6.1 함수중복

함수중복 같은 함수를 여러 개 만드는 것

다형성의 사례

함수중복 조건 중복된 함수의 이름이 동일해야 함

중복된 함수들은 매개변수 타입이나 매개변수의 개수가 달라야 함

함수 중복에 리턴 타입은 고려되지 않음

함수 중복의 성공 사례 컴파일러가 매개변수 개수와 타입에 따라 중복된 함수를 찾아 연결함

ex) int sum(int a, int b, int c) {

return a + b + c;

}

double sum(double a, double b) {

return a + b;

}

int sum(int a, int b) {

return a + b;

}

int main() {

cout << sum(2, 5, 33) << endl;

cout << sum(12.5, 33.6) << endl;

cout << sum(2,6) << endl;

}

함수 중복 실패 사례 함수이름, 매개변수 타입, 개수가 모두 같고 리턴타입만 다른 경우, 컴파일러가 어떤 함수인지 구분 불가

ex) int sum(int a, int b) {

return a + b;

}

double sum(int a, int b) {

return (double)(a + b);

}

int main() {

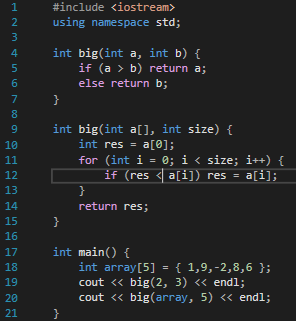
cout << sum(12.5, 33.6) << endl;

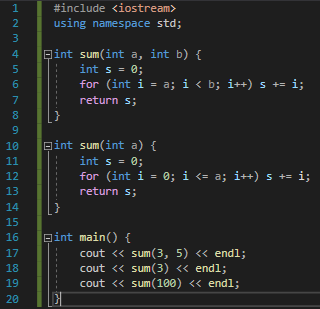
cout << sum(2,6) << endl;

}

함수중복의 편리함 작성 편리

함수를 잘못 호출하는 실수를 줄일 수 있음

예제6-1 

예제6-2 

생성자 함수 중복 이미 3,4장에서 생성자 함수를 중복 작성해봄

다양한 형태로 초깃값을 전달하기 위해 생성자 함수를 중복 선언함 ex)string 객체

소멸자 함수 중복 소멸자는 매개변수를 가지지 않기 때문에 중복 불가능

check time 1. 4

2. 이름, 매개변수

6.2 디폴트 매개 변수

디폴트 매개 변수 = 기본 매개 변수

함수가 호출될 때 매개변수에 값이 넘어오지 않는다면, 미리 정해진 디폴트 값을 받도록 선언된 매개변수

디폴트 매개 변수 선언 매개변수 = 디폴트 값 형태

ex) void star(int a = 5);

star(); //star(5)와 동일

ex) void msg(int id, string text = "Hello");

msg(10);

msg(20, "Good Morning");

msg() //컴파일 오류

디폴트 매개 변수의 제약조건 디폴트 매개 변수는 모두 끝 쪽에 몰려 선언되어야 함

ex) void calc(int a, int b = 5, int c, int d = 0); //컴파일 오류

ex) void calc(int a, int b = 5, int c = 0, int d = 0) //컴파일 성공

매개 변수의 값을 정하는 규칙 컴파일러는 함수 호출문에 나열된 실인자값들을 앞에서부터 순서대로 함수 매개변수에 전달

나머진 디폴트 값으로 전달

디폴트매개변수만 가진 함수 ex) void square(int width = 1, int height = 1);

square(); //square(1,1)

square(5); //square(5,1)

square(3, 8); //square(3,8)

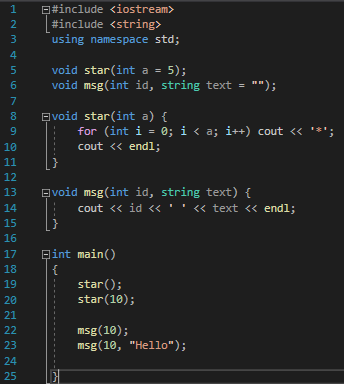
디폴트매개변수와 보통매개변수를 가진 함수 ex) void g(int a, int b = 0, int c = 0, int d = 0);

