Console.Write에서 형식을 사용하여 출력하는 방식

참조 - https://riucc.tistory.com/563

C#에서 string을 +로 표현하면 비효율/최적화가 안되는 이유

C# Class의 불변성 즉 불변성 클래스 때문

즉 불변성 클래스방식을 사용하는 C#에서는 string을 C와 같은 방식으로 주소에 접속하여 변환시키는 방식이 아닌, 주소를 새로 만들어 새로운 데이터를 할당하는 방식으로 작동하기에 변경 전 데이터가 더미데이터로서 남아 최적화에 악영향을 끼칠 수 있기 때문이다.

참조 - <https://guslabview.tistory.com/187>

C#에서는 GC(garbage collector)를 지원하기에 할당된 메모리를 자동으로 지우지만 이 또한 반복적으로 일어나면 프로그램에 부화를 줄 수도 있기에 처음부터 이러한 코딩을 피하는 것이 좋다.

참조 - <https://velog.io/@cedongne/C-%EA%B0%80%EB%B9%84%EC%A7%80-%EC%BB%AC%EB%A0%89%ED%84%B0>

추가

C#에서는 int, float등은 structure형식으로 저장되어 먼저 지정된 메모리 공간을 할당 받고 stack형식으로 쌓일 수 있다. 하지만 string형식의경우 class형식으로 저장되며 길이를 보장할 수 없기에 stack형식으로 보관할 수 없다. 이러한 이유로 heap형식으로 저장이 되는 방식을 사용한다.]

여기서 문제는 만약 string을 A+B+C로 저장할 경우 처음부터 ABC로 저장되는 것이 아닌 A만큼의 메모리를 할당하고 heap로 연결하고 이후 다시 AB형식을 저장하기 위해 다시 AB만큼의 메모리 할당, 다시 ABC크기로 메모리 할당 …… 이라는 방식으로 작동하기에 메모리를 더 많이 쌓아가는 문제가 발생한다. 이렇게 된다면 메모리에 할당되는 데이터 양이 많아질 뿐만 아니라, 이후 GC가 작동하고, Generation이 쌓여갈수록 프로그램에 부하가 더 쌓여 나간다. 단순한 작업일 경우 문제가 없을 수 있지만 게임과 같은 프레임단위가 중요한 곳에서는 이는 치명적인 문제가 될 수 있어 사용을 기피하는 것이다.

+ C++에서는 GC가 애초에 없는 대신 delete등으로 수동적으로 지울 수 있기에 문제없다고 한다.

Console.Write에서도 그렇다. 여기서는 입력된 string을 s.out로 string을 출력하는데 이는 위와같은 방식으로 s.out로 저장되어 출력되어버리기에 같은 문제가 발생한다고 볼수있다.

추가 참조

메모리 구조 - <https://helloworld-japan.tistory.com/33>

GC와 RefCount - https://reoul.github.io/cpp/cpp-45/