

03

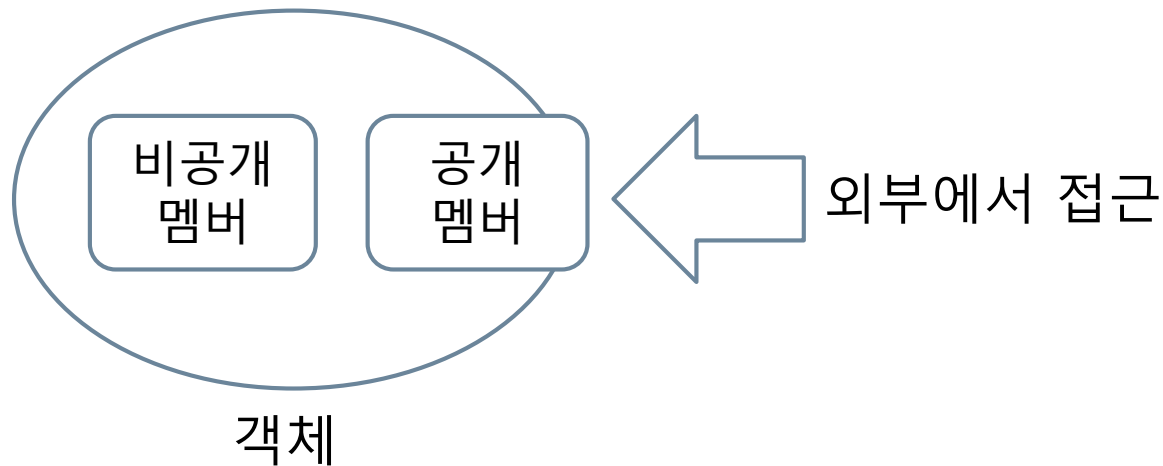
학습 목표

1. 실세계의 객체와 C++ 객체에 대해 이해한다.
2. C++ 클래스를 작성할 수 있다.
3. 객체를 생성하고 활용할 수 있다.
4. 생성자와 소멸자를 알고 작성할 수 있다.
5. `private`, `protected`, `public` 접근 지정자를 이해한다.
6. 인라인 함수의 목적을 이해하고 활용할 수 있다.

접근 지정자

3

- 객체의 캡슐화를 구현하기 위한 도구
- 클래스 멤버별로 클래스 외부로부터 접근 허용여부를 지정해 줌
- 클래스 외부에서 접근가능한 공개 멤버와 외부의 접근을 허용하지 않는 비공개 멤버를 지정가능
- 객체의 중요한 멤버를 보호를 비공개하고 외부와의 인터페이스를 위한 일부는 공개



접근 지정자

4

- **private** : 클래스 내부의 멤버 함수에서만 접근 가능, 클래스 외부에서는 접근 불가능
- **public** : 클래스 내,외부의 프로그램의 모든 함수에서 접근 허용
- **protected** : 클래스 자신과 상속받은 자식 클래스에만 허용(8장)

```
class Sample {  
private:  
    // private 멤버 선언  
public:  
    // public 멤버 선언  
protected:  
    // protected 멤버 선언  
};
```

접근 지정자

5

- 접근지정자가 선언되면 다른 접근 지정자가 선언될 때까지 모든 멤버에 적용됨

```
class Sample {  
private:  
    int x, y;        // private 멤버  
    void f1();       // private 멤버  
public:  
    int a, b;        // public 멤버  
    void f2();       // public 멤버  
protected:  
    int i, j;        // protected 멤버  
    void f3();       // protected 멤버  
};
```

중복 접근 지정과 디폴트 접근 지정

6

- 접근지정은 여러 번 중복 사용가능

```
class Sample {  
private:  
    int x, y;  
public:  
    Sample();  
private:  
    bool checkXY();  
};
```

중복 접근 지정과 디폴트 접근 지정

7

- 접근 지정자가 없는 경우는 private

```
class Circle {  
    int radius;  
public:  
    Circle();  
    Circle(int r);  
    double getArea();  
};
```

=

```
class Circle {  
private:  
    int radius;  
public:  
    Circle();  
    Circle(int r);  
    double getArea();  
};
```

접근 지정자 원칙

8

- 멤버 변수는 private이 원칙(캡슐화)
- 멤버 함수는 public이 원칙(외부에 일부공개)
- 생성자는 public이 원칙(객체 생성시 외부에서 호출)
- 클래스 외부에서는 멤버 함수를 통하여 멤버 변수에 접근하도록 함 -> 멤버 함수안에서 데이터의 잘못된 저장 및 사용을 방지

예제 : 멤버를 public으로 선언한 나쁜 사례

9

```
class Circle {  
public:  
    int radius;           // 멤버 변수를 보호하지 못함  
    Circle();  
    Circle(int r);  
    double getArea();  
};  
Circle::Circle() { radius = 1; }  
Circle::Circle(int r) { radius = r; }  
int main() {  
    Circle waffle;  
    waffle.radius = -5; // public멤버는 외부에서 마음대로 접근  
}
```

