'2001년 1학기 S/W 공학 기말고사 문제지

아래의 보기를 참조하여 적절한 용어를 선택하시오. (배점: 각 2점)

♥ 소프트웨어 설계 모델에는 네 가지가 있는데, 먼저 (1)_____ 은 분석단계에서의 정보영역 모델을 데이터구조로 변환시킨 것이고, (2)_____은 구조적 요소들을 소프트웨어 컴포넌트의 절차적 표현으로 변환시킨 것이고, (3)_____은 소프트웨어 모듈간, 또는 다른 시스템간, 또는 인간과의 통신 방법을 서술한 것이다. 마지막으로, (4)_____는 소프트웨어 구성요소간의 구조적 관계를 정의한 것이다.

♥ (5)_____과 (6)_____은 서로 보완적인 설계 개념으로, 전자는 설계자로 하여금 low-level의 구체적 사항에 대한 설명없이도 데이터와 프로시저를 명세할 수 있도록 해주며, 후자는 설계의 진행과 함께 low-level의 구체적인 사항들이 하나씩 드러나도록 지원해 준다.

♥ (7)_____ 소프트웨어는 전체 프로그램을 하나의 모듈로 구성시키는 방법으로, divide&conquer 개념에 의해, 작은 모듈들로 구성시키는 (8)_____ 소프트웨어에 비하여 복잡도와 비용 측면에서 비효율적이다.

♥ Class hierarchy (또는 program structure)에서 (9)_____은 다른 모듈에 의해 직접적으로 제어받는 모듈의 개수를 말하고, (10)_____은 몇 개의 모듈이 해당 모듈을 제어하는가를 말한다. 이 경우, 효과적인 설계를 위해서는 낮은 (11)_____과 높은 (12)_____이 바람직하다.

♥ 모듈의 독립성은 두 가지 기준으로 평가할 수 있는데, (13)_____은 모듈의 기능적 강도를 나타내고, (14)_____은 모듈 상호간의 독립성의 정도를 나타낸다. 따라서, 효과적인 설계를 위해서는 (15)_____을 낮추고, (16)_____을 강화시켜야 한다.

♥ 소프트웨어 아키텍처 설계시, (17)_____ mapping은 정보전달과정을 incoming flow, transform center, 그리고 outgoing flow 등으로 나누어 처리하고, (18)_____ mapping은 transaction center를 중심으로 수신경로와 여러 개의 액션경로로 나누어 처리한다.

♥ 인터페이스 설계 모델에는 네 가지가 있는데, 먼저 소프트웨어 엔지니어는 (19)_____을 만들고, 인간 엔지니어는 (20)_____을 만들고, 최종사용자는 (21)_____라고 불리는 이미지를 머릿속에 그리게 되며, 마지막으로 시스템 구현자는 (22)_____를 갖게 된다.

♥ (23)_____은 체계적인 방법으로 에러의 증상을 찾는 것이고, (24)_____은 에러의 증상으로부터 원인을 찾는 과정을 말한다.

♥ 테스팅 기법은 크게 두 가지로 나눌 수 있는데, (25)_____은 명시된 기능들에 대한 테스트를 말하고, (26)_____은 내부적 작동과정에 대한 테스트를 말한다. 따라서, (27)_____은 주로 테스팅 초기에 수행되고, (28)_____은 테스팅 후반부에 적용된다.

♥ 단위 테스팅 및 통합 테스팅시 해당 모듈의 상위 제어 모듈들을 대신할 (29)_____ 소프트웨어와 하위의 모듈들을 대체할 (30)_____ 소프트웨어가 필요하다. 한편, (31)_____ 소프트웨어는 (32)_____ 소프트웨어보다 일반적으로 만들기가 힘든데, 이러한 단점을 없앤 통합 테스팅이 (33)_____ 통합 테스팅 방법이다.

♥ 디버깅 방법 중 에러의 원인에 대한 가설들을 세워 이를 하나씩 확인해 나가는 방법을 (34)_____이라 한다.

♥ 객체지향 개념에서 클래스는 자신을 특성화시키는 (35)_____와 이것을 처리할 알고리즘인 (36)_____으로 구성되며, 이들은 하위클래스로 (37)_____된다. 또한 클래스의 이름하에 (38)_____되어져서 정보은폐 및 재사용성을 달성할 수 있다. 한편 (39)_____는 객체간의 상호작용 수단을 말한다.

♥ 객체지향분석에서 사용하는 CRC 색인 카드는 (40)_____, (41)_____, (42)_____의 약자를 말하며, 이를 주로 활용하는 객체지향 설계를 (43)_____ 설계라 하는데, 이것은 각 객체별 속성값에 대한 데이터 구조와 연산들에 대한 알고리즘 설계를 일컫는다.

♥ 객체지향 테스팅에서는 구조적 테스팅과는 달리, 단위 테스팅 대신에 (44)_____ 테스팅을 사용하며, 또한 통합 테스팅 대신에, 하나의 이벤트 또는 입력에 관계되는 클래스들을 통합하는 (45)_____ 테스팅과 같은 방법이 사용된다.

♥ 현존하는 프로그램으로부터 데이터, 아키텍처, 그리고 절차적 설계정보를 추출해 내는 것을 (46)_____ 이라 하고, 이와는 달리 분석, 설계로부터 프로그램을 구현하는 것을 (47)_____ 이라 하는데, 이러한 과정들을 통해 보수유지가 곤란한 현존하는 프로그램들을 보수유지가 가능토록하는 것을 (48)_____ 이라 한다.

♥ 일반적 과학변천사는 “hard" science에서 (49)_____ science로 변하고 있으며, 소프트웨어 공학 과정은 (50)_____ 에서 Evolutionary process로 변하고 있다.

