

02

C++ 프로그래밍의 기본

학습 목표

1. C++ 프로그램의 기본 요소를 이해한다.
2. namespace와 std:: 등 이름 공간을 사용할 수 있다.
3. 키 입력 받는 방법을 안다.
4. 문자열을 입력 받는 방법을 안다.
5. 간단한 C++ 프로그램을 작성할 수 있다.
6. #include <iostream>의 실체에 대해 이해한다.

예제 2-1: 기본적인 C++ 프로그램

3

```
/*
소스: SimpleC++.cpp
cout과 << 연산자를 이용하여 화면에 출력한다.
*/

#include <iostream>                // cout과 << 연산자 선언을 포함

// C++ 프로그램은 main() 함수에서부터 실행을 시작한다.
int main() {
    std::cout << "Hello\n";        //화면에 Hello를 출력 후 다음줄로 넘어감
    std::cout << " 첫 번째 맛보기입니다. ";
    return 0;                      // main() 함수가 종료->프로그램이 종료됨
}
```

Hello
첫 번째 맛보기입니다.

주석문(comment)

4

- 개발자가 작성한 프로그램에 대한 설명 -> 코드 이해를 도와줌
- 컴파일러에 의해 무시됨 -> 기계어로 번역되지 않음 -> 프로그램의 실행결과에 영향을 주지 않음
- 여러 줄 주석문 : `/* ... */`
- 한 줄 주석문 : `//...` 를 만나면 이 줄의 끝까지 주석으로 처리

#include <iostream>

5

- 전처리기(C++ Preprocessor)에게 내리는 지시
 - ▣ <iostream> 헤더 파일을 컴파일 전에 소스에 포함하도록 지시
- <iostream> 헤더 파일
 - ▣ 표준 입출력을 위한 클래스와 객체 등이 선언되어 있음
 - ios, istream, ostream, iostream 클래스 선언
 - cout, cin, <<, >> 등 객체 및 연산자 선언
 - ▣ C++헤더파일은 h 확장자 없음

main() 함수

6

- C++ 프로그램의 실행을 시작하는 함수
 - ▣ C++ 프로그램에 반드시 하나의 main() 함수가 존재해야 함
 - ▣ main() 함수가 종료하면 C++ 프로그램 종료
- main() 함수에서만 예외적으로 return문 생략 가능

```
int main() {           // main()의 리턴 타입 int
    .....
    return 0;          // 0이 아닌 다른 값으로 리턴가능
}
```

```
int main() {
    .....
    // 개발자의 편리를 위해 return 문 생략 가능
}
```

화면 출력

7

□ cout << 출력데이터;

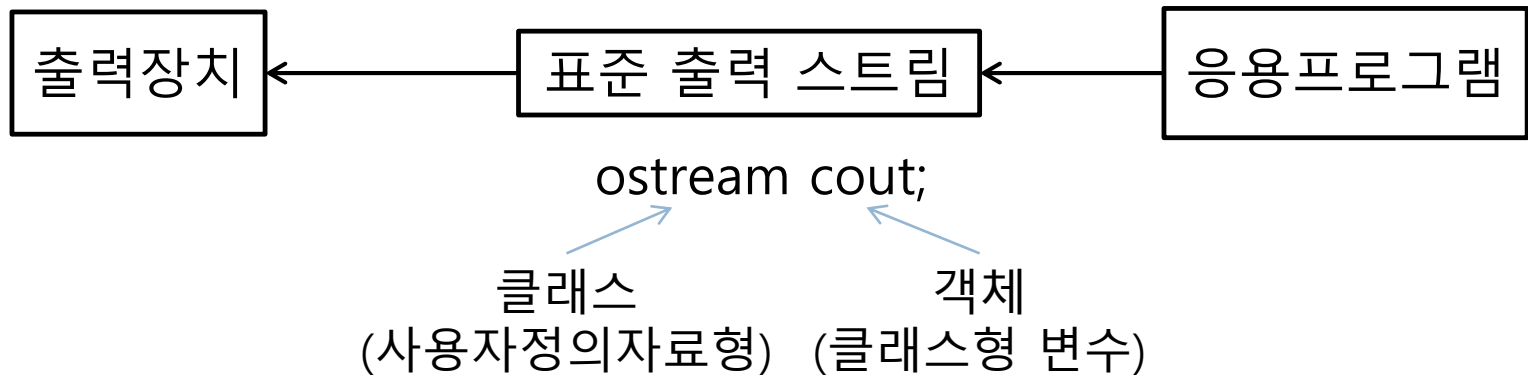
```
std::cout << "Hello\n";           // printf("Hello\n");  
std::cout << "첫번째 맛보기";     // printf("첫 번째 맛보기");
```

