Chap3

클래스/객체

클래스: 객체를 찍어내는 틀
멤버 함수, 멤버 변수 등 선언
객체: 클래스에서 찍어내는 실존하는 실체=인스턴스
객체들은 상호 별도 공간에 생성

클래스 선언부

- 멤버함수와 멤버 변수의 원형을 선언
- 클래스 마지막 {}; 세미콜론 잊지 않기~
- 선언부에서 변수 초기화 가능
- 접근지정자 사용하기도 함

```
public:
    ... -> 클래스 외부에서 접근 가능
```

*디폴트는 private !!

클래스 구현부

- 멤버함수를 구현
- 범위 지정 연산자 :: 를 사용

```
리턴타입 클래스명::함수명(매개변수){
    ...구현...}
```

* 선언부,구현부 나누는 이유
=> 클래스의 재사용(같은 이름의 함수,변수 가지고 구현만 바꿔서 사용 가능)

객체의 멤버에 접근
- 객체이름 . 멤버

생성자

- 객체가 생성될 때 자동으로 실행
- 객체를 초기화하는 작업(멤버변수 값 설정,메모리 동적할당, 네트워크 연결 등등)
- 여러개(함수중복) 만들 수 있지만 하나, 한번만 실행
- 생성자 함수 이름 = 클래스 이름
- 리턴타입X(void도 X)

위임 생성자

- 생성자가 다른 생성자를 호출 하는 것
- 초기화 코드가 중복되는 것을 막을 수 있음

```
Circle::Circle() : Circle(1) { }
Circle::Circle(int r){
    radius = r;
    ...
}
```

생성자 서두에서 초기화
- 클래스이름 :: 생성자이름(=클래스이름)( ) : 변수1(값),변수2(값)
{ }

```
Circle::Circle( ) : x(0),y(0) { }

Circle::Circle(int a) : x(a),y(0) { }

Circle::Circle(int a) : x(a+100),y(0) { }
```

기본 생성자

- 생성자는 반드시 있어야한다!!
- 하지만 정의되는 생성자 함수가 없을시, 기본 생성자 자동생성
=> Circle( )
- 하나라도 생성자가 선언되어 있으면 기본 생성자 생성 X

소멸자

- 객체 소멸 시 실행 -> 객체 메모리 반환(자동호출)
- 객체 메모리 반환, 파일 저장/닫기, 네트워크 닫기 작업 등등
- Circle::~Circle( ){ ... }
- 리턴 타입X
- 오직 한 개만!
- 없으면 자동으로 기본 소멸자 생성
- return 0; 문이 실행되면 객체는 생성 반대순으로 소멸

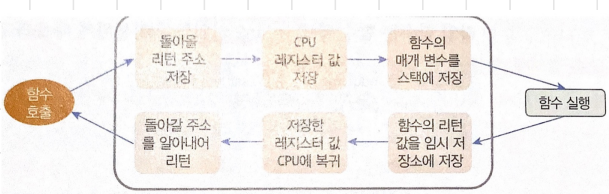
* 생성: 전역객체->지역객체
소멸: 지역객체의 가장 나중에 생성됐던 객체->...

접근지정자

- private(디폴트), public(생성자),protected(상속:부모에)

인라인 함수

- 함수가 호출되면 실행을 마치고 다시 호출된 곳으로 돌아오는 과정에서 오버헤드 발생 (오버헤드 크면 효율성 down)
ex) 반복문 안에서 함수 호출



- inline 키워드 사용(그럼 그 함수는 호출 시, 호출된 장소에 삽입)
- 인라인 함수를 호출하는 곳에서 험수 코드를 그대로 삽입
- 속도 향상 / 전체 크기 늘어남..
- 컴파일러에 따라 무시 가능(재귀,static,반복,switch,goto)
- 함수의 크기가 작을 경우, 함수 구현을 선언부에서 해도 된다

구조체
- 클래스와 동일한 기능 가짐
- struct 키워드로 선언 (구조체 타입의 객체 생성 시엔 키워드 x)
- 구조체 생성자 존재
- 디폴트 접근 지정자는 public

* 헤더파일과 cpp파일의 분리(main.cpp도 분리될 수 있음..)
클래스 선언부 -> 헤더파일
클래스 구현부 -> cpp파일

=> 헤더파일, 클래스 구현부 cpp파일, main cpp파일 모두가
링킹되어 실행파일 exe를 만든다

* 헤더파일을 분리시 cpp 파일에서 필요한 헤더파일을 include
해야하는데 이때 include가 중복되면 오류가 발생한다
따라서 오류 발생을 방지하기 위해서
헤더파일에서
#ifnedf ...
#endif ...
를 사용한다.

cpp파일에서 include가 중복되면 중복 검사 후 미포함한다
(처음 include 사용 시엔 #define ... 이 정의된다