2020.09.14 - 3주차

객체지향설계

Introduction C++

Content

1. 튜터 소개
2. C++ 문법 소개
3. 객체지향의 특징
4. 실습
5. 과제
6. QnA

튜터 소개

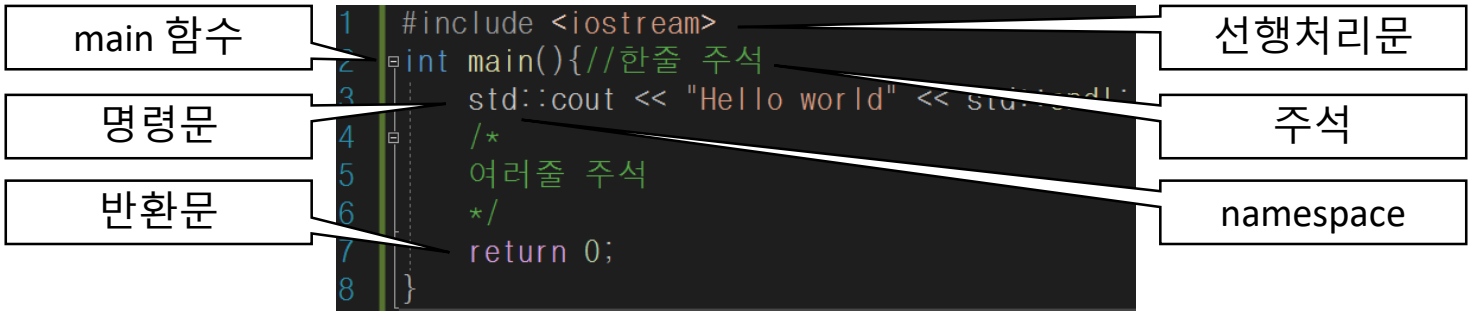
- 튜터
  - ✓ 이름 : 이재용
  - ✓ E-mail : ljy8972@naver.com

- TA
  - ✓ 이름 : 염철민
  - ✓ 소속 : 사이버보안 연구실(632호)
  - ✓ E-mail : ymc12377@naver.com

C++ 문법 소개

- C++
  - ✓ 컴퓨터에 명령을 내리는 기계어를 효율적으로 만들 수 있도록 고안된 고급언어

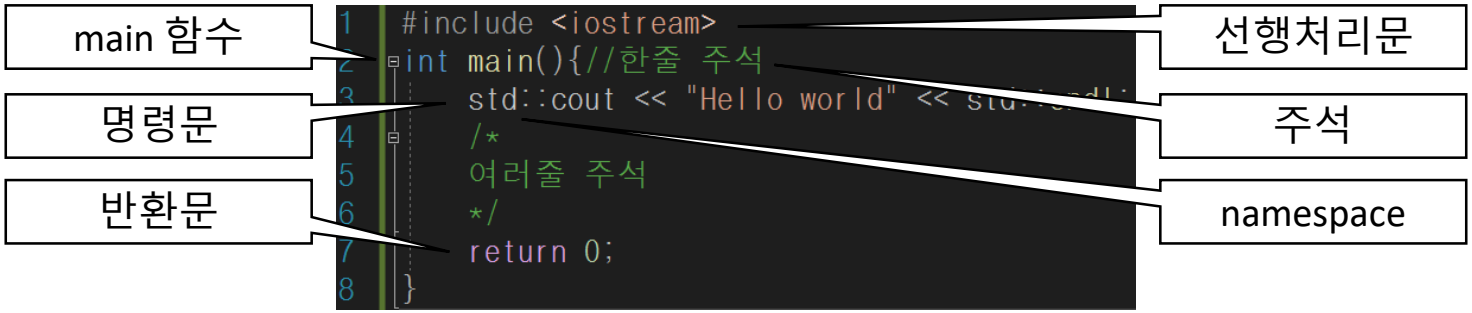
- 문법(1/2)