화면 출력

8

□ cout 객체

- ▣ 표준 출력 스트림을 나타내는 객체(변수)
- ▣ <iostream> 헤더 파일에 선언되어 있음
- ▣ std 이름 공간에 선언되어 있음 -> std::cout으로 사용



화면 출력

9

□ << 연산자

- ▣ 스트림 삽입 연산자(stream insertion operator)
- ▣ C/C++에서 시프트 연산자임 -> **a << 5;**
- ▣ C++에서는 스트림 삽입 연산자로 재정의 -> **cout << 5;**
- ▣ 왼쪽 cout 객체에 오른쪽 피연산자를 삽입하는 연산자
- ▣ cout 객체에 연결된 화면에 출력됨
- ▣ 여러 개의 << 연산자로 여러 데이터를 출력

```
std::cout << "Hello\n" << "첫 번째 맛보기입니다.";
```

<< 연산자 활용

10

- 문자열 및 기본 타입의 데이터 출력
 - ▣ `bool, char, short, int, long, float, double` 모든 타입의 데이터 출력

```
int n = 3;  
char c = '#';  
std::cout << c << 5.5 << '-' << n << "hello" << true;
```

#5.5-3hello1

- 수식이나 함수 호출문도 가능

```
std::cout << "n + 5 =" << n + 5;           // 수식의 결과값을 출력  
std::cout << f();                          // 함수f()의 리턴값을 출력
```

- 화면에서 줄바꿈 하기 : `'\n'`이나 `std::endl` 조작자(p572) 사용

```
std::cout << "Hello" << '\n';  
std::cout << "Hello" << std::endl;
```

예제 2-2 cout과 <<를 이용한 화면 출력

11

```
#include <iostream>
double area(int r);           // 함수 선언
int main() {
    int n = 3;
    char c = '#';
    std::cout << c << 5.5 << '-' << n << "hello" << true << std::endl;
    std::cout << "n + 5 = " << n + 5 << '\n';
    std::cout << "면적은 " << area(n); // 함수 area()의 리턴 값 출력
    return 0;
}
double area(int r) {          // 함수 정의
    return 3.14 * r * r;      // 반지름 r의 원면적 리턴
}
```

```
#5.5-3hello1
n + 5 = 8
면적은 28.26
```

namespace 개념

12

- 이름 : 함수명, 변수명, 클래스명, 객체명 등 모든 식별자
- 이름 충돌 : 같은 이름의 변수나 함수를 정의할 때 발생
- 실세계에서 이름 충돌 사례



namespace 개념

13

- 이름(identifier) 충돌이 발생하는 경우
 - ▣ 여러 명이 하나의 프로젝트를 서로 나누어 개발하는 경우
 - ▣ 다른 사람이 작성한 소스파일이나 목적 파일을 컴파일 또는 링크하는 경우
 - ▣ 이름 충돌을 해결하는데 많은 시간과 노력이 필요
- namespace 기능
 - ▣ 이름 충돌 해결하기 위한 기능
 - 2003년 새로운 C++ 표준에서 도입
 - ▣ 개발자가 자신만의 이름 공간을 생성할 수 있도록 함
 - 이름 공간 안에 선언된 이름은 다른 이름공간과 별도 구분

