TCT-기술인증테스트
시스템&솔루션개발
[2018년 #차]

사번 : _____

성명 : _____

1. 소프트웨어의 예비 설계(preliminary design)라고도 하는 상위 설계에 대한 설명이 아닌 것을 고르시오.
1) 아키텍처 설계 : 시스템의 전체적인 구조를 나타낸다.
2) 데이터 설계 : 시스템에 필요한 정보를 자료구조와 데이터베이스 설계에 반영한다.
3) 시스템 분할 : 전체 시스템을 여러 개의 서브시스템으로 분할한다.
4) 알고리즘 정의 : 각 모듈의 실제적인 내부를 알고리즘(pseudo-code) 형태로 표현한다.

(정답) 4
(해설) 각 모듈의 실제적인 내부에 대한 알고리즘은 하위설계에서 다룬다

2. 소프트웨어 아키텍처의 UML(Unified Modeling Language)의 4+1 관점 중 "시스템의 기능을 제공하기 위해 필요한 클래스나 컴포넌트의 종류와 이들의 관계를 파악하고 기술하는 것"을 나타내는 관점을 고르시오.

1) 유스케이스 관점(usecase view)
2) 논리적 관점 (logical view 또는 design view)
3) 구현 관점(implementation view)
4) 배치 관점(deployment view)

(정답) 2
(해설) 논리적 관점은 시스템의 기능을 제공하기 위해 필요한 클래스나 컴포넌트의 종류와 이들의 관계를 파악하고 기술하는 것을 다룬다

3. 단계적 개발 모델은 릴리스를 구성하는 방법에 따라 점증적 개발(incremental development) 방법과 반복적 개발(iterative development) 방법으로 나눌 수 있다. 다음 중 점증적 개발 방법의 특성이 아닌 것을 고르시오.

1) 대학 정보 시스템 개발시, 교무/학사 관련 시스템을 먼저 개발하여 사용하고, 다른 부서의 업무 시스템을 차츰 개발
2) 한꺼번에 많은 비용을 들이지 않아도 된다
3) 서브시스템들이 서로 관련이 있어 처음 설계할 때부터 이후에 개발할 다른 서브시스템과의 연관성을 고려해야 한다
4) 초기에 시스템 전체를 일차적으로 개발하여 인도한 후, 각 서브시스템의 기능과 성능을 변경 및 보강하여 완성도를 높인다

(정답) 4
(해설) 반복적 개발 : 초기에 시스템 전체를 일차적으로 개발하여 인도한 후, 각 서브시스템의 기능과 성능을 변경 및 보강하여 완성도를 높인다

4. 다음 모듈 설계의 원칙에서 빈칸에 알맞는 단어를 바르게 짝지은 것을 고르시오.

- '모듈 간'의 (ㄱ)은 (ㄴ) 한다.
- '모듈 내 구성 요소들 간'의 (ㄷ)은 (ㄹ) 한다.

1) (ㄱ) 응집(cohesion), (ㄴ) 느슨하게(loosely), (ㄷ) 결합(coupling), (ㄹ) 강하게(strongly)
2) (ㄱ) 응집(cohesion), (ㄴ) 강하게(strongly), (ㄷ) 결합(coupling), (ㄹ) 느슨하게(loosely)
3) (ㄱ) 결합(coupling), (ㄴ) 느슨하게(loosely), (ㄷ) 응집(cohesion), (ㄹ) 강하게(strongly)
4) (ㄱ) 결합(coupling), (ㄴ) 강하게(strongly), (ㄷ) 응집(cohesion), (ㄹ) 느슨하게(loosely)

(정답) 3
(해설) • 모듈 간의 결합(coupling)은 느슨하게(loosely) 한다.
• 모듈 내 구성 요소들 간의 응집(cohesion)은 강하게(strongly) 한다.

5. 큰 문제를 작은 단위로 쪼개어 하나씩 개발해나가는 것을 모듈화라고 할 때, 다음 중 모듈화의 특성중 가장 적절한 것을 고르시오.

1) 분할과 정복(divide and conquer)의 원리가 적용되어 복잡도가 증가할 수 있다.
2) 변경하기 어렵고, 변경으로 인한 영향이 커질 수 있다.
3) 프로그램의 효율적 관리가 어렵다.
4) 모듈 개수가 너무 많아지면 모듈 간 통합 비용이 증가할 수 있다.

(정답) 4
(해설) 2) 모듈의 크기가 작아지면 모듈 간의 통신 횟수가 많아져 복잡해질 수 있다.

6. 객체지향 방식에서 캡슐화를 통해 얻을 수 있는 장점이 아닌 것을 고르시오.

1) 캡슐화(데이터+메서드)로 인해 객체 사이의 독립성이 구조적으로 보장된다.
2) 메서드의 기능만 알면 객체를 사용할 수 있다.
3) 데이터와 메서드의 오버로딩을 피하고 기존 클래스에 있는 것을 재사용할 수 있다.
4) 객체 제공자와 객체 이용자(외부 객체)의 분담을 명확히 할 수 있다.

(정답) 3
(해설) 데이터와 메서드의 오버로딩을 피하고 기존 클래스에 있는 것을 재사용할 수 있다. ==> 상속의 장점

7. 다음 Thread 에 대한 설명 중 가장 적절하지 않은 것을 고르시오.

1) 수행 시간이 긴 작업을 수행하는 동안 사용자 입력 처리 등이 가능하다.
2) Thread 를 위한 공유 메모리/자원을 별도로 할당해야 하므로 Multi Process 보다 비효율적이다.
3) Thread 는 Data, Heap 영역을 공유하기 때문에 Thread 간 정보 전달이 용이하다
4) 코드의 흐름을 이해하기 어려우며, 문제 발생 시 디버깅이 난해하다.

(정답) 2
(해설) 자신이 속한 Process 의 자원을 공유하므로, Multi Thread 가 더 효율적이다.