Class Definition in C++

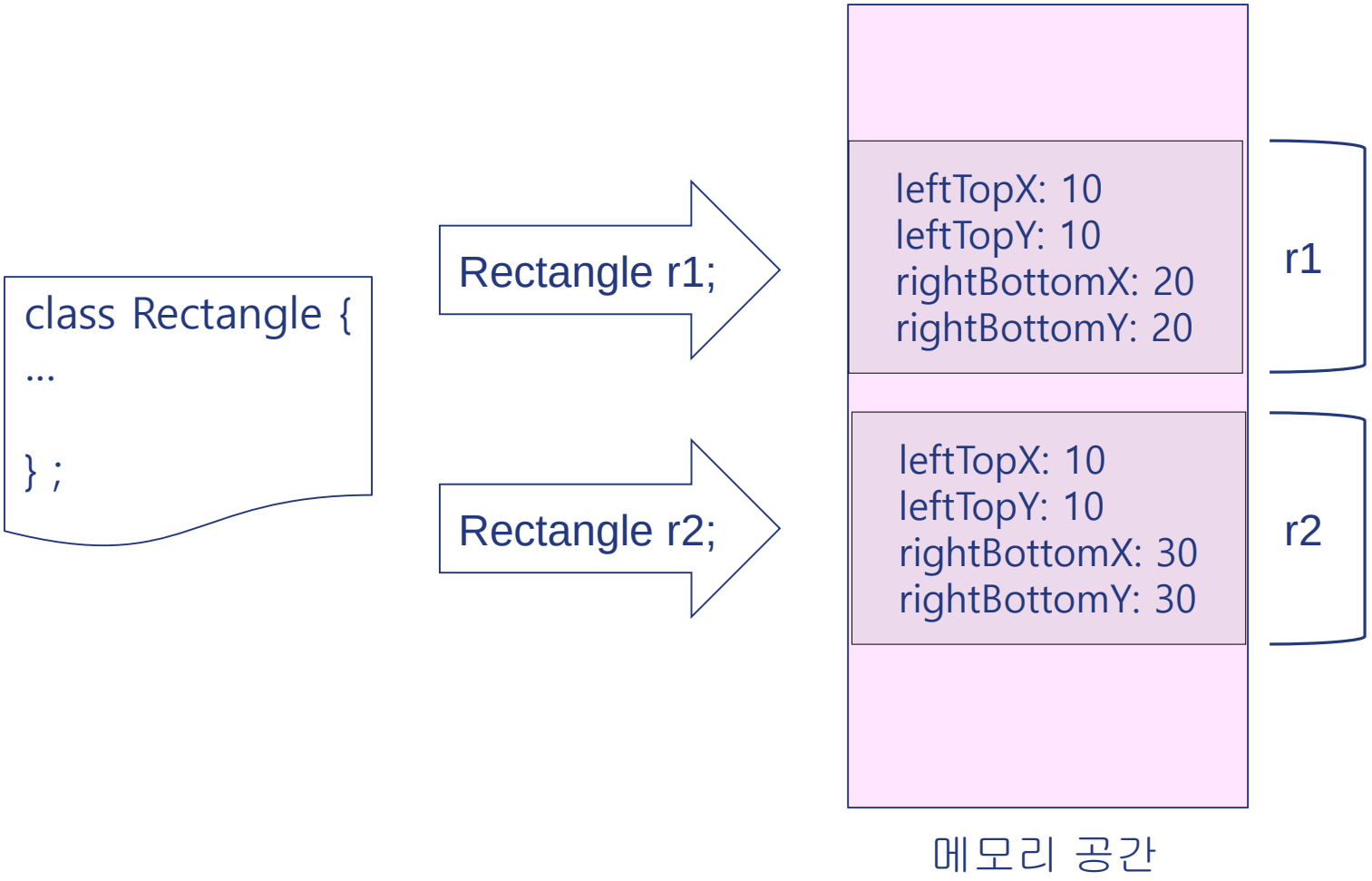
```
class 이름 {
  private:
    데이터 멤버
  public:
    멤버 함수
};
```

