1. C# 소개

: 사용 시 데이터를 캡처, 분석, 처리하는 비즈니스 애플리케이션, 2D 및 3D 게임, 공학용 애플리케이션 등을 빌드 가능하다.

2. 연습 – 첫 번째 코드 작성



** 일반적인 오류: Console과 WriteLine 소문자로 시작

명령의 끝에 세미콜론 누락

** 주석 처리: //

** Console.Write와 Console.WriteLine의 차이: 전자는 줄바꿈을 포함하고 후자는 포함하지 않는다.

3. 작동 방식 알아보기

(1) 소스코드 (code): 프로그래밍 언어로 작성하는 명령. 이 시점에서 개발자가 코드를 업데이트, 변경할 수 있지만 컴퓨터에서 코드를 이해할 수는 없다. => 컴파일!

(2) 컴파일: 컴파일러가 소스코드를 CPU에서 실행할 수 있는 형식으로 변환하는 것.

(3) 구문: C# 코드를 작성하는 규칙. 키워드 및 연산자를 정의하고 이를 결합하여 프로그램을 구성하는 방법을 정의한다.

(4) Console.WriteLine = Console클래스 + WriteLine()메소드 + 괄호 + "리터럴 문자열" + 세미콜론

5. 과제 완료

2. 다음 출력을 생성하는 코드 작성

Output 복사

```
This is the first line.
This is the second line.
```

```
1  Console.WriteLine("This is the first line.");
2  Console.Write("This is the second line.");
```

Ⅱ. C에서 리터럴 및 변수 값을 사용하여 데이터 저장 및 검색#

1. 소개

: C#으로 빌드하는 대부분의 애플리케이션에서는 데이터 작업을 수행한다. 이때 데이터가 애플리케이션에 하드 코드*될 수 있는데, 이 하드 코드된 값을 상수 또는 리터럴 값이라고 한다.

** 하드 코드: 일정하고 변경되지 않는 값

특정 데이터 형식을 저장할 수 있는 변수를 생성하고 값을 설정한 뒤 해당 값을 검색해보자. 또한 코드를 간소화해보자.

2. 연습 – 리터럴 값 출력

(1) 문자 리터럴 (char): 작은 따옴표 ' ' <-> 큰 따옴표 " "는 string 데이터 형식

** 작은 따옴표는 단일 문자일 때만 사용 가능

(2) 정수 리터럴 (int): 연산자 필요 x

(3) 부동 소수점 리터럴 (float, double, decimal): 부동 소수점 숫자는 소수를 포함하는 숫자이다. 세 가지 형식은 다양한 정밀도를 지원한다.

** 리터럴 접미사: 1) float 형식은 F (f)

2) double은 필요 x. 10진수 입력

3) decimal 형식은 m (M) //10진 리터럴

(ex) Console.WriteLine(0.25F);

(4) 부울 리터럴 (bool): 참 혹은 거짓을 나타낸다.

(ex) Console.WriteLine(true);

3. 변수 선언

: 변수란 값 형식을 저장하기 위한 컨테이너. 변수 선언 시 데이터 형식을 선언하고 변수에 이름을 지정한다.

** 변수이름에는 영문자와 숫자, 밑줄을 사용할 수 있으며, 첫 글자는 영문자와 밑줄만 올 수 있다. 또한 변수의 이름은 C# 키워드가 될 수 없다.

4. 연습 – 변수 값 설정 및 가져오기

```
1   string firstName;
2   firstName = "Bob";
3   Console.WriteLine(firstName);
```

5. 암시적 형식 지역 변수 선언 (var)

: 엄격하게 형식을 지정하지 않아도 되는 약한 형식(weak type).

컴파일러가 형식을 자동으로 지정해준다!

** 다만 한번 데이터 타입이 형식화되면 변경이 불가하다. 또한 초기화가 필수다.

(ex) var message = "Hello World!";

message = 10.703m; //에러. message는 string 타입이 됐음.

6. 과제: 리터럴 및 변수 값 표시

```
1   String name = "Bob";
2   int num1 = 3;
3   float temp = 34.4f;
4
5   Console.WriteLine("Hello, " + name + "! You have "+num1+
6   " messages in your inbox. The temperature is " + temp + " Celsius.");
```

Ⅲ. C#으로 기본 문자열 서식 지정

1. 소개

: 애플리케이션이 전달하려는 내용을 사용자가 이해할 수 있도록 서식을 올바르게 지정하자!

