1 다음중 틀린 것은 ?

1 protected 생성자를 사용하면 자신과 자신의 파생클래스 모두 객체를 생성할수 없다.
2 protected 소멸자를 사용하면 외부에서 객체를 파괴 할수 없다.
3 protected 소멸자를 사용하면 객체를 스택에 만들수 없다.
4 protected 멤버는 외부 함수에서는 접근할수 없지만 파생 클래스 에서는 접근할수 있다.

2 다음 설명중 틀린것은 ?

1 기반 클래스 포인터로 파생 클래스를 가리킬수 있다.
2 기반 클래스 포인터로 override된 멤버 함수 호출시 가상함수가 아닌경우 기반 클래스 멤버 함수가 호출된다.
3 기반 클래스 포인터로 파생 클래스의 고유한 멤버 데이타에 접근 할수 있다.
4 하나의 컨테이너에 A와 B를 동시에 저장하려면 공통의 기반 클래스가 있어야 한다.

3 Shape와 Rect가 위 코드와 같을때 다음중 Error는 ?

```
structShape
{
    virtual void Draw() = 0;
};
class Rect : public Shape
{
    virtual void Draw() {}
};
```

1 Shape s;
2 Rect r;
3 Shape* p1;
4 Rect* p2;

4 다음 설명중 틀린것은 ?

1 tightly coupling 보다는 loosely coupling이 확장성과 유연성이 좋다.
2 interface는 클래스 사이에 지켜야 하는 규칙을 담고 있다.
3 기반 클래스의 소멸자는 가상함수가 되어야 한다.
4 포함(구성, composition) 보다는 상속(inheritance)가 유연성이 좋다.

5 기능이 확장되어도 코드는 수정되지 않도록 디자인 해야 한다는 원칙은 무엇인가요 ? 대문자 약자로 적어 주세요

6 다음 설명중 잘못된 것은 ?

1 변하지 않은 코드안에 있는 변하는 부분은 분리되는 것이 좋다.
2 변하는 것을 가상함수로 분리하면 변하는 것의 실행시간 교체가 가능하다.
3 변하는 것을 다른 클래스로 분리하면 변하는 것(정책)을 재사용할수 있게 된다.
4 변하는 것을 담은 정책 클래스를 템플릿 인자로 교체하면 인라인 치환도 가능하다.

7 **다음 설명이 나타내는 패턴의 이름은 무엇인가요 ?**

다양한 알고리즘이 존재 하면 이들 각각을 하나의 클래스로 캡슐화하여 알고리즘의 대체가 가능하도록 한다. 이 패턴을 이용하면 클라이언트와 독립적으로 다양한 알고리즘으로 변형 할 수 있다. 알고리즘을 바꾸더라도 클라이언트는 아무런 변경을 할 필요가 없다."

8 **다음중 state pattern 에 대한 설명으로 잘못된 것은 ?**

1 객체 자신의 내부 상태에 따라 행위를 변경하도록 한다.

2 상태에 따른 행위를 국지화 하여 서로 다른 상태에 대한 행위를 별도의 객체로 관리 한다.

3 상태 전이 규칙을 명확하게 만든다.

4 상태 객체는 공유 될 수 없다.

9 **다음중 Policy Base Design 에 대한 설명으로 잘못된 것은 ?**

1 단위 전략을 담은 클래스를 템플릿 인자로 전달하는 하는 디자인 기법이다.

2 전략을 담은 코드가 인라인 치환이 될수도 있다.

3 단위 전략이 지켜야 하는 규칙은 인터페이스로 설계 된다.

4 실행시간에 단위 전략을 교체 할 수 없다.

10 **다음 설명이 나타내는 디자인 패턴은 무엇인가요 ?**

"클래스의 인터페이스를 클라이언트가 기대하는 형태의 인터페이스로 변환 한다. 이 패턴은 서로 일치 하지 않은 인터페이스를 갖는 클래스들을 함께 동작 시킨다."

11 **다음 설명이 나타내는 패턴은 ?**

"복잡한 서브 시스템에 대한 단순한 인터페이스 제공이 필요할 때 유용하다. 클라이언트와 구현 클래스 간에 너무 많은 종속성이 존재 할 때 이 패턴을 사용하면 클라이언트와 다른 서브 시스템간 의 결합도를 줄일 수 있다."

12 **다음중 GoF's 디자인 패턴에서 구조 패턴이 아닌 것은 ?**

1 Proxy

2 Bridge

3 Prototype

4 Composite

13 **다음 설명중 맞는 것은 ?**

1 상속을 사용하는 실행시간에 객체에 기능을 추가 할 수 있다

2 template method 패턴과 factory method 패턴은 유사한 코드 형태를 나타내게 된다.

3 composite 패턴은 클래스 계층에 새로운 가상함수를 쉽게 추가 할수 있게 한다.

4 bridge 패턴을 사용하면 하위 시스템의 복잡함을 단순한 인터페이스로 제공할수 있다.

14 adapter 패턴에 대한 설명으로 잘못된 것은 ?

1 인터페이스의 불일치를 해결한다.
2 객체 어답터와 클래스 어답터가 있다.
3 STL stack 컨테이너는 객체 어답터 이다.
4 클래스 어답터는 상속을 사용하는 경우가 많다.

15 다음 패턴중 GoF's 의 디자인 패턴에 의한 분류에서 나머지 셋과 다른 패턴은 ?

1 singleton
2 builder
3 state
4 factory method

16 다음중 iterator 패턴에 대한 설명으로 잘못된 것은 ?

1 객체 내부의 표현 방식을 모르고도 집합 객체의 각 요소들을 순회 할 수 있다
2 서로 다른 집합 객체 구조에 대해서도 동일한 방법으로 순회 할 수 있다
3 STL 의 iterator는 가상 함수 기반이 아니다.
4 Broadcast 방식의 교류를 가능하게 한다.

17 아래의 설명이 나타내는 패턴은 무엇인가요 ? 영어로 소문자로 적어 주세요

"객체 사이의 1:N의 종속성을 정의 하고 한 객체의 상태가 변하면 종속된 다른 객체에 통보가 가고 자동으로 수정이 일어 나게 한다."

18 다음 보기는 어떤 패턴에 대한 설명입니다. 나머지 셋과 다른 패턴은 무엇인가요 ?

1 상세 화된 서브 클래스를 정의 하지 않고도 서로 관련성이 있거나 독립적인 여러 객체의 군을 생성하기 위한 인터페이스를 제공한다.
2 추상화 개념과 이에 대한 구현간의 종속성을 없애고 싶을 때
3 클라이언트로부터 구현을 완벽하게 은닉하길 원한다
4 클래스 계층도 에서 클래스의 수가 급증하는 것을 방지 하고자 할 때

19 다음 설명이 나타내는 패턴은 무엇인가요 ? 영어 단어 소문자 한자로 작성해 주세요

새로운 오퍼레이션을 쉽게 추가 한다. 새로운 ConcreteElement를 추가하기 어렵다.캡슐화 전략을 위배할 수 있다.