Class Rectangle

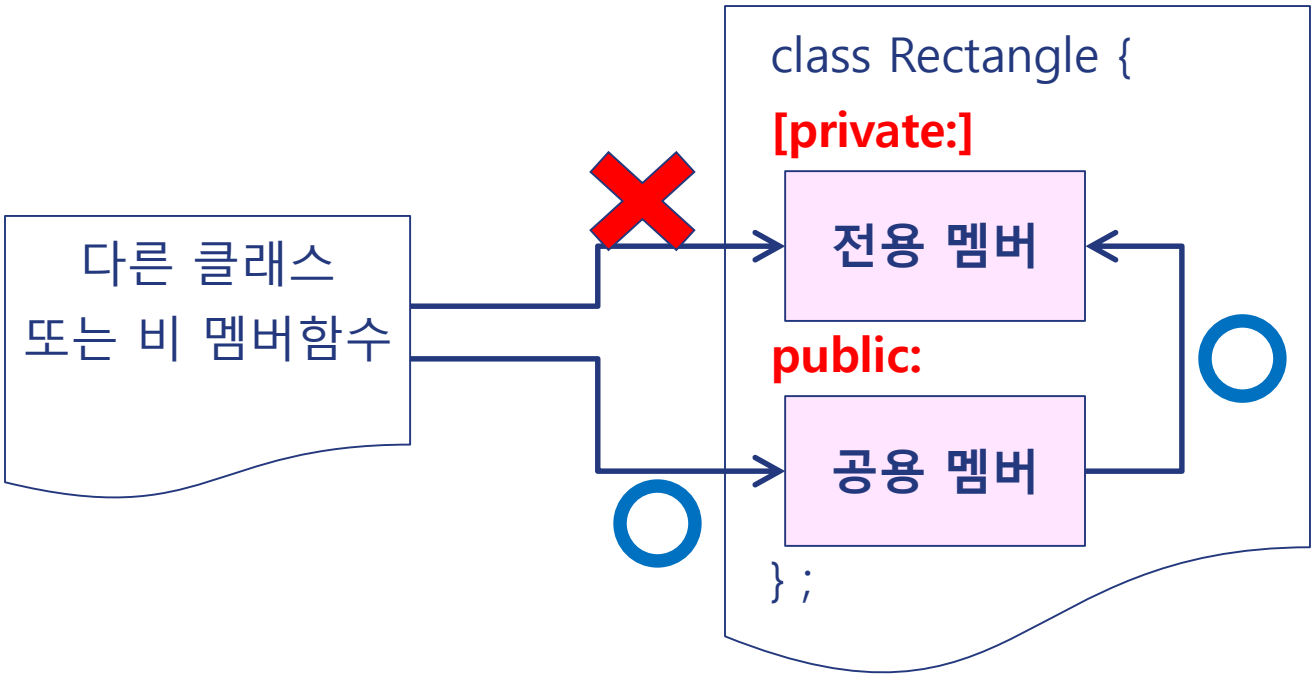
```
# include <iostream>
using namespace std ;

class Rectangle {
private:
  int leftTopX, leftTopY ;
  int rightBottomX, rightBottomY ;
public:
  Rectangle(int x1, int y1, int x2, int y2) { //생성자
    set(x1, y1, x2, y2) ;
  }
  void set(int x1, int y1, int x2, int y2) {
    leftTopX = x1 ; leftTopY = y1 ;
    rightBottomX = x2 ; rightBottomY = y2 ;
  }
  void getLeftTop(int& x, int& y) { x = leftTopX ; y = leftTopY ; }
  void getRightBottom(int& x, int& y) { x = rightBottomX ; y = rightBottomY ; }
  int getArea() { return (rightBottomX - leftTopX) * (rightBottomY - leftTopX) ; }
} ;
```

```
int main() {
    Rectangle r1 ;
    r1.set(10, 10, 20, 20);
    int x1, y1, x2, y2 ;
    r1.getLeftTop(x1, y1) ;
    r1.getRightBottom(x2, y2) ;

    Rectangle r2(x1+10, y1+10,
       x2+10, y2+10) ;

    cout << r1.getArea() << '\t' <<
       r2.getArea() << endl ;
}
```

객체의 생성


class Rectangle {
…
} ;
Rectangle r1;
Rectangle r2;
leftTopX: 10
leftTopY: 10
rightBottomX: 20
rightBottomY: 20
r1
leftTopX: 10
leftTopY: 10
rightBottomX: 30
rightBottomY: 30
r2
메모리 공간

정보은닉 (Information Hiding)