문자 이스케이프 시퀀스를 사용하여 특수 문자와 다른 언어의 문자를 포함하도록 텍스트의 리터럴 문자열에 형식을 지정하고, 문자열을 연결하는 방법을 학습하고 문자열 보간을 사용하여 대체 가능한 부분이 있는 리터럴 문자열 템플릿을 만든다.

2. 연습 – 문자 이스케이프 시퀀스를 사용하여 문자열 결합

(1) 문자 이스케이프 시퀀스: 문자열의 출력에 영향을 주는 특수 문자를 삽입하기 위한 런타임에 대한 명령. 문자열 안에 삽입한다.

=> ₩n: 새줄 추가 ₩₩: 문자열로 ₩ 출력

₩t: 탭 추가 ₩": 문자열로 " 출력

(2) 축자 문자열 리터럴 (@): ₩ 이스케이프 할 필요 없이 모든 공백과 문자 유지. 문자열 앞에 @를 붙인다.

(ex) Console.WriteLine(@" c:₩source₩repos

(this is where your code goes)"); => 문자열 그대로 출력됨.

(3) 유니코드 이스케이프 문자: ₩u 이스케이프 시퀀스를 사용하여 리터럴 문자열에 인코딩된 문자를 추가한 다음, 유니코드(UTF-16)의 일부 문자를 나타내는 네 문자 코드를 추가 가능.

```
9   Console.Write("\n\n\u65e5\u672c\u306e\u8acb\u6c42\u66f8\u3092\u751f\u6210\u3059\u308b\u306b\u306f\uff1a\n\t");
10  // User command to run an application
11  Console.WriteLine(@"c:\invoices\app.exe -j");
```

```
日本の請求書を生成するには：
    c:\invoices\app.exe -j
```

3. 연습 – 문자열 연결을 사용하여 문자열 결합

: 두 개 이상의 string 값을 새 string 값으로 단순 결합 (+)

4. 연습 – 문자열 보간을 사용하여 문자열 결합

: 템플릿과 하나 이상의 보간 식을 사용하여 여러 값을 단일 리터럴 문자열로 결합한다. 보간식은 { }안에 변수를 넣어 표시하고 문자열 앞에 $를 넣어 템플릿을 만든다.

```
string message = greeting + " " + firstName + "!";
string message = $"{greeting} {firstName}!";
```

-> 2개는 같은 코드.

** 축자 리터럴 및 문자열 보간 결합: $@ 사용

(ex) string projectName = "First-Project";

Console.WriteLine($@"C:₩Output₩{projectName}₩Data");

6. 과제: 명령 형식 지정 및 표시

```
string projectName = "ACME";
string englishLocation = $@"c:\Exercise\{projectName}\data.txt";
Console.WriteLine($"View English output:\n\t\t{englishLocation}\n");

string russianMessage = "\u041f\u043e\u0441\u043c\u043e\u0442\u0440\u0435\u0442\u044c \u0440\u0443\u0441\u0441\u043
string russianLocation = $@"c:\Exercise\{projectName}\ru-RU\data.txt";
Console.WriteLine($"{russianMessage}:\n\t\t{russianLocation}\n");
```

```
View English output:
        c:\Exercise\ACME\data.txt

Посмотреть русский вывод:
        c:\Exercise\ACME\ru-RU\data.txt
```

Ⅳ. C의 숫자에 대한 기본 작업 수행#

1. 소개

: 리터럴 및 숫자 데이터에 대한 기본 수학 연산을 수행하는 데 사용되는 연산자와 기술을 알아보자.

2. 연습 – 암시적 데이터 변환을 사용하여 더하기 수행

(1) 두 개의 숫자 값 더하기 (+)

(ex) int firstNumber = 12;

int secondNumber = 7;

Console.WriteLine(firstNumber + secondNumber);

(2) 여러 데이터 형식을 함께 사용하여 암시적 형식 변환을 강제로 사용

(ex) string firstName = "Bob";

int widgetsSold = 7;

Console.WriteLine(firstName + " sold " + widgetsSold + " widgets.");

=> Bob sold 7 widgets.

(3) 컴파일러의 의도를 명확히 설명하기 위해 괄호 활용

(ex) string firstName = "Bob";

int widgetsSold = 7;

Console.WriteLine(firstName + " sold " + (widgetsSold + 7) + " widgets.");

3. 연습 – 수학 연산 수행

** 기본 연산자: +, -, *, /

그러나 나누기 결과의 몫이 예상과 다를 수 있다. int로 정의되어 있을 시 소수점 뒤의 값을 포함할 수 없다. => 소수 자릿수를 지원하는 decimal 등의 데이터 형식을 사용!