[보기]

reengineering(재공학), reverse engineering(역공학), forward engineering(순공학), soft, core(핵심), life, linear thinking(선형적 사고), life thinking(생명적 사고), data design, architectural design(절차 설계), interface design, component-level design, decomposability(분할성), transform, understandability(이해성), monolithic(단일의), abstraction, refinement(상세화), modular, thread-based, use-based, fan-out, fan-in, coupling(결합도), transaction, cohesion(응집도), design model, system image, user's model, user model, black-box testing, white-box testing, debugging, testing, top-down, bottom-up, stub, driver, brute force, backtracking, cause elimination, encapsulate, attributes, abstract, operations, message, polymorphism, inherit, class, responses, responsibilities, constraints, collaborations, control

‘2001년 1학기 S/W 공학 기말고사 답안지

학과: _____ 학번: _____ 이름: _____

(1) (2)

(3) (4)

(5) (6)

(7) (8)

(9) (10)

(11) (12)

(13) (14)

(15) (16)

(17) (18)

(19) (20)

(21) (22)

(23) (24)

(25) (26)

(27) (28)

(29) (30)

(31) (32)

(33) (34)

(35) (36)

(37) (38)

(39) (40)

(41) (42)

(43) (44)

(45) (46)

(47) (48)

(49) (50)

'2002년 1학기 S/W 공학 기말고사 문제지

배점: 각 5점 (총 100점)

1. 구조적 설계에서 변환흐름에 관한 설명과 거리가 먼 것은?

㉮ transform center ㉯ incoming flow
㉰ action path ㉱ outgoing flow

2. 보기(표)에 있는 구조적 설계 절차들을 올바른 순서로 나열한 것은?

<보기>
(1) DFD를 프로그램 구조로 대응시킨다.
(2) 흐름의 범위를 지정한다.
(3) 정보흐름을 개발한다.
(4) 제어계층을 정의한다.
(5) 구조를 정제한다.

㉮ (1)-(2)-(3)-(4)-(5) ㉯ (3)-(2)-(1)-(4)-(5)
㉰ (4)-(5)-(3)-(2)-(1) ㉱ (4)-(5)-(1)-(2)-(3)

3. 사용자 인터페이스 설계시 나타나는 4가지 모델 중 최종사용자의 머릿속에 간직되는 모델은 무엇인가?

㉮ 설계모델(design model) ㉯ 사용자 모델(user model)
㉰ 사용자의 모델(user's model) ㉱ 시스템 이미지(system image)

4. 사용자 인터페이스 설계의 기본 원리가 되는 황금률에 해당되지 않는 것은?

㉮ 상호작용방법을 유지한다. ㉯ 사용자가 제어할 수 있게 한다.
㉰ 사용자의 기억부담을 줄여라. ㉱ 인터페이스를 일관성있게 만들어라.

5. 사용자 인터페이스 설계시 오류 메시지나 경고에 관한 다음의 지침 중 잘못된 것은?

㉮ 메시지는 이해하기 쉬워야 한다.
㉯ 오류로 인해 발생될 수 있는 부정적 결과는 가급적 피한다.
㉰ 오류로부터 회복을 위한 구체적인 설명이 제공되어야 한다.
㉱ 소리나 색 등을 이용하여 듣거나 보기 쉽게 의미 전달을 하도록 한다.

6. 박스다이어그램에 대한 설명 중 잘못된 것은?

㉮ N-S 차트 또는 Chapin 차트라고도 한다.
㉯ 기능영역이 명백히 표현된다.
㉰ 지역 및 전역 데이터의 영역이 쉽게 표현된다.
㉱ 제어의 인위적인 분기가 허용된다.

7. 다음 중 결정표(decision table)에 속하는 항목이 아닌 것은?

㉮ 규칙번호 ㉯ 동작 ㉰ 관계 ㉱ 조건

8. 소프트웨어 시험에 관한 설명 중 적합치 않은 것은?

㉮ 시험이 전체 프로젝트 노력의 30-40%를 차지하는 경우도 종종 있다.
㉯ 심리적으로 볼 때 건설적이라기 보다는 파괴적인 것으로 인식된다.
㉰ 성공적인 시험은 잘 알려진 에러들을 찾아내는 것이다.
㉱ 시험은 에러와 결점이 없다는 것을 완전히 보장해 줄 수 없다.

9. 좋은 시험에 대한 설명 중 잘못된 것은?

㉮ 좋은 시험은 에러를 찾아낼 가능성이 높다.
㉯ 좋은 시험은 중복적이어야 한다.
㉰ 좋은 시험은 “종류 중 가장 좋은 것”이어야 한다.
㉱ 좋은 시험은 너무 단순하거나 또는 너무 복잡하지 않아야 한다.

10. 다음 중 블랙박스 테스팅을 통해 발견하기 힘든 오류는?

㉮ 성능 오류들 ㉯ 외부 정보의 무결성 관련 오류들
㉰ 인터페이스 오류들 ㉱ 논리구조상의 오류들

11. 통합 테스팅에 대한 접근방법이 아닌 것은?

㉮ 빅뱅 테스팅 ㉯ 알파 테스팅
㉰ 점진적 통합 테스팅 ㉱ 회귀 테스팅

12. 다음의 시스템 시험 중 부적절한 외부의 침투로부터 시스템의 보호성을 검증하기 위한 시험은?

㉮ 복구시험(recovery testing) ㉯ 보안시험(security testing)
㉰ 스트레스시험(stress testing) ㉱ 성능시험(performance testing)

13. 디버깅에 대한 설명 중 잘못된 것은?

㉮ 디버깅은 성공적인 시험의 결과로 나타난다.
㉯ 디버깅은 원인에 증상을 연결시킨 불충분하게 이해된 지적 과정이다.
㉰ 디버깅이 힘든 이유는 심리적인 요소가 많이 관여하기 때문이다.
㉱ 디버깅에 대한 체계적인 접근은 아직까지 제안되고 있지 않다.