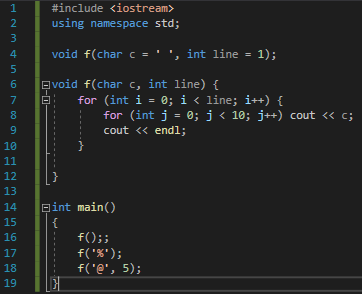
g(10); //g(10,0,0,0)

g(10, 5); //g(10,5,0,0)

g(10, 5, 20); //g(10,5,20,0)

g(10, 5, 20, 30); //g(10,5,20,30)

예제6-3 

예제6-4 

함수 중복 간소화 디폴트 매개 변수를 선언하면 함수 중복을 간소화할 수 있음

ex) class Circle {

Circle() { radius = 1; }

Circle(int r) { radius = r; }

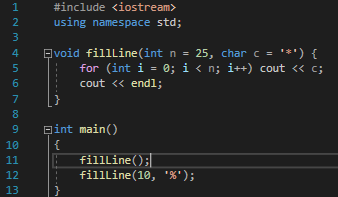
};

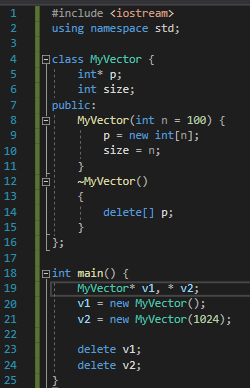
🡪

class Circle {

Circle(int r = 1) { radius = r; }

};

예제6-5 

예제6-6 

check time 1. 4

2. 2

3. int sum(int a, int b = 10) {

return a + b;

}

6.3 함수 중복의 모호성

중복된 함수 호출의 모호성 형변환으로 인한 모호성

참조 매개 변수로 인한 모호성

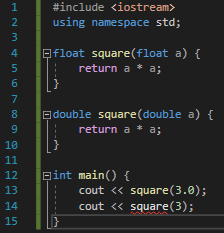
디폴트 매개 변수로 인한 모호성

형변환으로 인한 모호성 매개변수타입과 호출문의 실인자타입이 일치하지 않는 경우, 컴파일러는 자동으로 형변환 시도 ex) double square(double a);

square(3); //int -> double 형변환 컴파일러가 무엇으로 형변환할지 결정할 수 없을 때 컴파일 오류 발생 ex) double square(double a);

float square(float a);

square(3) //int->double? int->float?

예제6-7 

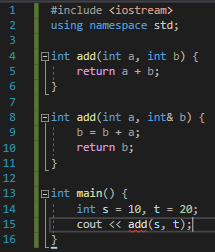
참조매개변수로 인한 모호성 중복된 함수 중에서 참조 매개 변수를 가진 함수가 있는 경우, 컴파일러는 중복된 함수 중 어느 것을 호출하는 지 알 수 없음

ex) int add(int a, int b);

int add(int a, int& b);

int s = 10, t = 10;

add(s, t); //컴파일 오류

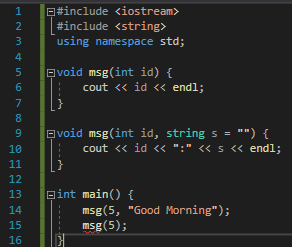
예제6-8 

디폴트 매개변수로 인한 모호성 디폴트 매개 변수가 가진 함수가 보통 매개변수를 가능 함수와 중복 작성될 때, 모호성이 존재할 수 있음

ex) void msg(int id);

void msg(int id, string s = "");

msg(6) //컴파일 오류, 함수호출 모호

예제6-9 

check time 1. 3

6.4 static 멤버

static의 특성 static 변수, 함수의 생명주기와 사용범위를 지정하는 방식 중 하나

생명주기 프로그램이 시작할 때 생성되고, 프로그램이 종료할 때 소멸

사용범위 변수나 함수가 선언된 범위 내에서 사용

non-static 멤버 각 객체마다 별도로 생성됨

인스턴스 멤버

static 멤버 클래스 당 하나만 생기고, 모든 객체들이 공유

클래스 멤버

static 멤버 선언 멤버 함수, 변수 앞에 static 지정자를 붙이면 됨

모든 멤버는 static으로 선언 가능

접근 지정자는 모두 가능

ex) class Person {

public:

int money;

void addMoney(int money) {

this->money = money;

