클래스

(Class)

들어가기 전에

- 오늘의 소스코드를 받아봅시다.
  - 중요한 코드는 꼭 다른 곳으로 복사해두세요.
  - phycom2015-2 디렉토리에서,
    - git stash
    - git pull
  - 디렉토리가 사라졌다면,
    - git clone https://github.com/geonmo/phycom2015-2.git

지금까지의 내용

- 변수란 무엇인가?
  - 변수와 상수
- 제어문이란?
  - 조건문과 반복문
- 함수란?
  - 한 로직을 처리하기 위한 프로그램 코드의 묶음
- 자료구조란?
  - 데이터를 저장/관리하는 가장 효율적인 방법들
- 포인터란?
  - 변수 "자체"를 가리키는 변수

Class란?

- 클래스는 객체지향 프로그램을 구현하기 위한 문법으로,
  - 1) 멤버변수(속성,Attribute)

  과
  - 2) 멤버함수(메소드,Method)

  로 이루어져 있다.

Class를 쓰는 이유

- 구조화된 자료묶음을 사용하고 싶을 때
  - 멤버변수와 관련된 내용
    - 예) 학생인 철수와 영희의 ID, Score, Address를 저장하고 싶을 때
      - Class를 안 쓸 경우
        - » cheolsu_id, cheolsu_score, cheolsu_address;
        - » younghee_id, younghee_score, younghee_address;
      - Class를 쓸 때
        - » Set up “student.id, student.score, student.address”;
        - » student cheolsu, younghee;

Class를 쓰는 이유

- 멤버변수의 접근을 특정한 경우로 제한하고 싶을 때
  - 멤버함수와 관련된 내용
    - 예) 철수의 ID를 입력하는데 숫자가 아니라 한글이 입력된다면?
      - 예외처리를 위하여
        1) 변수를 입력하는 set함수 와
        2) 변수의 값을 받아오는 get함수를 이용하여 제어 가능
    - 적절한 정책 설정으로 멤버함수 이외의 방법으로 멤버변수의 접근을 차단 가능!

Class의 구조

- public? private?
  - public은 외부에서 해당 키워드의 멤버에 접근이 가능하다는 키워드
  - private는 같이 클래스 안에 포함된 public 멤버에 의해서만 사용 가능하다는 키워드
- 생성자? 소멸자?
  - 클래스로부터 객체가 만들어질 때(생성), 그 객체에 포함된 멤버들의 값을 초기화(최초 정의)할 때 사용됨
    - 모든 객체는 생성자(+복사생성자) 호출 후 사용가능
  - 객체가 소멸될 때 소멸자 함수를 동작시켜 객체를 제거
  - 생성자와 소멸자는 반환값이 없고, 해당 클래스와 같은 이름을 사용해야 인식함.

Class의 구조

1. 클래스의 이름
2. 클래스 영역지정
3. 멤버변수 선언
4. 생성자, 소멸자 선언
5. 멤버함수 선언
6. 멤버변수, 생성자, 멤버함수 정의

student.cpp

```
class Student {
private:
  int id_;
  std::string name_;
  std::string address_;
  int score_;
public :
  Student() {
    id_ = 0;
    name_ = std::string("");
    address_ = std::string("");
    score_ = 0;
  }
  void setID(int id) { id_ = id;}
  void setID(std::string str) { std::cout<<"에
러!"<<std::endl; }
  void setAddress(std::string);
};
void Student::setAddress(std::string address) {
  address_ = address;
}
```

함수 오버로딩(동명 구현)

- setID(int id) 와 setID(std::string str) 은 완전히 다른 함수
  - 이름이 같아도 argument가 다르면 다른 함수로 인식
  - 적절하게 사용하면 이름 짓는 고통을 받지 않아도 됨

student.cpp

```
class Student {
private:
  int id_;
  std::string name_;
  std::string address_;
  int score_;
public :
  Student() {
    id_ = 0;
    name_ = std::string("");
    address_ = std::string("");
    score_ = 0;
  }
  void setID(int id) { id_ = id;}
  void setID(std::string str) { std::cout<<"에
러!"<<std::endl; }
  void setAddress(std::string);
};
```

