

## Week 5-1 : Name Spaces, Operator Overloading

### Part1. Name Spaces

```
#include <iostream>
void Func(void)
{
    std::cout<<"Microsoft"<<std::endl;
}
void Func(void)
{
    std::cout<<"Apple"<<std::endl;
}
int main(void)
{
    Func(); //declare same form function, error!!
    return 0;
}
```

같은 형태, 다르게 구현된 함수를 메인 함수에서 실행하려고 할 경우, 에러가 발생한다. 이름 공간이 이 문제를 해결해 줄 수 있다.

```
#include <iostream>

namespace Microsoft
{
    void Func(void)
    {
        std::cout<<"Microsoft"<<std::endl;
    }
}

namespace Apple
{
```

```
void Func(void)
{
    std::cout<<"Apple"<<std::endl;
}

int main()
{
    Microsoft::Func();
    Apple::Func();

    return 0;
}
```

- 동일한 Namespace의 함수를 호출할 경우

```
#include <iostream>

namespace Microsoft
{
    void Func(void);
}

namespace Microsoft
{
    void print(void);
}

namespace Apple
{
    void Func(void);
}

int main()
{
    Microsoft::Func();
    return 0;
}

void Microsoft::Func(void)
{
    std::cout<<"Microsoft"<<std::endl;
    print(); //동일 name space
    Apple::Func(); //다른 name space
}
```

```
}  
void Microsoft::print(void)  
{  
    std::cout<<"Microsoft second"<<std::endl;  
}  
void Apple::Func(void)  
{  
    std::cout<<"Apple"<<std::endl;  
}
```

### - namespace의 중첩

```
#include <iostream>  
namespace AAA  
{  
    int num=5;  
  
    namespace BBB  
    {  
        int num=6;  
    }  
    namespace CCC  
    {  
        int num=7;  
    }  
}  
  
int main()  
{  
    std::cout<<"AAA_num:"<<AAA::num<<"    BBB_num:"<<AAA::BBB::num<<"  
    CCC_num:"<<AAA::CCC::num<<std::endl;  
    return 0;  
}
```

### - 입출력과 연관된 namespace std::

std는 iostream으로 선언된 namespace이며, **using namespace std;**를 추가하면 std::cin, std::endl 등을 cin, endl과 같이 간결한 형태로 이용할 수 있다.

- using을 이용한 이름공간의 명시

```
#include <iostream>
namespace AAA
{
    int num=5;

    namespace BBB
    {
        int num=6;
    }
    void func(void)
    {
        std::cout<<"AAA namespace의 func함수"<<std::endl;
    }
}
using namespace AAA;
namespace DDD
{
    void print(void)
    {
        func(); //AAA를 전역으로 사용했기 때문에
        std::cout<<"DDD namespace의 print함수"<<std::endl;
    }
}
int main()
{
    using DDD::print;
    print(); //main내에서 지역적으로 DDD의 print함수 사용
    return 0;
}
```

- 이름공간의 별칭 지정

```
#include <iostream>

namespace AAA
{
    namespace BBB
```

```
    {
        namespace CCC
        {
            int num1=1;
            int num2=2;
        }
    }

namespace ABC=AAA::BBB::CCC;

int main(void)
{
    std::cout<<"num1:"<<AAA::BBB::CCC::num1<<"num2:"<<AAA::BBB::CCC::num2<<std::
endl;
    std::cout<<"num1:"<<ABC::num1<<"num2:"<<ABC::num2<<std::endl;
    //AAA::BBB::CCC:: 대신에 ABC::를 이용해도 동일
}
```

[Exercise]

1. Run the program below. what can you find in this program?

```
int price = 1000;

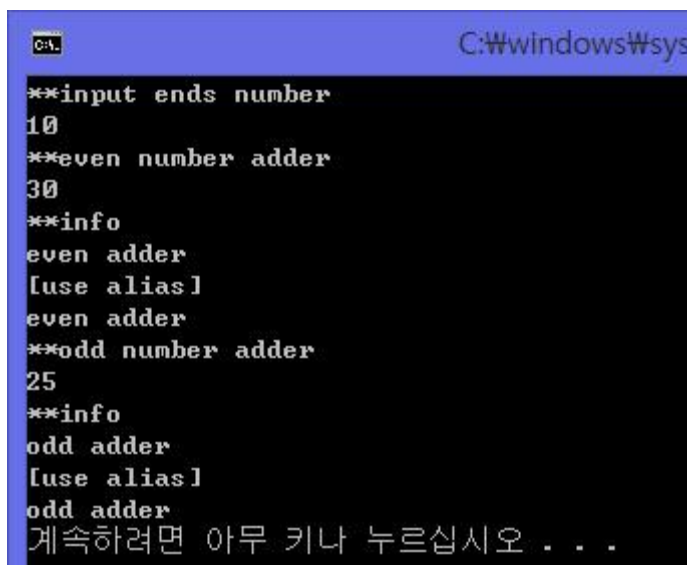
void discount(void)
{
    int price = 2000;
    cout<<price * 0.7<<endl;
    cout<<::price*0.7<<endl;
}

int main()
{
    discount();
    return 0;
}
```

2. Using namespace, make functions which add each even number, odd number. The header file include function templates.

\* namespace

- even
  - └ info
- odd
  - └ info



```
C:\windows\system32
**input ends number
10
**even number adder
30
**info
even adder
[use alias]
even adder
**odd number adder
25
**info
odd adder
[use alias]
odd adder
계속하려면 아무 키나 누르십시오 . . .
```