Design Pattern(C++)

Project

❑ 과제 배점

40점 만점

[참고] Quiz : 30점, Final Exam : 30점, Project : 40점

❑ 과제 설명

총 3개의 프로그래밍 과제 입니다. 각 과제별로 단계별 부분 점수가 있습니다.

❑ 제출 방법

"영문이름_사번" 으로 폴더를 만들고, 폴더안에

과제설명.txt 을 만들고

(1) 몇 번 과제를 해결했는지 적어 주세요..

(2) 어떤 컴파일러를 사용했는지를 적어 주세요

해결한 과제만, "**project1-1.cpp, project1-2.cpp, project2.cpp** …" 으로 소스를 만들어 주세요. 실행파일은 필요 없습니다. 소스만 포함해 주시면 됩니다. **소스 파일의 이름 규칙을 꼭 지켜 주세요(대소문자 포함)**

폴더를 포함해서 통째로 압축해서 LGE MOOC "Final Project"에서 "select file" 버튼을 눌러서 압축파일을 올려 주세요

주의!!, 반드시 압축 파일의 이름은 "영문" 이어야 합니다.(한글은 시스템상 다운로드 불가로 채점을 할 수 없습니다.)

❑ 제출 기한

과정 종료일 까지(Syllabus 참고)

과제 1. 숫자 야구 게임 만들기.

아래 코드는 사용자가 생각한 숫자를 컴퓨터가 맞추는 게임입니다.

1. 컴퓨터가 정수 3개를 사용자에게 보여주고
2. 사용자로부터 strike, ball의 개수를 입력 받은 후 3strike 라면 종료 됩니다.
3. 사용자로부터 입력된 결과를 분석해서 다시 3개의 정수를 구한 후 1부터 반복합니다.

참고로, 현재 컴퓨터가 사용하는 알고리즘은 무조건 중복되지 않은 난수 3개를 보여 줍니다.

```
#include <iostream>
#include <ctime>
#include <cstdlib>
#include <vector>
#include <tuple>
using namespace std;

class BaseBallGame
{
    typedef tuple<int, int, int> INPUT;
    typedef tuple<int, int> RESULT;
    vector<pair<INPUT, RESULT> > v;
public:
    BaseBallGame() { srand((unsigned)time(0)); }

    void run()
    {
        while (1)
        {
            //-----------------------------------------------
            // 이전의 결과가 담긴 vector v를 참고 해서
            // 사용자가 생각한 숫자를 예측해 냅니다.
            // 현재 구현은 무조건 랜덤 입니다.
            int x = 0, y = 0, z = 0;
            do {
                x = rand() % 9 + 1;
                y = rand() % 9 + 1;
                z = rand() % 9 + 1;
            } while (x == y || y == z || x == z);
            //-----------------------------------------------
            cout << "당신이 생각한 숫자는 " << x
                << ", " << y << ", " << z << " 입니까 ?" << endl;
            int strike = 0, ball = 0;
            cout << "strike 갯수 : ";
```

```
            cin >> strike;
            if (strike == 3)
            {
                cout << "성공 !" << endl;
                break;
            }
            cout << "ball 갯수 : ";
            cin >> ball;
            //--------------------------------------
            // 입력된 결과(strike, ball)을 기록해 두었다가
            // 다음수를 예측할때 사용합니다.
            v.push_back(make_pair(INPUT(x, y, z), RESULT(strike, ball)));
            dump();
        }
    }
    // 필요하신 분을 위해 참고용으로 만든 함수입니다.
    void dump()
    {
        printf("-------------------------------\n");
        printf("입력값   s b\n");
        for (auto& p  : v) // p는 pair<INPUT, RESULT> 입니다.
        {
            printf(" %d %d %d : %d %d\n",
                get<0>(p.first), get<1>(p.first), get<2>(p.first),
                get<0>(p.second), get<1>(p.second));
        }
        printf("-------------------------------\n");
    }
};
int main(void)
{
    BaseBallGame bbg;
    bbg.run();
}
```

사용자가 생각한 숫자를 맞추는 알고리즘은 다양하게 변경 할 수 있습니다. 또는, 초급용/중급용/고급용 알고리즘을 교체하면서 사용할 수도 있습니다. 하지만, 위 코드는 하나의 함수안에서 모든 것을 처리하고 있기 때문에 알고리즘을 교체 하려면 BaseBallGame 클래스 자체를 수정해야 합니다.

