**객체지향프로그래밍 :5차시**

중요 문단은 \*표시 남김.

**객체 배열**

Int arr[3];

정수형 배열.

Circle circles[3];

Circle 타입의 배열 선언

객체 배열의 공간 할당 -> 배열의 각 원소 객체마다의 생성자 실행  
단, 매개 변수 없는 생성자만 호출할 수 있음.

Circle circleArr[3](5);

와 같은 형태로 선언하려 하면 오류 발생함.

**동적 메모리 할당 및 반환**

1. 정적 할당

변수 선언을 통해 메모리 할당  
배열 선언을 통해 많은 양의 메모리를 할당할 수 있음.

1. 동적 할당

실행 중에 힙 메모리에서 할당.

힙(heap) : 운영체제가 프로세스(프로그램)의 실행을 시작 시킬 때 동적 할당 공간으로 준 메모리 공간.

C의 경우 malloc(), free() 라이브러리 함수를 사용하여 동적할당 함.

C++의 경우 new/delete 연산자로 할당/반환함.

**new**

데이터타입 \*포인터변수 = new 데이터타입 ; // new

기본 타입 메모리, 배열, 객체, 객체 배열 등을 할당 가능.  
힙 메모리로부터 객체를 위한 메모리 할당을 요청하는 “객체의 동적 생성”을 함.  
객체 할당 시 생성자를 호출함.

**delete**

데이터타입 \*포인터변수 = new 데이터타입 ; // new

delete 포인터변수; // delete

new로 할당 받은 메모리를 반환함.  
소멸자 호출 뒤 객체를 힙에 반환하는 “객체의 동적 소멸”을 함.

적절치 못한 포인터로 delete하면 시간 오류가 발생함.

int n;

int \*p = &n;

delete p; // 실행 시간 오류

// 포인터 p가 가리키는 메모리는 동적으로 할당 받은 것이 아님

ㄴ할당에 실패했을 때.

int \*p = new int;

delete p; // 정상적인 메모리 반환

delete p; // 실행 시간 오류. 이미 반환한 메모리를 중복 반환할 수 없음

ㄴ동일한 메모리를 두 번 반환하려 시도했을 때.

Ex. (new/delete 예시)

int \*pInt = new int; // int 타입의 메모리 동적 할당

char \*pChar = new char; // char 타입의 메모리 동적 할당

Circle \*pCircle = new Circle(); // Circle 클래스 타입의 메모리 동적 할당

delete pInt; // 할당 받은 정수 공간 반환

delete pChar; // 할당 받은 문자 공간 반환

delete pCircle; // 할당 받은 객체 공간 반환

**동적 할당 메모리 초기화 및 delete시 유의사항**

기본 타입의 경우 동적 할당 시 초기화할 수 있지만, 배열은 불가능하다.

배열 delete 시 []를 생략하거나, 기본 타입 delete 시 []를 포함하면 비정상적인 반환이 된다.

**동적할당**

1. 기본 타입
2. int\* p = new int; 동적할당 함.
3. \*p = 5; 할당된 공간에 값을 집어넣음.
4. delete p; 포인터를 반환함.

텍스트, 스크린샷, 폰트, 번호이(가) 표시된 사진

자동 생성된 설명

1. 배열
2. int \*p = new int[5]; 배열 동적할당 함.
3. for( int i = 0; i < 5; i++ ) p[i] = i; 할당된 공간에 반복문 등으로 값을 집어넣음.
4. delete [] p; 배열 포인터를 반환함.

텍스트, 스크린샷, 폰트, 번호이(가) 표시된 사진

자동 생성된 설명

1. 객체
2. Circle \*p = new Circle; 객체 동적할당 함.
3. Circle \*q = new Circle(30); Circle( int r )을 실행해 값을 집어넣음.
4. delete p; 포인터 반환.  
   delete q; 포인터 반환.

텍스트, 스크린샷, 폰트, 도표이(가) 표시된 사진

자동 생성된 설명

1. 객체 배열
2. Circle \*pArr = new Circle[3]; 객체 배열 동적할당 함.
3. pArr[0].setRadius(10); setRadius( int rad )를 실행해 값(10)을 집어넣음.
4. delete [] pArr; 객체 배열 포인터를 반환함.

텍스트, 스크린샷, 폰트, 번호이(가) 표시된 사진

자동 생성된 설명

동적으로 생성된 배열도 보통의 배열처럼 사용할 수 있고, 포인터로 배열에 접근해야 한다.

pArray->setRadius(10);

. 으로 접근하는게 아니라, ->으로 접근해야 함.

배열 소멸 delete [] pArr; 시에는 생성의 반대 순으로 소멸됨.

**동적 메모리 할당과 메모리 누수**

텍스트, 스크린샷, 폰트, 도표이(가) 표시된 사진

자동 생성된 설명

텍스트, 도표, 폰트, 라인이(가) 표시된 사진

자동 생성된 설명

프로그램이 종료되면 운영체제가 누수 메모리를 모두 힙에 반환함.

**this**

스스로를 가리키는 객체 자신의 포인터.  
클래스의 멤버 함수 내에서만 사용 가능함.

개발자가 선언한 변수가 아니라, 컴파일러가 선언한 변수임.  
(컴파일러에 의해 멤버 함수에 묵시적으로 삽입 선언되는 매개변수임.)

class Circle {

int radius;

public:

Circle() { this->radius=1; }

Circle(int radius) { this->radius = radius; }

void setRadius(int radius) { this->radius = radius; }

....

};

매개변수의 이름과 멤버 변수의 이름이 같은 경우 사용됨.

Circle(int **radius**) { this->**radius** = **radius**; }

단, 멤버 함수가 아닌 함수에서 this를 사용할 수 없으며,   
(객체가 없으니 당연히 사용 불가)

Static 멤버 함수에서도 사용 불가함.  
(객체 생성 전에 static 함수 호출이 있을 수 있기 때문에.)

텍스트, 스크린샷, 폰트, 도표이(가) 표시된 사진

자동 생성된 설명

이러한 형태로 이루어짐.

**sring 클래스**

c++ 표준 라이브러리 <string> 헤더를 선언하여 사용할 수 있음.

* append : string변수.append(“추가할 문자열”)

다른 자료형과 같이, 입출력은 cout / cin으로 할 수 있음.

* stoi : string to int. 문자열을 정수형으로 형변환 시켜준다.  
  (2011 C++ 표준 버전부터 지원헀음.)

당연하게도 new/delete로 문자열을 동적 생성/반환 할 수 있음.