14. 하향식 통합 테스팅에 대한 설명 중 적절한 것은?

㉮ 사례설계가 용이하다.
㉯ 스터브(stub)가 필요하다.
㉰ 주제어 기능을 일찍 시험할 수 없다.

㉣ 마지막 모듈이 추가될 때까지 실체로서의 프로그램은 존재하지 않는다.

15. Coad와 Yourdon이 정의한 객체지향이란??

㉮ 객체지향 = 객체 + 클래스 + 상속성 + 통신
㉯ 객체지향 = 객체 + 캡슐화 + 메시지
㉰ 객체지향 = 객체 + 추상화 + 상속성
㉣ 객체지향 = 객체 + 클래스계층 + 캡슐화 + 통신

16. 객체지향의 캡슐화(encapsulation) 개념의 특성과 거리가 먼 것은?

㉮ 재사용이 용이하다.
㉯ 인터페이스를 단순화시킨다.
㉰ 연산알고리즘이 단순해진다.
㉣ 변경이 발생할 때, 오류의 파급효과가 적다.

17. 기존의 소프트웨어공학 기법들과 차별화될 수 있는 객체지향 개념이 아닌 것은?

㉮ 캡슐화 (encapsulation)　　㉯ 상속성 (inheritance)
㉰ 다형성 (polymorphism)　　㉣ 모듈성 (modularity)

18. 객체지향 분석에서 말하는 CRC 모델과 관계가 먼 것은?

㉮ Class　　㉯ Representation
㉰ Collaborator　　㉣ Responsibility

19. 다음의 객체지향 통합 테스팅 전략 중 시스템에 대한 하나의 입력이나 이벤트에 응답하기 위해 필요한 클래스들을 통합해 가는 방법은?

㉮ 클래스 (class) 테스팅　　㉯ 클러스터 (cluster) 테스팅
㉰ 사용기반 (use-based) 테스팅　　㉣ 쓰레드기반 (thread-based) 테스팅

20. 보기에 있는 소프트웨어 리엔지니어링 프로세스를 올바르게 나열한 것은?

<보기>
(1) Inventory analysis
(2) Document restructuring
(3) Data restructuring
(4) Code restructuring
(5) Forward engineering
(6) Backward engineering

㉮ (1)-(2)-(4)-(3)-(6)-(5)　　㉯ (1)-(2)-(5)-(3)-(4)-(6)
㉰ (1)-(2)-(3)-(4)-(6)-(5)　　㉣ (1)-(2)-(6)-(4)-(3)-(5)

2002학년도 1학기 소프트웨어공학 기말고사 답안지

학과: 학번: 이름:

(1) (11)
(2) (12)
(3) (13)
(4) (14)
(5) (15)
(6) (16)
(7) (17)
(8) (18)
(9) (19)
(10) (20)

'2003년 1학기 S/W 공학 기말고사 문제지

각 3점 배점 (단, 16번 문제는 4점 배점)

[줄긋기]

1. 소프트웨어 구조를 구성하는 요소들을 소프트웨어 컴포넌트로 바꾸는 것. — 자료설계
2. 소프트웨어가 자신의 내외부, 사용자등과 어떻게 통신하는지를 기술 — 구조설계
3. 소프트웨어의 구조적 요소들간의 관계와 설계패턴을 적용할 때의 제한사항을 정의 — 인터페이스 설계
4. 정보영역 모델을 소프트웨어로 구현하는데 필요한 자료구조로 바꾼다. — 컴포넌트 설계

[번호선택형]

5. 설계 지침으로 적합지 못한 것은?

① 요구사항을 포함해야 한다.
② 쉽게 읽고 이해할 수 있어야 한다.
③ 논리적 관점에서 자료, 기능, 행위 등이 설명되어야 한다.
④ 반복적 방법으로 진행되어야 한다.

[줄긋기]

6. 문제를 모듈로 나눌때, 최상위 단계에서는 광범위한 문제 수준으로, 낮은 단계에서는 구체적인 절차를 기술한다. — 정제
7. 기능을 단계적으로 상세화하여 프로그램 언어의 명령어 수준에 도달하는 방법으로 프로그램이 개발된다. — 추상화
8. 프로그램을 지적으로 다룰 수 있게 하는 소프트웨어의 속성을 말한다. — 모듈양식
9. 프로그램 컴포넌트, 이들의 상호작용 방법, 컴포넌트에 의해 사용되는 자료구조 등의 계층적 구조를 말한다. — 소프트웨어 구조

[단답선택형]

10. 프로그램 구조는 수평적과 수직적으로 분리된다. 수평적 분리는 (입력-자료변환-출력, 제어와 실행)로 나뉘고, 수직 분리는 (입력-자료변환-출력, 제어와 실행)로 나뉜다.

[줄긋기]

11. 하나의 모듈은 다른 모듈들이 쉽게 액세스할 수 없도록 설계되어야 한다. — 정보은닉
12. 응집력과 결합도에 의해 측정. — 기능독립성
13. 모듈간의 상호연결을 측정. — 응집도
14. 소프트웨어 프로시저가 가급적 하나의 작업을 수행하면서 다른 프로시저와는 상호작용을 얼마나 적게 하는가를 측정. — 결합도

[단답선택형]

15. 프로그램 구조는 (달걀형, 팬케잌형, 소나무형)을 유지해야 한다.

[주관식]

16. 아래 DFD 그림에서 프로세스 b를 Transaction center로 놓고, 영역 I, II, III을 각각 Transform 흐름으로 놓고 프로그램구조를 그리시오. [4점]

[번호선택형]

17. 인터페이스 설계의 3가지 황금률이 아닌것은?