예제 : 멤버 변수는 private 지정

10

```
class Circle {  
private:  
    int radius;           // 멤버 변수를 외부로부터 보호  
public:  
    Circle();  
    Circle(int r);  
    double getArea();  
};  
Circle::Circle() { radius = 1; }  
Circle::Circle(int r) { radius = r; }  
int main() {  
    Circle waffle(5);     // 생성자에서 radius 설정  
    waffle.radius = -5;   // error, private 멤버 외부에서 접근불가  
}
```

예제 : 생성자는 public 지정

11

```
class Circle {  
private:  
    int radius;  
    Circle();  
    Circle(int r);  
public:  
    double getArea();  
};  
Circle::Circle() { radius = 1; }  
Circle::Circle(int r) { radius = r; }  
int main() {  
    Circle circle; // error  
}
```



```
class Circle {  
private:  
    int radius;  
public:  
    Circle();  
    Circle(int r);  
    double getArea();  
};  
Circle::Circle() { radius = 1; }  
Circle::Circle(int r) { radius = r; }  
int main() {  
    Circle circle; //정상동작  
}
```

(a) 생성자를 private으로 선언한 경우

(b) 생성자를 public으로 선언한 경우

클래스 내부에서 멤버 접근

12

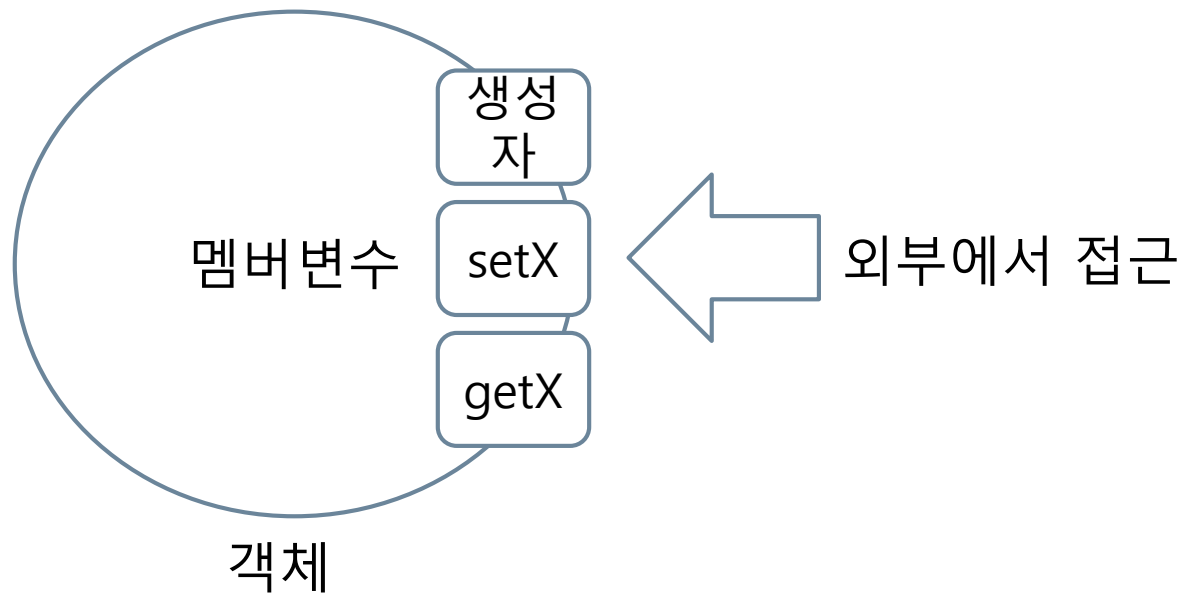
- 접근지정자는 클래스 외부에서 접근을 제한 하는 기능
- private 멤버는 클래스 외부에서 접근이 제한 된다는 의미
- 클래스 내부의 멤버함수에서 멤버변수에 접근은 항상 가능

```
class Circle {  
private:  
    int radius;  
public:  
    Circle();  
    Circle(int r);  
    double getArea();  
};  
Circle::Circle() { radius = 1; }  
Circle::Circle(int r) { radius = r; }  
double Circle::getArea(int r) { return 3.14*radius*radius; }
```

멤버 접근 함수(setter/getter)

13

- 멤버변수를 간접적으로 접근할 수 있도록 멤버변수를 설정하는 함수 setXxx(), 멤버 변수를 리턴하는 함수 getXxx()를 항상 만들 것.
- 생성자는 객체생성시 초기화 용도이고 setXxx()/getXxx() 함수는 객체 생성 이후에 값을 설정 및 읽을 때 사용



멤버 접근 함수(setter/getter)

14

```
// 멤버변수에 값 쓰기(set, write)
```

```
void Circle::setRadius(int r)
```

```
{
```

```
    if(r<0) // 에러처리코드 작성
```

```
    else radius = r;
```

```
}
```

```
// 멤버변수의 값 읽기(get, read)
```

```
int Circle::getRadius()
```

```
{
```

```
    return radius;
```

```
}
```

