**09.23 C# 정리본**

C# = .net(닷넷) 기반 프로그램 개발에 사용하는 프로그래밍 언어

CLR = 공용언어 런타임(닷넷의 핵심요소?)

코드를 관리하는 주 역할을 담당한다. / 프로그램 오류가 발생하면 처리한다.

관리되는 코드들의 형식은 안전성이 보장된다… 메모리 관리 , 스레드 실행, 코드 실행 등등 관리한다.

**CLS / CTS**

CTS : 닷넷에서 사용되는 언어들이 어떤 타입을 가져야 하는가에 대한 정의

Ex) int, unsigned int 타입을 가지고 있을 때 이 타입이 정의되어 있지 않으면 사용 불가..

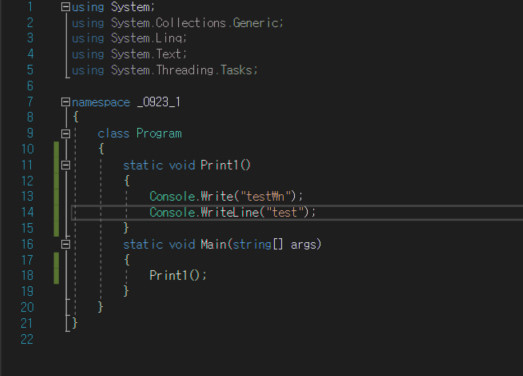
CLS : 교차언어개발(서로 다른 언어들을 사용할 수 있다?)

**C# 기본형**

Class 내부에 static void 있는 게 기본형이다.

관리형 힙 >> 힙에 new로 만들면 삭제하지 않아도 자동으로 관리되기 때문에, 알아서 삭제해 준다.

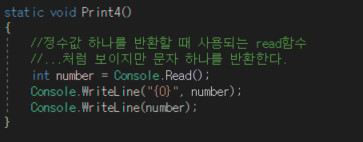
실제 참조 변수는 하나지만 다른 두 개 이상의 주소가 참조할 수 있다고 한다.



<그림 1. 간단한 출력코드>

위 그림1은 C#내에서 간단한 출력코드를 만들어 내는 모양이다. 참고하도록 하자.

**<<Read와 Write 함수>>**



<그림2. Read와 write함수>

Read 함수는 기본적으로 정수값 하나를 반환할 때 사용되는데, 사실 입력하게 되면 문자 하나를 반환한다.

Ex) a를 치고 엔터를 입력한 경우 97 이라는 아스키 코드를 반환한다.

WriteLine 함수는 사용하게 되면 자동으로 뒤에 \n (NULL)을 가지는데, 위 방식처럼 그냥 number를 써도 되고 {0} 과 같은 (c에서는 %d 등으로 쓰이는 것 같다.) 방식으로 사용하면 된다.

**<<C# 구성요소>>**

교재 p.6에 보면 여러가지 데이터 관련 형식(타입)들이 있으니 한번 보도록 하자.

1. 값 형식 vs 참조 형식

Object / String (참조 형식)

오브젝트는 하나의 클래스이다. 모든 타입형들은 오브젝트의 자식? 이라고 한다.

닷넷에서 오브젝트 타입을 부르는 타입명이다.

Sytem.object >>모든 타입의 기반이 되는 타입이다. (대충 부모라는 뜻)

어떤 타입을 정의하면 암묵적으로 오브젝트 타입으로 정해진다.

클래스를 만들었을 때, 오브젝트를 부모로 둘 수는 없다. 이미 정의된 것을 두 번 정의하는 것과 같은 이치.

이것의 멤버는 기본 생성자를 제공하는데, 입력 매개 변수가 없는 생성자이다.

(Public object()형식을 가짐.)

참조 형식은 관리화 힙에 할당된 객체를 사용한다. 4라는 정수형 객체를 생성하면 값 형식에는 4 자체가 들어가지만 참조 형식은 1000이라는 임의의 주소에 저장된 4 라는 데이터를 가진다는 뜻이다.

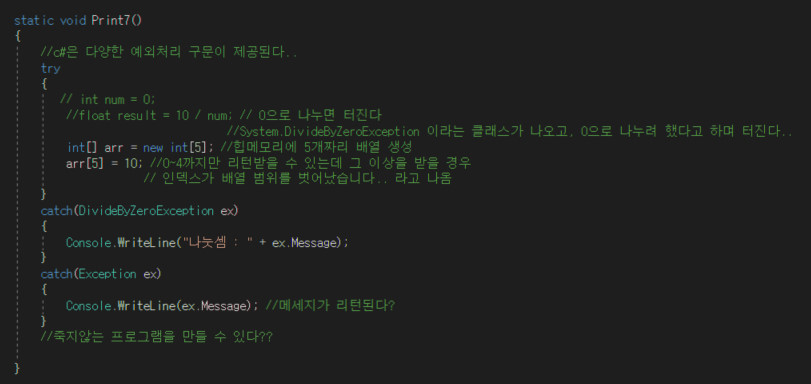
104 ref 1000 힙>> 1000 4

100 val 4 힙>> 주소 값

대충 위와 비슷한 느낌

한마디로 스택 메모리에는 &1000이라는 값, 힙 메모리에 4라는 값이 저장된다고 보면 된다.