과제는 다음과 같습니다.

과제 내용

위 코드에서 변하는 것에 해당 하는 “이전 결과를 바탕으로 사용자가 생각한 숫자를 예측하는 부분”을 교체 가능한 설계로 변경해 보세요. 즉, BaseBallGame 클래스가 변하지 않고도 알고리즘을 교체 할 수 있도록 설계 구조를 변경하세요

아래의 3가지 형태로 전부 작성해 주세요

1. 변하는 부분을 가상함수로(template method) 해서 구현해 보세요

2. 변하는 것을 다른 클래스로 분리한 후, 인터페이스 기반으로 교체 가능한 설계(strategy)로 만들어 보세요

3. 변하는 것을 다른 클래스로 분리한 후, 템플릿의 인자로 교체 가능한 설계(PolicyBase)로 만들어 보세요.

평가 규칙.

본 과정은 알고리즘 과정이 아니므로 알고리즘 자체는 평가 대상이 아닙니다. 프로그램의 구조만 평가 합니다.

앞의 예문에서는 참고용으로 tuple, pair 등을 사용했는데, 잘 모르시는 분은 다른 방식으로 하셔도 전혀 문제 없습니다.

과제 2. Composite 패턴으로 File, Folder 만들기

아래의 main 함수 코드가 실행 될 수 있도록 File 및 Folder 클래스를 만들어 주세요

```
int main()
{
    // 조건 1. Folder와 File의 객체를 생성할 수 있어야 합니다.
    Folder* rootFolder = new Folder("ROOT"); // 폴더는 이름만 있습니다.
    Folder* aaaaFolder = new Folder("AAAA");
    Folder* bbbbFolder = new Folder("BBBB");

    File* file1 = new File("a.txt", 10); // 화일은 이름과 크기가 있습니다
    File* file2 = new File("b.txt", 20);
    File* file3 = new File("c.txt", 30);
    File* file4 = new File("d.txt", 40);


    // 조건 2. 폴더안에 파일 및 다른 폴더를 넣을 수 있어야 합니다.
    rootFolder->add(aaaaFolder);
    rootFolder->add(bbbbFolder);
    rootFolder->add(file1);

    aaaaFolder->add(file2);
    aaaaFolder->add(file3);

    bbbbFolder->add(file4);


    // 조건 3. 파일과 폴더 크기를 출력할 수 있어야 합니다.
    //         폴더는 자신만의 크기는 없지만 크기를 구할 수 는 있습니다.
    cout << file1->getSize()      << endl; // 10
    cout << aaaaFolder->getSize() << endl; // 50
    cout << rootFolder->getSize() << endl; // 100


    // 조건 4. 화면 출력
    file1->print();         // 파일이므로 이름과 크기만 출력해 주세요.
                            // ex) (a.txt, 10)

    rootFolder->print();    // ROOT폴더 전체의 모양을 보기좋게 출력해 주세요
                            // [ROOT]
                            //          [AAAA]
                            //              (b.txt, 20)
                            //              (b.txt, 30)
```

```
                            //          [BBBB]
                            //              (d.txt, 40)
                            //          (a.txt, 10)


    // 조건 5. 폴더 제거시 폴더 안에 있는 모든 파일과 폴더가 제거 되게 해주세요
    delete rootFolder;
}
```

과제 3. Queue 어답터 만들기

Queue가 필요합니다.

```
#include <iostream>
#include <list>
using namespace std;
int main()
{
    Queue<int> q;
    q.Push(10);
    q.Push(20);
    cout << s.Pop() << endl; // 10
    cout << s.Pop() << endl; // 20
}
```

Queue 를 완전히 새로 만들 필요 없이 STL 의 list 를 사용하면 간단하게 만들 수 있습니다.

STL 의 list 를 사용해서 위 코드에서 사용할 수 있는 Queue Adapter 를 만들어 보세요

1. 클래스 어답터로 만들어 보세요
2. 객체 어답터로 만들어 보세요.