- ✓ main 함수 : 모든 C++프로그램이 가지고있어야 하며 가장 먼저 실행되는 함수
- ✓ 명령문 : C++프로그램의 동작을 명시하고, 컴퓨터에 동작을 알려주는데 사용
- ✓ 반환문 : 함수의 종료를 의미하며 함수를 호출한 곳으로 결과값을 반환

C++ 문법 소개

➢ 문법(2/2)



- ✓ 선행처리문 : #include, #define과 같이 선행처리기에 처리되는 문장
- ✓ 주석 :코드의 이해를 돕거나 디버깅을 위해 작성하는 일종의 메모
  컴파일러는 주석은 무시하고 컴파일함으로 실행파일에선 확인 불가
- ✓ namespace : 이름이 기억되는 영역, 이름이 소속된 공간을 뜻함
  C++프로그램의 표준 구성요소인 클래스, 함수 변수 등은 std라는
  namespace에 저장

C++ 문법 소개

- iostream
  - ✓ 사용자와 프로그램 사이의 입출력을 담당하는 수단
  - ✓ C++에서는 cout객체로 출력작업을, cin 객체로 입력 작업을 수행
    (기존 C언어 스타일처럼 printf()함수나 scanf()함수 사용 가능)
- cout 객체
  - ✓ 데이터를 출력하는 데 사용되며 미리 정의된 출력 스트림을 나타내는 객체
  - ✓ std::cout ⟨⟨ 출력할데이터;
- cin 객체
  - ✓ 데이터를 입력받는 데 사용되며 미리 정의된 입력 스트림을 나타내는 객체
  - ✓ std::cin ⟩⟩ 저장할변수;

※ 삽입 연산자(⟨⟨)와 추출 연산자(⟩⟩)가 데이터의 흐름을 나타내므로 직관적

※ C++ 표준 입출력 객체는 입출력 데이터의 타입을 자동으로 변환시켜 줌

객체지향의 특징

- 추상화
  - ✓ 객체에 대한 추상화 작업 결과가 클래스
  - ✓ 상태(속성, 관계)와 행동을 공유하는 개체들을 클래스로 정의
  - ✓ 클래스 내부의 정보를 외부로부터 숨김
- 캡슐화
  - ✓ 데이터와 오퍼레이션을 묶는 것을 뜻함
  - ✓ 내부 구조를 숨기고 데이터간 연동은 오퍼레이션을 통해 이뤄짐
  - ✓ 정보은닉 : 인터페이스를 통해서만 접근
  - ✓ 인터페이스 : 서비스를 정의 하는 오퍼레이션 집합, 작업의 분업화 가능, 유지보수 용이

실습 01

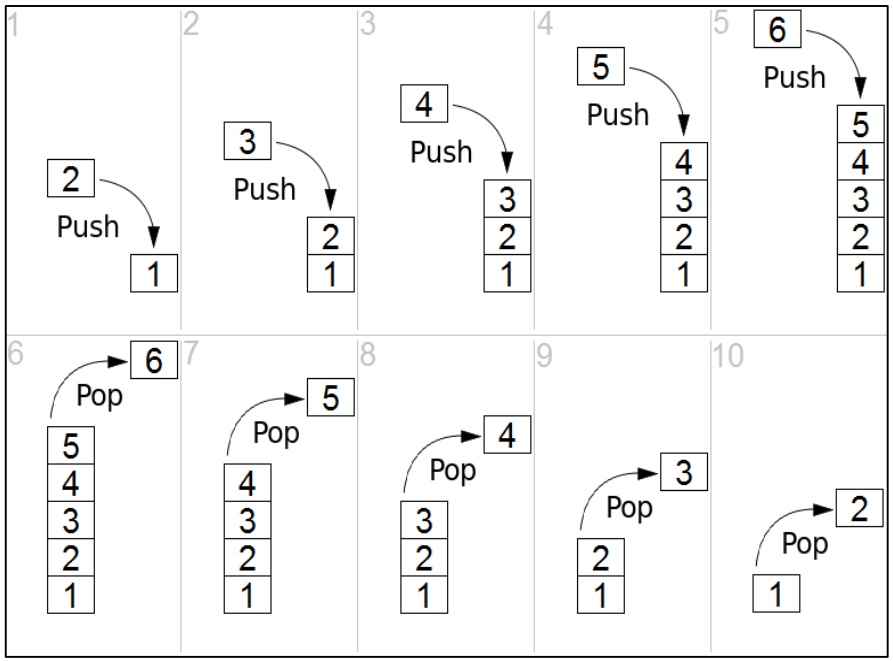
- 아래 코드의 컴파일 에러를 해결하여, main 함수의 cout 구문이 실행 되도록 해보세요. 단, main 함수는 수정하지 말것.

```
#include <iostream>
    class SetValue {
    int x, y;
};
int main() {
    SetValue obj;
    obj.x = 33;
    obj.y = 44;

    std::cout << "X= " << obj.x << " , Y= " << obj.y << std::endl;
    system("pause");
    return 0;
}
```

