**객체지향 설계 중간고사 참고문서**

1. **전체적인 구현 설명**

**아래는 슈도(pseudo)코드이므로, explicit, virtual등을 활용하여 구현하십시오.**

**Human**

* **이름 저장용 객체**
* **변수**
  + **Char \* name**
* **메소드**
  + **Human()**
  + **Human(char\* name)**
  + **Char\* get\_name()**
  + **Int get\_id()**

**Student : human**

* **human을 상속받은 학번을 저장하는 객체**
* **변수**
  + **Int student\_id**
* **메소드**
  + **student()**
  + **student (char\* name)**
  + **student(int student\_id)**
  + **student(char\* name, int student\_id)**
  + **int get\_student\_id()**
  + **Int get\_id()**

**smart\_obj**

* **메모리 할당을 안전하게 하기 위한 관리 클래스의 수퍼 클래스**
* **Pointer map을 통해 할당중인 메모리 관리**

**shared\_ref**

* **레퍼런스 카운팅을 통해 메모리 관리를 안전하게 하기 위한 클래스**

**weak\_pointer**

* **shared\_ref 의 값이 메모리 해제되었는지 pointer map을 통해 확인하고 해제되지 않은 값만 접근하는 클래스(pointer map에 해당 key가 존재하는지, 있다면 해당 value가 nullptr이 아닌 경우에만 값에 접근한다.)**

**test.cpp와 각 헤더파일, 주석을 참고하여 구현해주세요.**

**구현해야할 파일은 아래와 같습니다.**

**human.h human.cpp student.h student.cpp shared\_ref.cpp weak\_pointer.cpp**

1. **Copy 생성자와 = 오퍼레이터**

**A temp1 = new A();**

**A temp2 = temp1;**

**위와 같이 코드를 작성하면 copy 생성자가 호출됩니다.**

**-> A(A& temp)**

**A temp1 = new A();**

**A temp2;**

**temp2 = temp1;**

**위와 같이 코드를 작성하면 = 오퍼레이터가 실행됩니다.**

**-> A& operator=(const A &temp)**

1. **Make파일 예시**

**.SUFFIXES: .cpp .o**

**OBJECTS = a.o b.o**

**TEST\_FILE = test.o $(OBJECTS)**

**CXX = g++**

**TEST = test**

**$(TEST) : $(TEST\_FILE)**

**$(CXX) -o $(TEST) $(TEST\_FILE)**

**clean :**

**rm -rf $(TEST) $(TEST\_FILE)**

**test.o : test.cpp**

**a.o : a.cpp**

**b.o : b.cpp**

1. **헤더파일과 virtual, explicit 예시**

**class A {**

**public:**

**int a;**

**A();**

**explicit A(int a);**

**virtual int get\_a();**

**};**

**\*\*\*\* 최종 실행파일의 이름은 test로 작성해주세요!!!! \*\*\*\***

1. **Map 사용법**

**기본형태**

* **map<key,value> : key와 value를 pair 형태로 선언합니다.**

**iterator(반복자)**

* **begin() : beginning iterator를 반환**
* **end() : end iterator를 반환**

**추가 및 삭제**

* **insert( make\_pair(key,value) ) : 맵에 원소를 pair 형태로 추가**
* **erase(key) : 맵에서 key(키값)에 해당하는 원소 삭제**
* **clear() : 맵의 원소들 모두 삭제**

**조회**

* **find(key) : key(키값)에 해당하는 iterator를 반환**
* **count(key) : key(키값)에 해당하는 원소들(value들)의 개수를 반환**

**기타**

* **empty() : 맵이 비어있으면 true 아니면 false를 반환**
* **size() : 맵 원소들의 수를 반환**

**#include <iostream>**

**#include <map>**

**#include <string>**

**using namespace std;**

**int main(){**

**// <string, int> => <key, value>**

**map< string, int > m;**

**// insert(key,value)**

**m.insert(make\_pair("a", 1));**

**m.insert(make\_pair("b", 2));**

**m["c"] = 3; // also possible**

**// erase(key)**

**m.erase("b");**

**m.erase(m.find("c")); // also possible**

**// empty(), size()**

**if(!m.empty())**

**cout << "m size : " << m.size() << '\n';**

**// find(key)**

**cout << "a : " << m.find("a")->second << '\n';**

**// count(key)**

**cout << "a count : " << m.count("a") << '\n';**

**// map< string, int >::iterator it; also possible**

**for(auto it = m.begin(); it != m.end(); it++)**

**cout<<"key : "<<it->first<<" "<<"value : "<<it->second <<endl;**

**return 0;**

**}**