클래스의 정의: Rectangle.h

```
# ifndef __RECTANGLE_H
# define __RECTANGLE_H
class Rectangle {
//private:   // default
    int leftTopX, leftTopY ;
    int rightBottomX, rightBottomY ;
    // 클래스 내부에서 정의된 멤버 함수는 기본적으로 inline 함수임
    void setLeftTop(int x, int y) { leftTopX = x ; leftTopY = y ; }
    void setRightBottom(int x, int y) { rightBottomX = x ; rightBottomY = y ; }
public:
    // 생성자가 정의되지 않았으면 컴파일러가 자동으로 생성함: Rectangle() {}
    // 자신의 멤버 함수를 호출함
    void set(int x1, int y1, int x2, int y2) { setLeftTop(x1, y1) ; setRightBottom(x2, y2) ; }
    void getLeftTop(int& x, int& y) { x = leftTopX ; y = leftTopY ; }
    void getRightBottom(int& x, int& y) { x = rightBottomX ; y = rightBottomY ; }

    int getWidth() { return rightBottomX - leftTopX ; }
    int getHeight() { return rightBottomY - leftTopY ; }

    // 별도의 구현 파일을 이용함
    int getArea() ;
    void moveBy(int deltaX, int deltaY) ;
} ;
# endif
```

Rectangle.cpp

```
// Rectangle.cpp

# include "Rectangle.h"

// 클래스 멤버 함수를 클래스 외부에 정의하고 있음
int Rectangle::getArea() {
    return getWidth() * getHeight() ;
}

void Rectangle::moveBy(int deltaX, int deltaY) {
    setLeftTop(leftTopX+deltaX, leftTopY+deltaY) ;
    setRightBottom(rightBottomX+deltaX, rightBottomY+deltaY) ;
}
```

RectangleMain.cpp

```
# include <iostream>
# include "Rectangle.h"
using namespace std ;

int main() {
    int x1, y1, x2, y2 ;
    cin >> x1 >> y1 >> x2 >> y2 ;
     //객체를 생성함
    Rectangle r1 ;
    r1.set(x1, y1, x2, y2) ;

    int x3, y3, x4, y4 ;
    r1.getLeftTop(x3, y3) ;
    r1.getRightBottom(x4, y4) ;

    Rectangle r2 ;
    r2.set(x3, y3, x4, y4) ;
    r2.moveBy(10, 20) ;

    cout << endl << r1.getArea();
    cout << '₩t' << r2.getArea() << endl ;
}
```

```
# include <iostream>
# include "Rectangle.h"
using namespace std ;

int main() {
  int x1, y1, x2, y2 ;
  cin >> x1 >> y1 >> x2 >> y2 ;

  Rectangle r1 ;
  r1.set(x1, y1, x2, y2) ;          // OK
  r1.leftTopX = r1.leftTopX + 1 ; // ERROR

  Rectangle r2 ;
  r2.setLeftTop(x3, y3) ;           // ERROR
  r2.setRightBottom(x4, y4) ; // ERROR
  r2.set(x3, y3, x4, y4) ;          // OK

  cout << endl << r1.getArea() << '₩t'
    << r2.getArea() << endl ;
}
```

Good Design:
Information hiding (C.9)

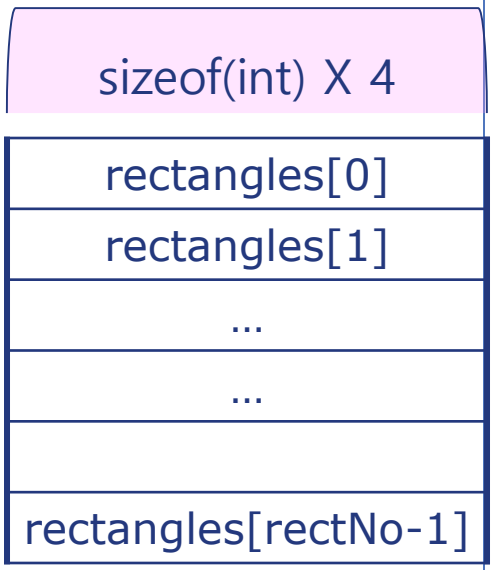
- the characteristics of a module where inessential information of the module is hidden from the outside.
- Information Hiding can promote
  - Maintainability of class: Changes to the hidden members do not cause any additional modification of other classes
  - Understandability of classes: Only the exposed members need to be understood to

```
class Distance {
public:
    double meters() const { return magnitude*unit; }
    void set_unit(double u){
            // ... check that u is a factor of 10 ...
            // ... change magnitude appropriately ...
            unit = u;
    }
private:
    double magnitude;
    double unit;    // 1 is meters, 1000 is kilometers, 0.001 is millimeters, etc.
};
```