① 사용자가 제어할 수 있게 한다.
② 논리적 관점이 아닌 구현의 관점에 치중하라.
③ 사용자의 기억부담을 줄여라.
④ 일관성 있게 만들어라.

[단답선택형]

18. 에러를 발견하는 것을 (testing, debugging) 이라 하고, 에러의 원인을 찾는 것을 (testing, debugging) 이라 한다.

19. 시험은 (큰, 작은) 부분에서 시작하여 (큰, 작은) 부분으로 옮겨야 한다.

20. 소프트웨어 인터페이스에서 진행되는 시험을 (블랙박스, 화이트박스) 시험이라 하고, 논리적 경로들을 검사하는 것을 (블랙박스, 화이트박스) 시험이라 한다.

[번호선택형]

21. 기본 경로 시험에서 흐름도를 살펴보니 4개의 영역을 갖고 있으며, 간선 (edge)은 11개 노드는 9개이었다. 순환복잡도는 얼마인가?

① 2 ② 3 ③ 4 ④ 5

[줄긋기]

22. 소프트웨어의 각 구성요소에 집중한다.	system testing
23. 소프트웨어의 설계에 초점을 둔다.	validation testing
24. 소프트웨어 요구사항에 중점을 둔다.	integration testing
25. 소프트웨어와 다른 시스템 요소를 하나로 시험한다.	unit testing

[단답선택형]

26. (유도자, 뿌리)는 시험사례 데이터를 받아들여서 구성 요소로 보내고 결과를 인쇄하는 부수적인 프로그램을 말한다.

27. Validation testing 중 사용자에 의해 개발자의 위치에서 진행되는 시험을 (alpha testing, beta testing) 이라 한다.

[번호선택형]

28. 객체지향 방법의 encapsulation이 갖는 장점과 거리가 먼 것은?

① 정보은닉 ② 재사용 ③ 낮은 결합도 ④ 높은 상속성

29. 객체지향 분석에서 말하는 CRC 모델링에 해당되지 않는 것은?

① Class ② Representation ③ Responsibility ④ Collaborator

[단답선택형]

30. OOA는 (분류화, 상세화) 작업인 반면, OOD는 (분류화, 상세화) 작업이라 볼 수 있다.

31. OO 통합 테스팅 전략중 (스레드기반 테스팅, 사용기반 테스팅)은 시스템에 대한 하나의 입력이아 이벤트에 응답하기 위해서 필요한 클래스들의 집합을 통합하는 테스칭을 말한다.

32. 수학적 명세를 도입한 소프트웨어 공학 기법을 (정형적, 컴포넌트기반, 클라이어트/서버) 방법이라 한다.

33. 소스코드로부터 설계 복구를 위한 프로세스를 (re-engineering, reverse engineering, forward engineering, document restructuring)이라 한다.

'2004 1st Semester S/W Engr. Final Exam.

[Choose one] 1 point each

A design should be (1)_____; that is the SW should be logically partitioned into elements that perform specific function and subfunctions.

(2)_____ is a top-down design strategy where a program is developed by successively refining levels of procedural detail.

(3)_____ represents the organization of program components and implies a hierarchy of control.

(4)_____ is a measure of the number of modules that are directly controlled by another module, however, (5)_____ indicates how many modules directly control a given module.

The horizontal partitioning of program structure defines three partitions -- (6)_____, (7)_____, and (8)_____. The vertical partitioning suggests the (9)_____ and (10)_____.

The adventage of (11)_____ is that the inadvertent errors introduced during modification are less likely to propagate to other locations within the SW.

(12)_____ is a measure of the relative functional strength of a module. (13)_____ is a measure of the relative interdependence among modules.

Transform flow in architectural design is characterized as (14)_____, transform flow, and (15)_____, however, the transaction flow has (16)_____ and transaction center.

SW testing represent the ultimate review of (17)_____, (18)_____, and (19)_____.

(20)_____ testing alludes to tests that are conducted at the SW interface, however, (21)_____ testing is predicated on close examination of procedural detail.

In the unit testing, (22)_____ is nothing more than a main program, however, (23)____ is for subordinate module.

Top-down integration testing requires (24)_____ module but the bottom-up integration testing requires (25)_____ module.

(26)_____ testing is designed to confront programs with abnormal situation, however, (27)_____ testing attempts to verify that protection mechanisms built into a system will protect it from improper penetration.

Object-oriented = objects + (28)_____ + (29)_____ + communication

(30)_____ means that all of information including data and operation is packaged under one name and can be reused as one specification or program component.

(31)_____ enables a number of different operations to have the same name.

In object-oriented analysis, (32)_____ models the system from the end-user's point of view and (33)_____ provides a simple means for identifying and organizing the classes that are relavent to system or product requirements.

(34)_____ testing integrates the set of classes required to respond to one input or event for the system, however, (35)_____ testing begins the constrcution of the system by testing those classes that use very few of server classes.

In (36)_____ software engineering, a specification and design are descibed using a formal syntax and sementics that specify system function and behavior.

(37)_____ software engineering emphasizes rigor in specification and design, and formal verification of each design element using correctness proofs that are mathematically based.

Inventory analysis --> Document restructuring --> (38)_____ --> Code restucturing --> Data restructuring --> (39)_____. This process is called (40)_____.

[Draw line] 1 point each

(41) Data design	▪	▪ transform structural element into a procedural description
(42) Architectural design	▪	▪ describe how the SW communicate with itself, system, and human
(43) Interface design	▪	▪ define the relationship between major structural element
(44) Component-level design	▪	▪ transforms the information domain model into the data structure.

(45) Graphical design notation	▪	▪ N-S chart
(46) Tabular design notation	▪	▪ rule
(47) Program design language	▪	▪ pseudocode

(48) verification	▪	▪ refers to a different set of activities that ensure that the SW that has been built is traceable to customer requirements.
(49) validation	▪	▪ refers to the set of activities that ensure that SW correctly implements a specific function.