(ex) decimal decimalQuotient = 7.0m / 5;

Console.WriteLine($"Decimal quotient: {decimalQuotient}");

=>출력 Decimal quotient: 1.4

**이때 몫이 decimal 형식이어야 하며 나뉘는 숫자 중 하나 이상이 decimal 형식이어야 함.

+ 정수 나누기 결과를 캐스팅하는 코드 추가

: 두 변수를 int 형식으로 유지하면서 나누기의 결과 또한 정상 출력되길 바랄 때, ( )를 사용하여 캐스트 연산자를 추가한다.

(ex) int first = 7;

int second = 5;

decimal quotient = (decimal)first / (decimal)second;

Console.WriteLine(quotient);

** 정수 나누기 이후 나머지를 확인하는 코드 작성: % 활용

```
1   Console.WriteLine($"Modulus of 200 / 5 : {200 % 5}");
2   Console.WriteLine($"Modulus of 7 / 5 : {7 % 5}");
```

```
Modulus of 200 / 5 : 0
Modulus of 7 / 5 : 2
```

*** 작업순서: 괄호 > 지수 > 곱하기/나누기 > 더하기/빼기

4. 증가 및 감소

: 데이터 구조와 상호작용하는 반복 논리 또는 코드를 작성할 때 값을 증분/감소시켜야 하는 경우가 많다.

** 복합 대입 연산자: +=, -=, *=, ++, --

** ++와 —의 경우 위치에 따라 연산 수행의 시기가 다르다. 값 앞에 사용할 시 값이 검색되기 전에 값이 증가되고 뒤에 사용할 시 검색된 후에 값을 늘린다.

5. 과제: 화씨에서 섭씨로 변환

```
1   int fahrenheit = 94;
2   decimal result = (fahrenheit-32)*5/9m;
3   Console.WriteLine("The temperature is " + result + "Celsius.");
```

```
The temperature is 34.444444444444444444444444444Celsius.
```

V. 단계별 프로젝트 – 학생 성적 계산 및 프린트

1. 소개

: 학생 성적 애플리케이션을 개발하자. 학생 이름을 기반으로 변수에 값을 선언 및 할당하고 다양한 숫자 계산을 수행하고 결과를 표시한다.

2. 가이드 프로젝트 준비

```
// initialize variables - graded assignments
int currentAssignments = 5;

int sophia1 = 93;
int sophia2 = 87;
int sophia3 = 98;
int sophia4 = 95;
int sophia5 = 100;

int nicolas1 = 80;
int nicolas2 = 83;
int nicolas3 = 82;
int nicolas4 = 88;
int nicolas5 = 85;

int zahirah1 = 84;
int zahirah2 = 96;
int zahirah3 = 73;
int zahirah4 = 85;
int zahirah5 = 79;

int jeong1 = 90;
int jeong2 = 92;
int jeong3 = 98;
int jeong4 = 100;
int jeong5 = 97;
```

3. 연습 – 각 학생의 과제 점수 합계 계산

```
33  Console.WriteLine("Sophia: " + sophiaSum);
34  Console.WriteLine("Nicolas: " + nicolasSum);
35  Console.WriteLine("Zahirah: " + zahirahSum);
36  Console.WriteLine("Jeong: " + jeongSum);
```

출력

```
Sophia: 0
Nicolas: 0
Zahirah: 0
Jeong: 0
```

```
11  int nicolas2 = 83;
12  int nicolas3 = 82;
13  int nicolas4 = 88;
14  int nicolas5 = 85;
15
16  int zahirah1 = 84;
17  int zahirah2 = 96;
18  int zahirah3 = 73;
19  int zahirah4 = 85;
20  int zahirah5 = 79;
21
22  int jeong1 = 90;
23  int jeong2 = 92;
24  int jeong3 = 98;
25  int jeong4 = 100;
26  int jeong5 = 97;
27
28  int sophiaSum = sophia1 + sophia2 + sophia3 + sophia4 + sophia5;
29  int nicolasSum = nicolas1 + nicolas2 + nicolas3 + nicolas4 + nicolas5;
30  int zahirahSum = zahirah1 + zahirah2 + zahirah3 + zahirah4 + zahirah5;
31  int jeongSum = jeong1 + jeong2 + jeong3 + jeong4 + jeong5;
32
33  Console.WriteLine("Sophia: " + sophiaSum);
34  Console.WriteLine("Nicolas: " + nicolasSum);
35  Console.WriteLine("Zahirah: " + zahirahSum);
36  Console.WriteLine("Jeong: " + jeongSum);
```