실습 02

- 아래 프로그램은 C++을 이용하여 간단한 스택을 구현한 것이다. 스택은 데이터를 보관하기 위한 저장소로서, push를 할경우 데이터가 스택에 들어가고, pop을 할 경우, 스택의 가장 위에 있는 데이터가 스택에서 나온다. 스택의 현재 위치를 가르키는 스택 포인터의 값은 pop을 할 경우 줄어들고, push를 할 경우 증가한다.




```
int main() {
	class Stack s;
	s.push(7);
	s.push(88);
	s.push(999);
	std::cout << s.pop() << "을 스택에서 꺼냈습니다." << std::endl;
	std::cout << s.pop() << "을 스택에서 꺼냈습니다." << std::endl;
	std::cout << s.pop() << "을 스택에서 꺼냈습니다." << std::endl;
	system("pause");
	return 0;
}
```

```
C:₩Users₩ymc12₩Desktop₩source₩CUN_OOP_00₩OO
7 이 스택에 들어갔습니다.
88 이 스택에 들어갔습니다.
999 이 스택에 들어갔습니다.
999 을 스택에서 꺼냈습니다.
88 을 스택에서 꺼냈습니다.
7 을 스택에서 꺼냈습니다.
계속하려면 아무 키나 누르십시오 . . .
```

과제 01

- "실습 1"의 코드가 OOP의 추상화 및 캡슐화 개념을 만족하도록 바꾸어 보세요.
- 단, SetValue 클래스의 기존 구문은 수정하지 않고 새로운 멤버 함수만 추가하여 수정할 것.

  - ✓ 추상화 : 클래스 내부의 정보를 외부로부터 숨김
  - ✓ 캡슐화: 클래스 멤버 변수와 멤버 함수와의 연결 (binding) --〉 내부 변수 접근을 위한 인터페이스 제공

과제 02

- 다음 프로그램의 확장자를 .c 와 .cpp의 두 가지 경우로 실행(컴파일)해 보세요.
- 각 경우에 에러가 나거나 나지 않는 이유에 대해 생각해 보세요.

```c
#include <stdio.h>

void print(int var){
    printf("Integer number: %d ₩n", var);
}

void print(float var){
    printf("Float number: %f ₩n", var);
}

void print(int var1, float var2){
    printf("Integer number: %d ₩n", var1);
    printf(" and float number: %f", var2);
}
```

```c
int main(){
    int a = 7;
    float b = 9;

    print(a);
    print(b);
    print(a, b);

    return 0;
}
```

과제 제출

- 제출형식
  - ✓ 사이버캠퍼스에 보고서 및 깃헙 링크 업로드

- 보고서 작성방법
  - ✓ 표지 : 제출일, 학번 이름, 주차(week03)
  - ✓ 과제 수행 과정, 실행결과 캡쳐 후 결과분석
  - ✓ 과제를 수행하며 어려웠던 점 또는 새로 알게 된 부분을 고찰한 내용
  - ✓ 과제의 해답 코드를 깃허브에 올리고 링크를 보고서에 기입
  - ✓ 파일형식 : PDF

QnA

사이버캠퍼스 질문게시판 또는
ljy8972@naver.comfh 이나
ymc12377@naver.com 로 메일