(50) unit testing	▪	▪ concentrate on whole component
(51) integration testing	▪	▪ check the SE requirement
(52) validation testing	▪	▪ focus on SW architecture
(53) system testing	▪	▪ concentrates on each component

(54) alpha testing	▪	▪ conducted at the developer's site.

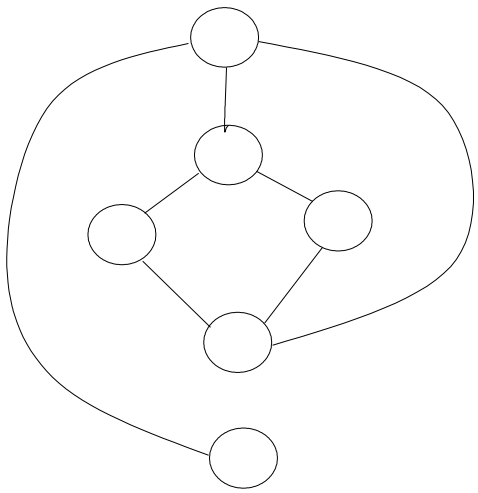
(55) beta testing ▪ ▪ conducted at the customer's site.

(56) Data ▪ ▪ associativity within one context
(57) Information ▪ ▪ no associativity
(58) Knowledge ▪ ▪ creation of generalized principles
(59) Wisdom ▪ ▪ associativity within multiple contexts

[Answer] 10 point each

(42) Draw a flowchart and box diagram of If-then-else procedure, respectively.

(43) What is the cyclomatic complexity, V(G), in the following flow graph?



{Example} specification, recovery, modular, hierarchical, fan-out, input, coupling, program structure, forward engineering, fan-in, level of abstraction, CRC, control, repreatable, data structure, width, superordinate, subordonate, information hiding, driver, action path, depth, output, stepwise refinement, data transformation, cohesion, composability, work, use-based, structural partitioning, outgoing flow, data coupling, stub, security, design, performance, white-box, functional independence, black-box, code generation, stress, reverse engineering, regression, incoming flow, classification, modularity, inheritance, client-server, encapsulation, polymorphism, thread-based, CASE, web-based, cleanroom, use-case, formal method, component-based, SW reengineering

'2005년 1학기 S/W 공학 기말고사 문제지 및 답안지

학과: 학번: 이름:

◉ 다음 보기를 보고 적절한 용어를 선택하시오.

보기: Validation, Verification, Qualification

(1). _____ refers to the set of activities that ensure that s/w correctly implemented a specific function. (2). _____ refers to a different set of activities that ensure that s/w that has been built is traceable to customer requirements.

◉ 주요 특성간의 관계를 연결하시오.

(3). Integration testing	o	o concentrate on each component
(4). Unit testing	o	o focus on s/w architecture
(5). Validation testing	o	o s/w requirement analysis is tested
(6). System testing	o	o s/w and other system elements are tested as a whole

◉ 다음 보기를 보고 적절한 용어를 선택하시오.

보기: Driver, Cluster, Stub

(7) _____ is nothing more than a main program. (8) _____ serve as a subordinate to the component to be tested.

◉ 다음 보기를 보고 적절한 용어를 선택하시오.

보기: Validation testing, Performance testing, Alpha testing, Beta testing

(9) _____ is conducted at the developer's site by end-users.

(10) _____ is conducted at end-user's sites.

◉ 다음 보기를 보고 적절한 용어를 선택하시오.

보기: testing, analyzing, debugging

(11) _____ occurs as a consequence of the successful (12) _____.

◉ 주요 특성간의 관계를 연결하시오.

(13). Black-box testing o o knowing the specified function

(14). White-box testing o o knowing the internal workings.

(15) 다음중 white-box testing기법에 해당되는 것은?

① basis path testing ② equivalance partitioning

③ boundary value analysis ④ orthogonal array testing

(16) S/W project management에서 말하는 네가지 P는?

◉ 주요 특성간의 관계를 연결하시오.

(17). end-users o o define business issues

(18). customers o o plan, motivate, organize, control the practitioners

(19). practitioners o o interact with the s/w

(20). project managers o o specify the requirements

(21). senior managers o o deliver the technical skill

◉ 주요 특성간의 관계를 연결하시오.

(22). indirect measure o o LOC, execution speed, defect, etc.

(23). direct measure o o functionality, quality, efficiency, etc.

◉ 어느 project의 진행 결과를 보면, 5명이 10개월간 일하여 10000 LOC의 결과물을 얻었고, defect는 50개이며, 이 때 비용은 $1,000,000가 소요되었다.

(24) 이 project의 effort(PM)은 얼마인가?

(25) 이 project의 productivity(LOC/PM)은 얼마인가?

(26) 이 project의 quality(defect/PM)은 얼마인가?

(27) 다음중 S/W resource의 세가지 유형에 해당하지 않는 것은?

① test case ② people

③ reusable s/w component ④ development environment

(28) 어느 S/W 기능에 대한 LOC 범위는 각각 optimistic: 4,000 LOC, most likely: 5,000 LOC, pessimistic: 9,000 LOC로 추정되었다. 이 S/W 기능에 대한 expected value를 구하시오.

◉ 주요 특성간의 관계를 연결하시오.

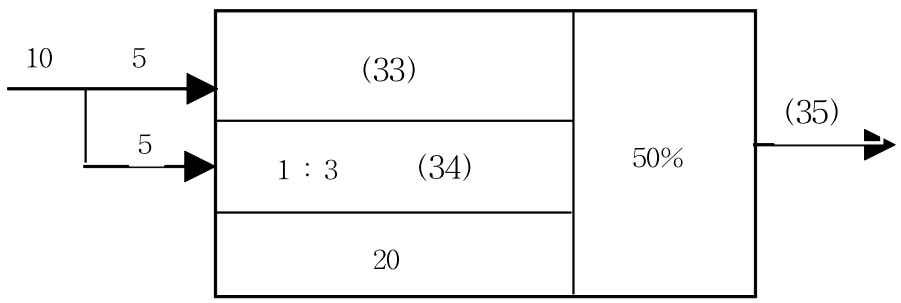
(29). risk identification o o assumes the mitigation effort have failed.