예제 : 멤버 접근 함수

15

```
class Circle {  
private:  
    int radius;  
public:  
    Circle();  
    Circle(int r);  
    void setRadius(int r);  
    int getRadius();  
    double getArea();  
};  
Circle::Circle() { radius = 1; }  
Circle::Circle(int r) { radius = r; }  
void Circle::setRadius(int r) { radius = r; } // 멤버변수에 값 쓰기  
int Circle::getRadius() { return radius; } // 멤버변수의 값 읽기  
double Circle::getArea() { return 3.14*radius*radius; }
```

예제 : 멤버 접근 함수

16

```
int main()
{
    Circle cir;
    // cir.radius = 10; //직접 참조는 오류
    cir.setRadius(10);
    // cout << "반지름은" << cir.radius << endl; // 직접 참조는 오류
    cout << "반지름은" << cir.getRadius() << endl;
    cout << "면적은" << cir.getArea() << endl;

    return 0;
}
```


예제3-9 컴파일 오류가 발생하는 곳은 어디인가?

17

```
#include <iostream>
using namespace std;
class PrivateAccessError {
private:
    int a;
    void f();
    PrivateAccessError();
public:
    int b;
    PrivateAccessError(int x);
    void g();
};
PrivateAccessError::PrivateAccessError() {
    a = 1;           // (1)
    b = 1;           // (2)
}
PrivateAccessError::PrivateAccessError(int x) {
    a = x;           // (3)
    b = x;           // (4)
}
```

예제 3-9 컴파일 오류가 발생하는 곳은 어디인가?

18

```
void PrivateAccessError::f() {  
    a = 5;                // (5)  
    b = 5;                // (6)  
}  
void PrivateAccessError::g() {  
    a = 6;                // (7)  
    b = 6;                // (8)  
}  
int main() {  
    PrivateAccessError objA;    // (9) - 오류  
    PrivateAccessError objB(100); // (10)  
    objB.a = 10;                // (11) - 오류  
    objB.b = 20;                // (12)  
    objB.f();                   // (13) - 오류  
    objB.g();                   // (14)  
}
```

C++ 언어의 특징

19

- 캡슐화(encapsulation)
 - ▣ 데이터와 함수를 하나의 캡슐(객체)안에 모아 놓고 외부로부터 안전하게 보호하는 기능
- 정보은닉(information hiding)
 - ▣ 멤버변수를 private으로 선언하고 해당 변수를 접근하는 함수를 별도로 정의해서 안전한 형태로 멤버 변수에 접근할 수 있도록 하는 기능
 - ▣ 잘못된 접근을 하더라도 변수의 오염을 방지하고 컴파일 과정에서 쉽게 발견이 가능함
- C언어는 프로그래머가 조심해야 하고 C++언어는 문법에서 오류를 방지하는 기능을 지원함

실습과제1

20

- 아래 코드는 객체지향형 언어의 철학을 위반하고 있다. 올바르게 수정하라.

```
class Circle {  
public:  
    int radius;  
    Circle();  
};  
Circle::Circle() { radius = 1; }  
int main() {  
    Circle waffle;  
    waffle.radius = 5;  
    cout << "원의 반지름은 " << waffle.radius << endl;  
    return 0;  
}
```

원의 반지름은 5

실습과제2

21

- 삼각형 클래스 Triangle 를 만들고 모든 멤버변수는 private으로 지정하고 다음처럼 동작하도록 모든 기능을 구현하시오.

// 코드추가

```
int main() {  
    Triangle tri;  
    tri.setWidth(3);  
    tri.setHeight(5);  
    cout << "삼각형의 폭은 " << tri.getWidth() << endl;  
    cout << "삼각형의 높이는 " << tri.getHeight() << endl;  
    cout << "삼각형의 면적은 " << tri.getArea() << endl;  
    return 0;  
}
```

폭3,높이5 삼각형생성
삼각형의 폭은 3
삼각형의 높이는 5
삼각형의 면적은 7.5
폭3,높이5 삼각형소멸

실습과제3

22

- 삼각형 클래스 Triangle를 만들고 모든 멤버변수는 private으로 지정하고 다음처럼 동작하도록 모든 기능을 구현하시오.

// 코드추가

```
int main() {  
    Triangle tri(-10,-5);  
    tri.setWidth(-3);  
    tri.setHeight(-5);  
    tri.setWidth(3);  
    tri.setHeight(5);  
    cout << "삼각형의 폭은 " << tri.getWidth() << endl;  
    cout << "삼각형의 높이는 " << tri.getHeight() << endl;  
    return 0;  
}
```

길이는 양수여야함, 대신에 폭1,높이1 삼각형생성
폭은 양수이어야 함
높이는 양수이어야 함
폭3,높이5 삼각형소멸

과제 제출 방법

23

- 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
```