

Class

EUnS

January 11, 2020

OOP

class

operator

memory

Copy

1 OOP**2** class**3** operator**4** memory**5** Copy

OOP란?

- 객체 지향 프로그래밍(Object oriented programming)
- 사실 본인도 잘 모름
- 어떤 물체를 클래스 설정하고 애를 메소드(멤버 함수)로 가지고노는것.
- 큰 단위의 프로젝트의 협업을 위한것(큰 코드를 다룰때) : 추상화
- 그럼에도 클래스 내부를 이해하는것은 중요하다.

class

OOP

class

operator

memory

Copy

```
1  class name
2  {
3  public:
4      name();
5      ~name();
6      void function1();
7      void function2();
8      void function3();
9  private:
10     int memberVariable1;
11     char memberVariable2;
12     void function4();
13 };
14 void name::function1() { }
15
```

용어 정리

OOP

class

operator

memory

Copy

- 멤버 함수(member variable)
- 메소드(method) = 멤버 함수(member function)
- public : 외부에서 자유롭게 사용할수있는 것들
- private : class내 public에서 접근가능
- 객체 = 오브젝트(object) = 인스턴스(instance) : 데이터 할당된 class를 지칭
- 캡슐화(encapsulation) : 대충 내부구현을 숨긴다는 뜻
- 정보 은닉 : 불필요한 class 멤버 접근을 제한하는것 private으로 넣는다.
- 추상화 : 실제 구현을 감싸는것.
- protect : 상속 받은 자식에서도 접근가능
- 다형성 : 상속을 통한 다양함을 나타낸다는 OOP 특징
- 오버라이딩 : 같은 이름의 메소드를 상속 받는것

생성자 소멸자

- 초기화 : 멤버 이니셜라이저
- 반환값 X

```
1  class name
2  {
3  public :
4      name() : memberVariable1(0), memberVariable2('a')
5      {;}
6      ~name();
7  private :
8      int memberVariable1;
9      char memberVariable2;
10     static int n = 0 ;
11 };
12
```

C vs Cpp struct, class와 차이

OOP

class

operator

memory

Copy

- ① Cpp에서는 struct에 메소드 선언이 가능하다.
- ② C방식으로 struct 안붙이고 뒤에 이름만 사용가능
- ③ 기본접근자가 public임 class는 private
- ④ Coding standard : struct는 C 스타일로만 사용

operator

- ① class의 연산을 일일이 지정해줘야됨
- ② 참고 여기에서보고 필요한거 만들어쓰세요

```
1  class vector
2  {
3  public:
4      vector operator+(const vector& b);
5  private:
6      int x, y;
7  };
8
9  vector vector::operator+(const vector& b)
10 {
11     vector c;
12     c.x = b.x + x;
13     c.y = b.y + y;
14     return c;
15 }
16
```


class 메모리 구조

- ① 변수 선언순으로 올라감.
- ② 함수 .text 영역에
- ③ static 변수는 .BSS 영역
- ④ 바이트 패딩 일어남

Copy

OOP

class

operator

memory

Copy

```
1      class name
2      {
3      public:
4          name();
5          ~name();
6      private:
7          int memberVariable1;
8      };
9      name a();
10     name b = a;
```

Copy & move

- 클래스 생성시 몇가지 정의가 자동으로 생성된다.
- 기본생성되는것들
 - default constructor
 - default destructor
 - copy constructor
 - move constructor
 - copy assignment operator
 - move assignment operator
- = *default*, = *delete* 로 명시할수있다.

```
1      class name
2      {
3      public:
4      private:
5          int memberVariable1;
6      };
7
8      class name
9      {
10     public:
11         name() = default;
12         ~name() = default;
13         name(const name&) = default;
14         name& operator=(const name&) = default;
15         name(name&&) = default;
16         name& operator=(name&&) = default;
17     private:
18         int memberVariable1;
19     };
20
```

default copy와 move는 단순히 값을 복사하기만함.

```
1  class name
2  {
3  public:
4  name(): mString = nullptr;
5  {
6      mString = new char[20];
7      ...
8  }
9  private:
10     char* mString;
11 };
12 name a;
13 name b = a;
14
```

주소를 한번 찾아보자.

move (C++11)

- rvalue vs lvalue
- 너무 어려워요
- 필요하면 그때..

```
1      name a ,b;  
2      ...  
3      name c = a*b;  
4
```

과제

- 과제 1
- auto, using namespace std; 쓰지말것