(30). risk projection o o likelihood and consequences

(31). risk mitigation o o systematic attempt to specify threats to the project plan

(32). risk management o o proactive approach to aviod the risk

◉ 다음의 defect amplification model을 보고 해당난의 결함 개수를 구하시오.



◉ 주요 특성간의 관계를 연결하시오.

(36). clean room s/w engineering o o use mathematical specification

(37). formal method o o use box structure specification

◉ 다음 보기를 보고 적절한 용어를 선택하시오.

보기: reverse engineering, reengineering, forward engineering

(38) _____은 (39) _____을 통해 code, data 등을 재구성한 뒤, (40) _____을 통해 완성된다.

◉ 다음 보기를 보고 적절한 용어를 선택하시오.

보기: Data, Information, Knowledge, Wisdom

(41) _____ is derived by associating facts within a given context. (42) _____ occurs when generalized principles are derived from disparate knowledge.

‘2007년 1학기 S/W 공학 기말고사 문제지 및 답안지

학과: 학번: 이름:

[1] Problem solving 절차를 순서대로 올바르게 표현한 것은 ?

<보기>

ⓐ carry out the plan ⓑ examine the result
ⓒ plan the solution ⓓ understand the problem

1. ⓓ - ⓑ - ⓐ - ⓒ
2. ⓓ - ⓑ - ⓒ - ⓐ
3. ⓓ - ⓒ - ⓑ - ⓐ
4. ⓓ - ⓒ - ⓐ - ⓑ

[2] 요구사항 분석을 위한 communication 요령으로 적절치 않는 것은?

1. prepare before you communicate 2. stay focused, modularize your discussion
3. free Email communication is best 4. strive for collaboration

[3] "A complete delivery package should be assembled and tested."라는 말은 소프트웨어 공학의 어느 단계에 적합한 원칙인가?

1. Analysis modeling 2. Design modeling 3. Testing 4. Deployment

[4] "The pareto principle should be applied."라는 말은 소프트웨어 공학의 어느 단계에 적합한 원칙인가?

1. Analysis modeling 2. Design modeling 3. Testing 4. Deployment

[5] "Always consider the architecture of the system to be built."라는 말은 소프트웨어 공학의 어느 단계에 적합한 원칙인가?

1. Analysis modeling 2. Design modeling 3. Testing 4. Deployment

[6] "The functions that the software performs must be defined."라는 말은 소프트웨어 공학의 어느 단계에 적합한 원칙인가?

1. Analysis modeling 2. Design modeling 3. Testing 4. Deployment

[7] Computer-based system의 여섯 가지 시스템 요소들을 나열하시오.

_____, _____, _____, _____, _____, _____,

* Requirement engineering을 달성하기 위한 7가지 기능들과 그에 적합한 내용들을 연결하시오.

[8] Inception

[9] Elicitation

[10] Elaboration

[11] Negotiation

[12] Specification

[13] Validation

[14] Management

1. set of context-free questions

2. analysis modeling

3. quality

4. standard template

5. scope

6. conflict

7. identify, control, track

* 다음 Analysis model들을 참조하여 아래 질문에 답하시오.

<보기>

1. Entity-relationship Diagram (ERD)
2. Activity Diagram
3. Use-case Diagram
4. Swim-lane Diagram
5. Data Flow Diagram (DFD)
6. Class Diagram
7. Process Specification (PSPEC)
8. CRC Index card
9. Collaboration Diagram
10. State Diagram
11. Sequence Diagram

[15] Data modeling에 관계하는 모델들은?

[16] Flow-oriented modeling에 관계하는 모델들은?

[17] Class-based modeling에 관계하는 모델들은?

[18] cardinality와 modality는 어느 모델에서 사용하는가?

[19] use case에서 정의된 특정 시나리오의 상호작용 흐름을 actor 별로 구분하지 않고 그래픽하게 표현한 모델은?

[20] context diagram은 어는 모델에서 표현될 수 있는가?

[21] Class들의 상태와 상태변환을 일으키는 이벤트간의 관계를 시간 순으로 표현한 모델은?

* 다음 Design 개념들에 적합한 용어들을 연결하시오.

[22] Abstraction — 1. elaboration

[23] Architecture — 2. cohesion

[24] Modularity — 3. error propagation

[25] Information hiding — 4. divide & conquer

[26] Functional Independence — 5. structure

[27] Refinement — 6. data, procedure, control

[28] Component-level design에 대한 설명 중 적절치 못한 것은?

1. cohesion은 높아야 한다.
2. coupling은 낮아야 한다.
3. 반복적이어서는 안된다.
4. 항상 대안을 고려해야 한다.

[29] User interface design을 위한 황금규칙과 관련이 없는 것은?

1. Make the interface consistent.
2. Reduce the user's memory load.
3. Elaborate the behavioral representation
4. Place the user in control

[30] Testing과 가장 거리가 먼 것은?

1. Verification 2. Validation 3. Quality 4. Maintenance

[31] 네 개의 testing strategy를 순서대로 나열하시오.

1. _____ testing, 2. _____ testing, 3. _____ testing, 4. _____ testing

[32] 위의 네 가지 전략 중 driver와 stub를 필요로 하는 전략을 모두 나열하시오.

[33] top-down, bottom-up, sandwich 등의 접근이 필요한 전략을 모두 나열하시오.

[34] alpha, beta testing 등은 어느 전략에서 실행되는가?