namespace 개념

14

□ 이름 공간 생성

```
// kitae.cpp
namespace kitae {                      // kitae 라는 이름 공간 생성
    // 함수, 변수, 상수, 클래스, 객체 선언
    int a;
    void max(){...}
}

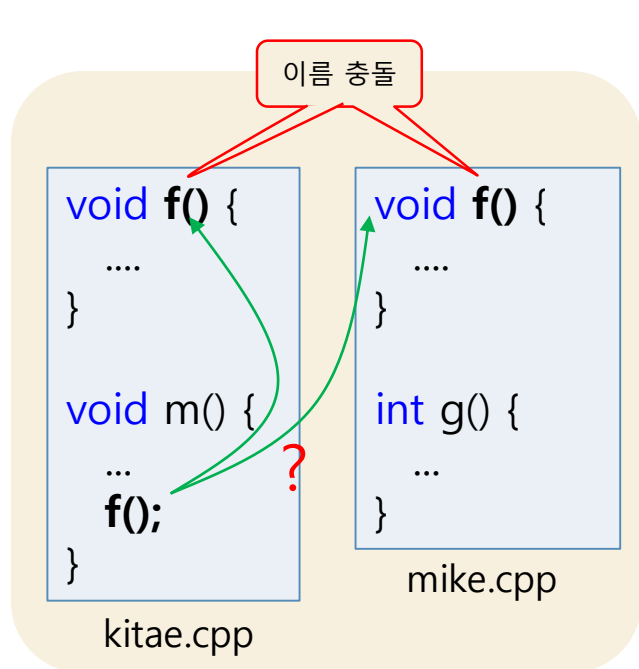
// mike.cpp
namespace mike {                       // kitae 라는 이름 공간 생성
    // 함수, 변수, 상수, 클래스, 객체 선언
    int a;
    void max(){...}
}
```

□ 이름 공간의 사용

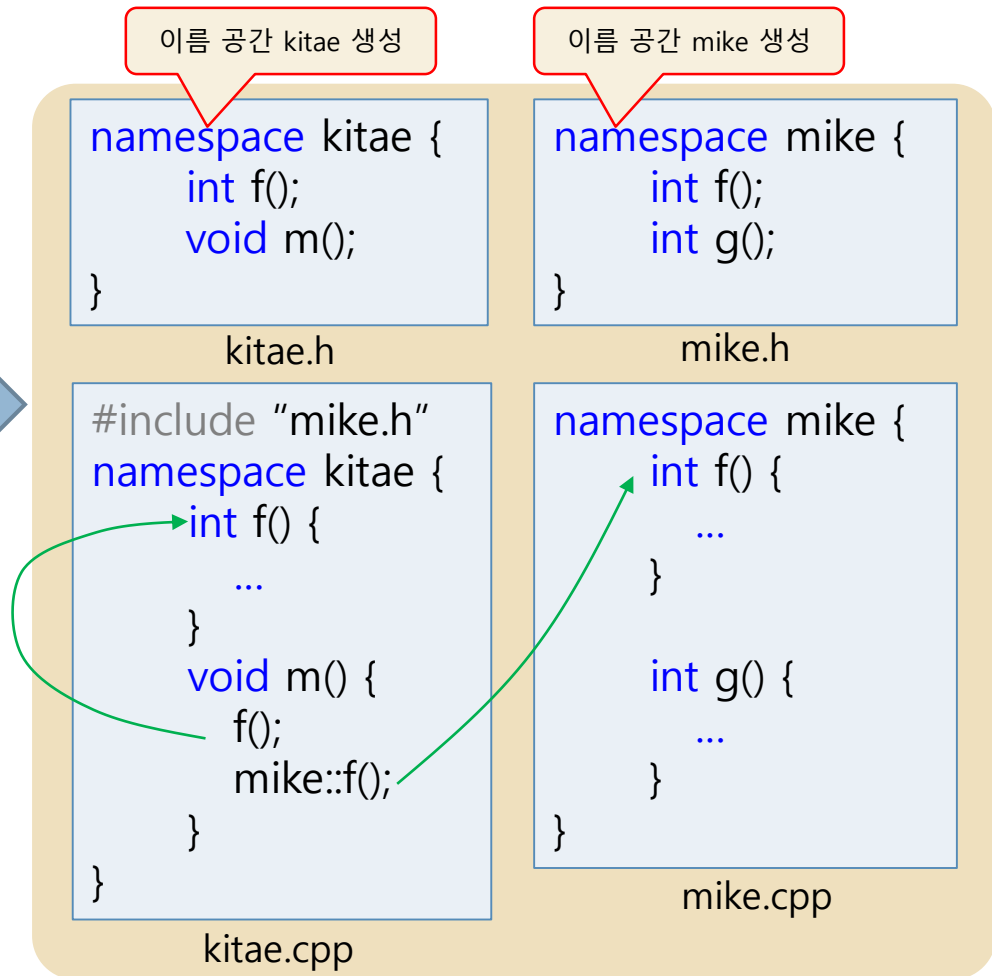
- 이름공간::이름 -> kitae::a, mike::a, kitae::max(), mike::max();

namespace 개념

15



(a) kitae와 mike에 의해 작성된 소스를 합치면 `f()` 함수의 이름 충돌. 컴파일 오류 발생



(b) 이름 공간을 사용하여 `f()` 함수 이름의 충돌 문제 해결

std:: 란?