**<<예외 처리문>>**



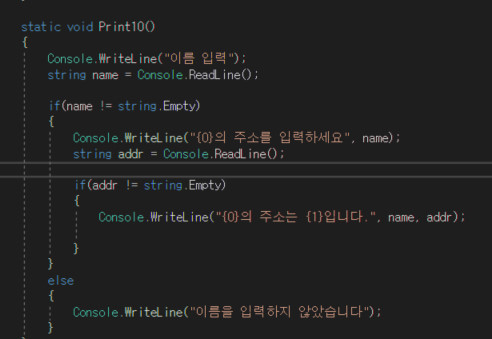
<그림 3. 예외처리 코드>

그림 3에서 보이듯이 try ~ catch 문 형식으로 사용할 수 있는 예외처리 코드이다.

Int num 은 0으로 초기화되고, 어떠한 값은 0으로 나눌 수 없기 때문에 0으로 나눌 수 없습니다.. 라는 결과를 가지며 터진다.

예외로 설정한 catch 문에 exception 의 메시지를 리턴하는 ex.Message의 결과를 참고하자.

**<<if / else 문>>**



<그림 4. If/ else 문>

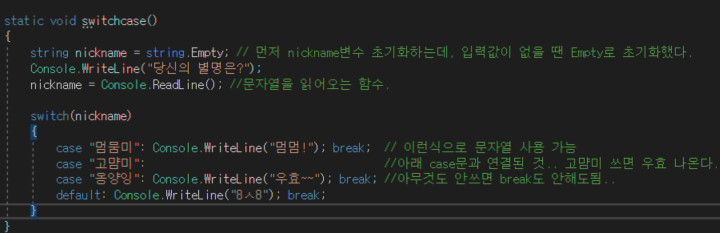
조건문은 그냥 똑같이 쓰면 된다. WriteLine 에서 헷갈리긴 한데 대충 익숙해지면 쓸만할 듯.

String.Empty 는 아무 입력도 없는 경우를 말한다.

그림 4에서 표현한 방식은 이름을 입력하지 않으면 else문이 뜨도록 만든 것.

**<<switch 문>>**

C#의 스위치문은 문자열의 case도 받을 수 있다!



<그림 5. Switch문>

위 그림 5처럼 스위치문 사용하면 되는데, readline으로 문자열 읽고

Case 뒤에 문자열도 사용할 수 있다. 그 외에는 일반 C++과 똑같음.

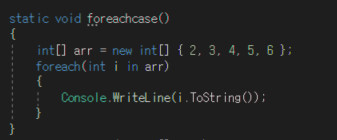
문자열 뒤에 아무것도 오지 않으면 break 안 해도 된다. 있으면 무조건 해야됨.

안한 case 문은 바로 아래에 break한 문과 똑같은 결과를 가진다.

Default는 왜 한지 몰겠다 솔직히.

<<foreach문>>

기존 for문처럼 막 for(int i = 0; i<5; i++) 이런식으로 표현 안하고 내부 배열을 표시할 때



<그림 6. Foreach문>

이렇게 foreach 를 사용해도 된다. 결과는 한줄마다 2 3 4 5 6 이 나온다.

같은 의미긴 한데, 내부에 for문을 사용하면 WriteLine내부 괄호에 뭔가 다른걸 써넣어야 하는데..?

그건 나중에 알아보자.

**<<참조 형식과 값 형식의 차이점 >>**

다시 오브젝트를 보기 전, 참조 형식과 값 형식의 차이에 대해서 되새김질 해보자.

값 형식은 자신의 저장공간이 존재하는 것이다.(구조체, 열거형 등등)

참조 형식은 new로 새로 힙메모리에 저장공간을 만들어서 그곳에서 참조하는 방식이다. (클래스 등등)

관리화 힙에 할당된 객체를 참조하여 사용되는데, 이 때 한 객체가 여러 변수를 가져도 상관없다!

알아서 관리해 주니까 안정성도 올라간다고 생각하면 편할 거 같다.

<object 형식의 멤버 메서드들>

형식에는 여러 가지가 있다..

Public virtual bool Equals(object obj) <<예스노만 받을거라서 bool형식

Public static bool Equals(object obj1, object obj2) <<