[35] Security testing과 stress testing 등은 어느 전략에서 필요한지 모두 나열하시오.

[36] Debugging 전략에 해당되지 않는 것은?

1. brute force
2. stepwise refinement
3. backtracking
4. cause elimination

[37] Testing tactics에는 black-box와 white-box 방법이 있다. 독립된 실행경로를 찾아서 testing 하는 것은 무슨 방법인가?

[38] Boundary value analysis(BVA)는 어느 방법에 해당 되는가?

[39] Testing 과정 중 초기단계에 적용되기 적합한 방법은?

[40] Cleanroom software engineering에 대한 설명 중 적합하지 않은 것은?

1. incremental 전략을 사용한다.
2. 고도의 high-quality s/w를 위한 것이다.
3. 통계적 testing 방법을 사용한다.
4. use case를 기반으로 한다.

[41] dirty source code로부터 final specification을 추출해내는 과장을 무엇이라고 하는가?

1. reverse engineering
2. reengineering
3. forward engineering
4. requirement engineering

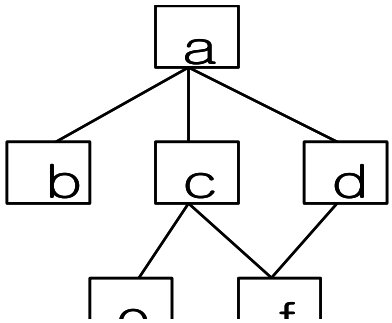
[42] 다음 컴퓨터 핵심 용어들을 발전 순서대로 바르게 나열하시오.

<보기>

Knowledge, Wisdom, Information, Data

1. _____ 2. _____ 3. _____ 4. _____

* 다음 S/W architecture를 보고 아래 질문에 답하시오.

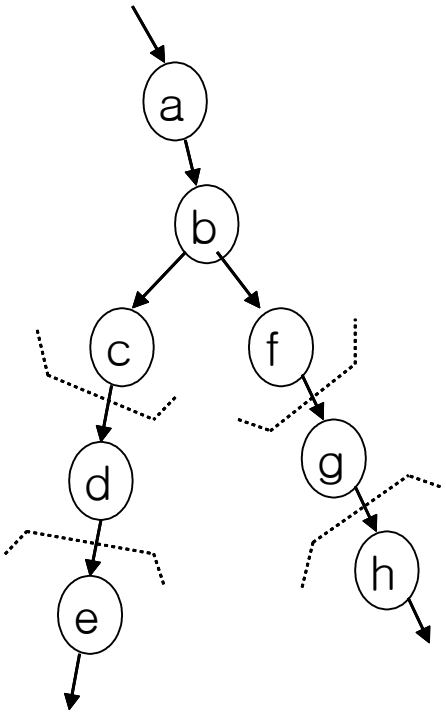


[43] size는 얼마인가?

[44] width는 얼마인가?

[45] arc-to-node ratio는 얼마인가?

[46] 다음의 DFD를 토대로 S/W architecture를 그리시오. {10점}



‘2008년 1학기 S/W 공학 기말고사 문제지 및 답안지

학과: 학번: 이름:

<설계>

[1] Component 설계에서 강조된 용어들과 거리가 먼 것은?

1. cohesion 2. coupling 3. DFD 4. activity diagram

* 바르게 연결하시오.

[2] Graphical design notation — Pseudocode

[3] Tabular design notation — Decision table

[4] Program design language — Flowchart

[5] Interface 설계에서 세 개의 “Golden rules"에 해당되지 않는 것은?

1. Allow the visualization 2. Place the user in control
3. Reduce the user's memory load 4. Make the interface consistent

* 바르게 연결하시오.

[6] User model — combines outward manifestation of computer system

[7] Design model — image of system that end users carry in their head

[8] User's mental model — establishes the profile of end users of the system

[9] Implementation model — incorporates data, architecture, interface, procedure

[10] user interface 설계과정은 spiral model에 의해 잘 표현되는데, 다음의 activity들을 바른 순서대로 나열하시오. (3점)

1. interface construction 2. interface validation
3. interface design 4. user, task, environment analysis and modeling

<테스팅>

[11] 테스팅의 일반적 특성에 해당되지 않는 것 두 가지는? (3점)

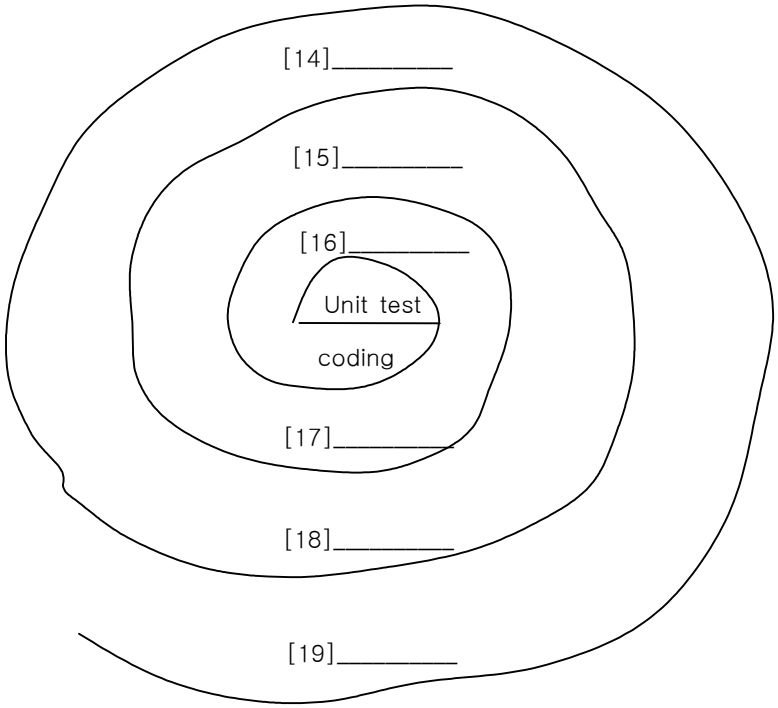
1. should conduct informal review.
2. begins at the component level toward integration.
3. different techniques are appropriate at different point in time.
4. conducted by the developer not by independent test group.
5. debugging must be accommodated in any testing strategy.