16

□ std

▣ C++ 표준에서 정의한 이름 공간(namespace)

- 모든 C++ 표준 라이브러리는 std 이름 공간 안에 구현되어 있음
- cout, cin, endl 등

▣ std 이름 공간에 선언된 이름을 접근하기 위해 std:: 접두어 사용

- std::cout, std::cin, std::endl

□ std:: 생략 : using 지시어 사용

```
using std::cout;                // cout에 대해서만 std:: 생략
```

```
....
```

```
cout << "Hello" << std::endl;    // std::cout에서 std:: 생략
```

```
using namespace std;           //std이름공간에 선언된 모든 이름에 std::생략
```

```
....
```

```
cout << "Hello" << endl;         // cout, endl 앞에 모두 std:: 생략
```


#include <iostream>과 std

17

- <iostream> 헤더파일 안에 선언된 모든 클래스, 객체, 함수는 std 이름 공간안에 구현되어 있음
- std:: 를 생략하려면 using namespace std; 문장 필요

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

```
// std이름공간에 선언된 모든 이름에 std::생략
```

예제 2-1: 기본적인 C++ 프로그램

18

```
/*
소스: SimpleC++.cpp
cout과 << 연산자를 이용하여 화면에 출력한다.
*/
#include <iostream>
using namespace std;
int main() {
    cout << "Hello\n";
    cout << " 첫 번째 맞보기입니다. ";
    return 0;
}
```

Hello
첫 번째 맞보기입니다.

논리형(bool)

19

- 논리형(**bool**) : 참(**true**, 1)과 거짓(**false**, 0)을 나타내는 자료형
 - ▣ 관계, 논리 연산 결과를 저장하는데 이용.
 - ▣ 1바이트 메모리가 할당됨
 - ▣ **true**는 1으로 **false**는 0으로 정의되어 있음(상수)
 - ▣ **bool** flag = **true**; // flag에는 1이 저장됨.
 - ▣ **bool** flag = **false**; // flag에는 0이 저장됨.
- bool형은 0이 아닌 값은 **true**로 0은 **false**로 자동 형변환됨.
 - ▣ **bool** c = -1; // c에 true(1)가 대입된다.
 - ▣ **bool** d = 0; // d에 false(0)가 대입된다.

논리형(bool)

20

```
#include <iostream>
using namespace std;
int main()
{
    bool b1 = true, b2 = false;
    cout << true << "\n";
    cout << false << "\n";
    cout << "b1 = " << b1 << "\n";
    cout << "b2 = " << b2 << "\n";
    bool c = -1, d = 0;
    cout << "c = " << c << "\n";
    cout << "d = " << d << "\n";
    return 0;
}
```

```
1
0
b1 = 1
b2 = 0
c = 1
d = 0
```

실습과제1

21

- 다음 결과처럼 나오도록 C++ 프로그램을 작성 하시오.

이름 : 홍길동
주소 : 군산시 대학로 558
학번 : 201012
차종 : 벤츠

실습과제2

22

- 다음 결과처럼 나오도록 C++ 프로그램을 작성 하시오.
- 특수문자의 출력방법은 C언어와 같음

군산대학교 "홍길동"
"축하합니다."
100% ₩취업율₩

실습과제3

23

- 다음 결과처럼 나오도록 C++ 프로그램을 작성 하시오.



실습과제4

24

- 교재 88페이지 실습문제 1번을 푸시오.

실습과제5

25

- 교재 89페이지 실습문제 2번을 푸시오.

과제 제출 방법

26

- 소스코드, 라인단위의 주석, 실행결과를 포함하는 pdf파일을 작성한 후 eclass 과제 게시판에 업로드, **반드시 하나의 pdf파일로 업로드할 것**
- 기한 : 과제 게시판에 마감시간 참조
- 실행결과를 캡처할 때 글자를 알아보기 쉽게 확대해서 캡처할 것.
- 소스코드의 첫 부분은 아래처럼 제목,날짜,작성자(학번,이름)를 작성할 것

```
// *****  
//   제   목   : 정수 4개의 평균을 구하는 프로그램  
//   날   짜   : 2023년 9월10일  
//   작성자   : 15010101 홍길동  
// *****  
  
// 소스코드 작성
```