출력

```
Sophia: 473
Nicolas: 418
Zahirah: 417
Jeong: 477
```

4. 연습 – 각 학생의 할당 점수 평균 계산

: 평균을 저장할 땐 10진수 데이터 형식을 사용하자.

```
33  /*Console.WriteLine("Sophia: " + sophiaSum);
34  Console.WriteLine("Nicolas: " + nicolasSum);
35  Console.WriteLine("Zahirah: " + zahirahSum);
36  Console.WriteLine("Jeong: " + jeongSum);*/
37
38  decimal sophiaScore = sophiaSum / currentAssignments;
39  decimal nicolasScore = nicolasSum / currentAssignments;
40  decimal zahirahScore = zahirahSum / currentAssignments;
41  decimal jeongScore = jeongSum / currentAssignments;
42
43  Console.WriteLine("Sophia: " + sophiaScore);
44  Console.WriteLine("Nicolas: " + nicolasScore);
45  Console.WriteLine("Zahirah: " + zahirahScore);
46  Console.WriteLine("Jeong: " + jeongScore);
```

출력

```
Sophia: 94
Nicolas: 83
Zahirah: 83
Jeong: 95
```

```
37
38  decimal sophiaScore = (decimal) sophiaSum / currentAssignments;
39  decimal nicolasScore = (decimal) nicolasSum / currentAssignments;
40  decimal zahirahScore = (decimal) zahirahSum / currentAssignments;
41  decimal jeongScore = (decimal) jeongSum / currentAssignments;
42
43  Console.WriteLine("Sophia: " + sophiaScore);
44  Console.WriteLine("Nicolas: " + nicolasScore);
45  Console.WriteLine("Zahirah: " + zahirahScore);
46  Console.WriteLine("Jeong: " + jeongScore);
```

출력

```
Sophia: 94.6
Nicolas: 83.6
Zahirah: 83.4
Jeong: 95.4
```

=> 나누기 계산 결과를 10진수 값으로 만들려면 피제수 또는 제수가 10진수 형식이어야 함.

5. 연습 – 이스케이프 문자 시퀀스를 사용하여 출력 형식 지정

```
43  Console.WriteLine("Student\t\tGrade\n");
44  Console.WriteLine("Sophia:\t\t" + sophiaScore + "\tA");
45  Console.WriteLine("Nicolas:\t" + nicolasScore + "\tB");
46  Console.WriteLine("Zahirah:\t" + zahirahScore + "\tB");
47  Console.WriteLine("Jeong:\t\t" + jeongScore + "\tA");
```

출력

```
Student     Grade

Sophia:     94.6    A
Nicolas:    83.6    B
Zahirah:    83.4    B
Jeong:      95.4    A
```

Ⅵ. 안내 프로젝트 – 최종 GPA 계산

1. 소개

: 학생의 성적과 학점을 사용하여 전체 GPA를 계산하는 애플리케이션을 개발하자.

2. 가이드 프로젝트 준비

```
string studentName = "Sophia Johnson";
string course1Name = "English 101";
string course2Name = "Algebra 101";
string course3Name = "Biology 101";
string course4Name = "Computer Science I";
string course5Name = "Psychology 101";

int course1Credit = 3;
int course2Credit = 3;
int course3Credit = 4;
int course4Credit = 4;
int course5Credit = 3;
```

3. 연습 – 각 과정의 숫자 성적 값 저장

```
string studentName = "Sophia Johnson";
string course1Name = "English 101";
string course2Name = "Algebra 101";
string course3Name = "Biology 101";
string course4Name = "Computer Science I";
string course5Name = "Psychology 101";

int course1Credit = 3;
int course2Credit = 3;
int course3Credit = 4;
int course4Credit = 4;
int course5Credit = 3;

int gradeA = 4;
int gradeB = 3;

int course1Grade = gradeA;
int course2Grade = gradeB;
int course3Grade = gradeB;
int course4Grade = gradeB;
int course5Grade = gradeA;

Console.WriteLine($"{course1Name} {course1Grade} {course1Credit}");
Console.WriteLine($"{course2Name} {course2Grade} {course2Credit}");
Console.WriteLine($"{course3Name} {course3Grade} {course3Credit}");
Console.WriteLine($"{course4Name} {course4Grade} {course4Credit}");
Console.WriteLine($"{course5Name} {course5Grade} {course5Credit}");
```