코딩 기법

- Tip
  - 멤버변수는 private, 멤버함수는 public
  - 멤버변수의 값이 의미를 가지고 있다면 반드시 멤버함수로 제어할 것
  - 효율적인 코딩을 위한 코딩 방식
    - 클래스의 첫 글자는 반드시 대문자
      - class Student {};
    - 멤버함수가 get, set으로 시작한다면 소문자, 다음부분에는 대문자를 섞음
      - getID(); , getAddress();, setAddress(std::string)
    - 멤버변수는 소문자로, (option) 변수 끝에 _를 붙여주면 좋다
      - id_, address_, score_

소스코드 설명

- info.cpp(main)
  + student.cpp(student class)
  - 학생정보 클래스 제작(header/code 분리)

- read.cpp
  - 파일 입/출력 시스템 설명
    - ifstream, ofstream 등

Quiz

- 위에서 배운 내용으로 다음 프로그램을 작성해보자.
  - 프로그램 : 클래스를 이용한 POS시스템 개발
    - 명세
      - commodity.txt 파일을 읽어 상품목록을 임의의 자료구조(본인생각에 좋은 것으로)로 저장
      - 다음 명령을 지원해야 함
        - » 구매 요청 목록에 등록
          - 물품 번호로 입력 받을 것
          - 등록 시 상품가격의 합산 제공(화면 출력)
        - » 구매 요청 목록으로 결제
          - 결제 시 거스름돈 계산 (받은 금액은 cin으로 받을 것)
        - » 구매 완료 시 목록 비우기
        - » 옵션
          - 1. 프로그램 종료 시, 그때까지 판매된 모든 물품을 sold.txt파일로 저장(그때그때 저장해도 상관 없음.)
          - 2. 구매 취소기능은 옵션으로..
    - 제한조건: class Pos, class Commodity는 필수

실행예제

- 물품 구매

```
[geonmo@gate2 solution]$ ./Pos
Request list : 0
총 구매금액 : 0
Total : 10
ID : 880000 Name : 빵 Price : 500
ID : 880001 Name : 도넛 Price : 3000
ID : 880002 Name : 케익 Price : 12000
ID : 880003 Name : 생수 Price : 500
ID : 880010 Name : 삼단도시락 Price : 4500
ID : 880011 Name : 이단도시락 Price : 3000
ID : 880012 Name : 특도시락 Price : 6000
ID : 880020 Name : 젓가락 Price : 1400
ID : 880021 Name : 숟가락 Price : 1000
ID : 880022 Name : 빨래집게(묶음) Price : 1000
1번 구매, 0번 프로그램 종료
```

실행예제

- 물품 구매

```
구매하고 싶은 물품의 ID는? 880002
구매 수량은? 7
Request list : 1
7 x ID : 880002 Name : 케익 Price : 12000
총 구매금액 : 84000
Total : 10
ID : 880000 Name : 빵 Price : 500
ID : 880001 Name : 도넛 Price : 3000
ID : 880002 Name : 케익 Price : 12000
ID : 880003 Name : 생수 Price : 500
ID : 880010 Name : 삼단도시락 Price : 4500
ID : 880011 Name : 이단도시락 Price : 3000
ID : 880012 Name : 특도시락 Price : 6000
ID : 880020 Name : 젓가락 Price : 1400
ID : 880021 Name : 숟가락 Price : 1000
ID : 880022 Name : 빨래집게(묶음) Price : 1000
1번 구매, 0번 프로그램 종료
```

실행예제

- 물품 구매

```
구매하고 싶은 물품의 ID는? 880002
구매 수량은? 2
Request list : 1
9 x ID : 880002 Name : 케익 Price : 12000
총 구매금액 : 108000
Total : 10
ID : 880000 Name : 빵 Price : 500
ID : 880001 Name : 도넛 Price : 3000
ID : 880002 Name : 케익 Price : 12000
ID : 880003 Name : 생수 Price : 500
ID : 880010 Name : 삼단도시락 Price : 4500
ID : 880011 Name : 이단도시락 Price : 3000
ID : 880012 Name : 특도시락 Price : 6000
ID : 880020 Name : 젓가락 Price : 1400
ID : 880021 Name : 숟가락 Price : 1000
ID : 880022 Name : 빨래집게(묶음) Price : 1000
1번 구매, 0번 프로그램 종료
```

TDD

- 힌트1.
  - Commodity 클래스부터 작성 후 테스트코드를 작성해봅니다.
    - 파일로부터 Commodity class 객체를 생성하고 이를 컨테이너에 담아봅시다.