}

static int sharedMoney; //sharedMoney는 Person객체의 개수와 상관없이 단 1개만 선언됨

static void addshared(int n) {

sharedMoney += n;

}

};

int Person::sharedMoney = 10; //static멤버변수는 외부에 전역변수로 선언되어야 함

static 멤버 변수는 외부에 전역 변수로 선언되어야 함 변수의 공간을 할당받는 선언문이 클래스 바깥의 전역공간에 선언되어야 함

static 멤버 사용: 객체의 멤버로 접근하는 방법 사용방법 객체.static멤버

객체포인터->static멤버

보통멤버와 동일하게 접근

static 멤버 사용: 클래스명과 범위지정 연산자(::)로 접근 사용방법 클래스명::static멤버

ex) Person::sharedMoney = 200;

Person::addshared(200);

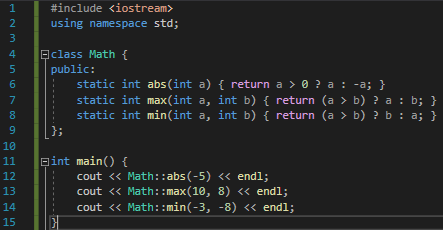
코드 사례 static 멤버들은 객체가 생기기 전에도 사용 가능함

::를 사용하면 객체를 언급하지 않고도 static멤버 활용가능

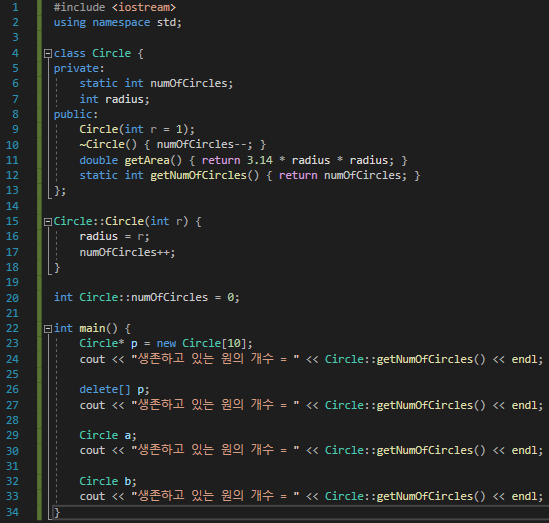
ex) Person::sharedMoney = 200;

cout << Person::sharedMoney << endl;

static의 활용 전역변수나 전역함수를 클래스에 캡슐화 전역함수, 전역변수를 statric멤버로 클래스에 캡슐화시킬 것

예제 6-10 

객체 사이의 공유 변수를 만들고자 할 때 클래스의 모든 인스턴스가 공유하는 변수∙함수를 만들고자 할 때 사용

예제6-11 

static 멤버 함수의 특징 static 멤버 함수는 오직 static멤버들만 접근 static멤버함수는 non-static멤버에 접근하는 것은 허용되지 않음

non-static멤버가 생성되기 전 static멤버함수가 실행될 수 있기 때문

but, non-static멤버함수는 static멤버변수에 접근가능

static 멤버 함수는 this 사용 불가 static 멤버 함수는 객체가 생기기 전부터 호출 가능 🡪 this 사용 불가

check time 1. 인스턴스, static, static

2. 4 🡪 static 멤버함수에서 non-static 멤버변수에 접근하고 있음

3. 3 🡪 non-statice 멤버함수가 ::를 통해 호출되고 있음