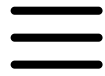


# C++ Study

## C++의 기본 구조

울산대학교 | 박예찬 | Github:devpyc | yechan6855@naver.com



```
#include <iostream>
```

```
int main(){
```

```
    std::cout<<"Hello, C++!"<<std::endl;
```

```
}
```



# 각 코드의 의미?

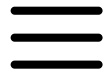


## 입출력을 가능하게 하는 헤더

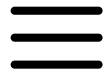
iostream 헤더 파일은 C++ 표준 입출력에 필요한 객체들을 담고있음

`std::cin`, `std::cout`, `std::endl` 등

C언어에서의 `stdio.h` 와 비슷하다고 보면된다.



# std는 왜 붙어있고, 왜 사용할까?



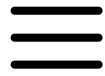
# namespace

std는 C++ 표준 라이브러리의 모든 함수, 객체 등이 정의된 이름 공간

코드의 크기가 늘어남에 따라 중복된 이름을 가진 함수들이 많아짐  
이를 구분하기 위해 소속된 이름 공간이 다르면 다른것으로 취급함



**매번 일일이 `std::`를 붙여야할까?**



# std::cout 대신 cout 쓰기

using namespace std; 를 사용하면 된다.

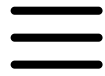
std::cout → cout 으로 간단하게 사용 가능하다.

코드가 짧아지는 장점이 있지만,  
다른 라이브러리와 충돌할 수 있음





# 모든 프로그램의 출발점

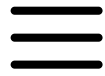


# main()

운영체제는 프로그램을 실행할때, 항상 main() 부터 시작함



# 기본 입출력



## cin(입력), cout(출력)

cin으로 입력을 받을때는 기본적으로 공백(띄어쓰기, 줄바꿈)으로 데이터를 구분함

쉽게 생각하면,

cin은 원하는곳에 데이터를 집어넣기

cout은 구멍 뚫어서 데이터 빼오기

```
#include <iostream>
using namespace std;
```

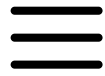
```
int main(){
    int a,b;
    cin>>a>>b;
    cout<<"합계: "<<a+b<<endl;
}
```



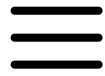
# endl과 '\n'의 차이?



endl에는 줄바꿈과 동시에 버퍼를 비우는 기능이 있고,  
‘\n’은 줄바꿈만 하기 때문에 endl에 비해 빠르다.



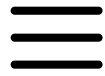
퇴근할게요~



# return 0;

return 0; 정상 종료, 0이 아닌 값이라면 비정상 종료를 의미  
main() 함수에서는 return 0; 를 생략 하더라도 컴파일러가 자동으로 추가해줌





# return 0;

울산대학교 | 박예찬 | Github:devpyc | yechan6855@naver.com