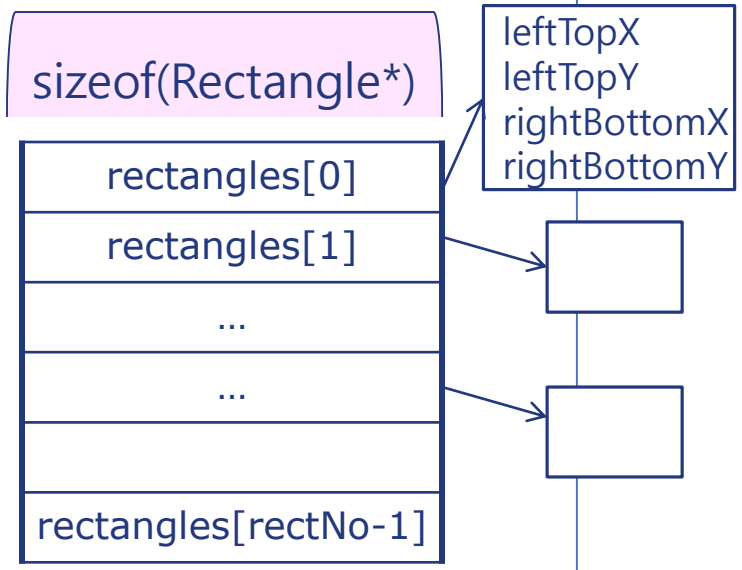
https://github.com/isocpp/CppCoreGuidelines

객체의 동적 생성

```
# include <iostream>
# include "Rectangle.h"
using namespace std ;

int main() {
   int rectNo ;
   cin >> rectNo ;
   //heap 영역에 생성 가능
   Rectangle* const rectangles = new Rectangle[rectNo] ;

   for ( unsigned int i = 0 ; i < rectNo ; i ++ ) {
      cout << "Enter Rectangle information" << endl ;
      int x1, y1, x2, y2 ;
      cin >> x1 >> y1 >> x2 >> y2 ;
      rectangles[i].set(x1, y1, x2, y2) ;
   }
   int totalArea = 0 ;
   for ( unsigned int i = 0 ; i < rectNo ; i ++ )  totalArea += rectangles[i].getArea() ;

   delete [] rectangles ;
   cout << "The total area: " << totalArea << endl ;
}
```



```
int main() {
   int rectNo ;
   cin >> rectNo ;
   Rectangle** const rectangles = new Rectangle*[rectNo] ;

   int count = 0 ;
   while ( count < rectNo) {
     string command ;
     cin >> command ;
     if ( command == "ADD" ) {
       const Rectangle* const r = readRectangle() ;
       rectangles[count] = r ;
       count ++ ;
     }
     else if ( command == "AREA" ) {
       cout << getTotalArea(rectangles, count) << endl ;
     }
     else if ( command == "CLEAR" ) {
       deleteRectangles(rectangles, count) ;
       count = 0 ;
     }
     else {
       cerr << "Invalid command!" << endl ;
     }
   }
   delete [] rectangles ;
}
```



```
# include <iostream>
# include <string>
# include "Rectangle.h"
using namespace std ;

Rectangle* readRectangle() {
  int x1, y1, x2, y2 ;
  cin >> x1 >> y1 >> x2 >> y2 ;

  Rectangle* const r = new Rectangle ;
  r->set(x1, y1, x2, y2) ;
  return r ;
}
void deleteRectangles(Rectangle ** const ppR, int n) {
  for ( int i = 0 ; i < n ; i ++ ) {
    delete ppR[i] ;
     ppR[i] = '₩0' ;
   }
}
int getTotalArea(Rectangle **  const ppR, int n) {
  int totalArea = 0 ;
  for ( int i = 0 ; i < n ; i ++ )
    totalArea += ppR[i]->getArea() ;
  return totalArea ;
}
```