출력

```
English 101 4 3
Algebra 101 3 3
Biology 101 3 4
Computer Science I 3 4
Psychology 101 4 3
```

4. 연습 – 크레딧 시간 및 성적 포인트의 합계 계산

```
 8   int course1Credit = 3;
 9   int course2Credit = 3;
10   int course3Credit = 4;
11   int course4Credit = 4;
12   int course5Credit = 3;
13
14   int gradeA = 4;
15   int gradeB = 3;
16
17   int course1Grade = gradeA;
18   int course2Grade = gradeB;
19   int course3Grade = gradeB;
20   int course4Grade = gradeB;
21   int course5Grade = gradeA;
22
23   int totalCreditHours = 0;
24
25   totalCreditHours += course1Credit;
26   totalCreditHours += course2Credit;
27   totalCreditHours += course3Credit;
28   totalCreditHours += course4Credit;
29   totalCreditHours += course5Credit;
30
31   int totalGradePoints = 0;
32
33   totalGradePoints += course1Credit * course1Grade;
```

출력

```
57 17
English 101 4 3
Algebra 101 3 3
Biology 101 3 4
Computer Science I 3 4
Psychology 101 4 3
```

5. 연습 – 10진수 출력 형식 지정

```
23  int totalCreditHours = 0;
24  totalCreditHours += course1Credit;
25  totalCreditHours += course2Credit;
26  totalCreditHours += course3Credit;
27  totalCreditHours += course4Credit;
28  totalCreditHours += course5Credit;
29
30  int totalGradePoints = 0;
31  totalGradePoints += course1Credit * course1Grade;
32  totalGradePoints += course2Credit * course2Grade;
33  totalGradePoints += course3Credit * course3Grade;
34  totalGradePoints += course4Credit * course4Grade;
35  totalGradePoints += course5Credit * course5Grade;
36
37  decimal gradePointAverage = (decimal) totalGradePoints/totalCreditHours;
38
39  int leadingDigit = (int) gradePointAverage;
40  int firstDigit = (int) (gradePointAverage * 10 ) % 10;
41  int secondDigit = (int) (gradePointAverage * 100 ) % 10;
42
43  Console.WriteLine($"{course1Name} {course1Grade} {course1Credit}");
44  Console.WriteLine($"{course2Name} {course2Grade} {course2Credit}");
45  Console.WriteLine($"{course3Name} {course3Grade} {course3Credit}");
46  Console.WriteLine($"{course4Name} {course4Grade} {course4Credit}");
47  Console.WriteLine($"{course5Name} {course5Grade} {course5Credit}");
48
49  Console.WriteLine($"Final GPA: {leadingDigit}.{firstDigit}{secondDigit}");
```

출력

```
English 101 4 3
Algebra 101 3 3
Biology 101 3 4
Computer Science I 3 4
Psychology 101 4 3
Final GPA: 3.35
```

6. 연습 – 이스케이프 문자 시퀀스 사용하여 출력 형식 지정

```
43  Console.WriteLine($"Student: {studentName}\n");
44  Console.WriteLine("Course\t\t\tGrade\tCredit Hours");
45
46  Console.WriteLine($"{course1Name}\t\t\t{course1Grade}\t\t{course1Credit}");
47  Console.WriteLine($"{course2Name}\t\t\t{course2Grade}\t\t{course2Credit}");
48  Console.WriteLine($"{course3Name}\t\t\t{course3Grade}\t\t{course3Credit}");
49  Console.WriteLine($"{course4Name}\t{course4Grade}\t\t{course4Credit}");
50  Console.WriteLine($"{course5Name}\t\t{course5Grade}\t\t{course5Credit}");
51
52  Console.WriteLine($"\nFinal GPA:\t\t\t {leadingDigit}.{firstDigit}{secondDigit}");
```

출력

```
Student: Sophia Johnson

Course              Grade   Credit Hours
English 101         4       3
Algebra 101         3       3
Biology 101         3       4
Computer Science I  3       4
Psychology 101      4       3

Final GPA:           3.35
```