* 바르게 연결하시오.

[12] Verification — Are we building the product right?

[13] Validation — Are we building the right product?

* 빈 칸에 들어갈 적절한 용어를 채우시오.



[20] Unit testing에서의 테스팅 대상이 아닌 것은?

1. module interface
2. local data structure
3. error handling path
4. architecture

[21] 테스팅 전략에 관한 설명 중 잘못된 것은?

1. driver는 테스트 대상 모듈의 상위 모듈을 대체한다.
2. sandwich testing은 top-down과 bottom-up testing의 상호 보완을 위한 개념이다.
3. Acceptance testing에는 alpha testing과 beta testing이 있다.
4. performance testing은 validation test에 속한다.

[22] Debugging 전략에 해당되지 않는 것은?

1. brute force
2. backtracking
3. cause elimination
4. forward chaining

[23] 테스트의 효용성을 의미하는 “Testability"에 해당되지 않는 것은?

1. operability
2. simplicity
3. maintainability
4. understandability

* 바르게 연결하시오.

[24] white-box testing — boundary value analysis

[25] black-box testing — basis path testing

* 바르게 연결하시오.

[26] white-box testing　　knowing the special function

[27] black-box testing　　knowing the internal workings

<프로젝트 관리>

* 일명 chief programmer team이라 부르는 대표적인 closed paradigm의 팀조직 중 빈 곳을 채우시오. (3점)

senior engineer, technical staff, [28]_____, specialists, support staff, and [29] _____

[30] Project manager에게 project metrics가 필요한 이유에 해당되지 않는 것은?

1. access the status of on-going project　　2. track potential risks
3. adjust work flow　　4. increase the modularity

* 2명의 인원이 10effort를 통해 총 100KLOC 크기의 프로젝트를 완성하였는데, 고객에게 인도되기 전에 발견한 오류수는 100개이고, 인도 후에 발견된 오류수는 10개였다.

[31] 개발기간은 얼마인가?

[32] defect per KLOC는 얼마인가?

[33] error per person-month는 얼마인가?

[34] S/W project estimation 요령 중 적절치 못한 것은?

1. delay estimation until late　　2. use the refactoring concept
3. use empirical model　　4. base on similar project

[35] estimation 방법에 해당되지 않는 것은?

1. FP-based estimation　　2. process-based estimation
3. S/W equation　　4. component-based estimation

[36] Scheduling 원칙이 아닌 것은?

1. compartmentalization　　2. independence
3. time allocation　　4. defined milestones

[37] 노력안배에 대한 40-20-40 규칙에서 20에 해당되는 것은?

1. 분석　　2. 설계　　3. 코딩　　4. 테스팅

[38] task network와 거리가 먼 것은?

1. activity network　　2. PERT　　3. critical path　　4. responsibility

[39] RMMM에 해당되지 않는 것은?

1. risk mapping　　2. risk mitigation
3. risk monitoring　　4. risk management

[40] user satisfaction을 위한 요소들과 거리가 먼 것은?

1. compliant product
2. good quality
3. small number of component
4. delivery within budget and schedule

[41] defect amplification model의 구성요소가 아닌 것은?

1. level of expertise
2. errors passed through
3. newly generated errors
4. percent efficiency for error detection

[42] _____은 정당한 변경을 순조롭게 이끌어 주기 위한 S/W 형상관리 개념으로 S/W 개발을 위한 milestone을 말한다.

1. baseline
2. SCI (Software configuration item)
3. configuration audit
4. ECO (engineering change order)

* 바르게 연결하시오.

[43] reengineering	extract data, architectural, procedural design information from an existing program
[44] reverse engineering	reconstitute the existing S/W
[45] forward engineering	deal with unmaintainable S/W problem
[46] component-based software engineering	emphasize the design and construction based in the reusability
[47] cleanroom software engineering	lead to extremely high-quality software by using the box structure specification
[48] formal method	mathematical representation

2009학년도 1학기 소프트웨어공학 기말고사 문제지 및 답안지

학번: _____ 이름: _____

◈ 50문제 배점 각 2점 총 100점

1. Requirement analysis (model)에 대한 설명 중 잘못된 것은?
① information, function, behavior 모델링을 중심으로 한다.
② how 보다는 what에 초점을 둔다.
③ 고객에게 quality를 평가할 근거를 마련해준다.
④ System description과 Design model간의 교량 역할을 한다.
⑤ 가급적 구현의 관점을 고려하여야 한다.

다음 분석 모델의 요소들(elements)을 바르게 연결하시오.

2. Data Element	Activity Diagram
3. Scenario-based Element	Data Flow Diagram
4. Flow-oriented Element	CRC model
5. Class-based Element	ERD
6. Behavioral Element	State Diagram

Data modeling에는 data object, attribute, relationship 이외에도 반복적 발생관계를 표현하는 7. _____과 선택적 발생관계를 표현하는 8. _____등이 있다.

9. use-case에 대한 설명 중 잘못된 것은?
① producer와 consumer간의 상호작용을 나타낸다.
② use-case template에는 primary actor, precondition, scenario 등이 명시된다.
③ use-case diagram에서 각 use-case는 타원형으로 표기된다.
④ 분석모델들 중 가장 나중에 완성된다.
⑤ Scenario-based modeling에서 필요로 한다.

DFD에 관한 연결 관계를 완성하시오

10. Level 0 DFD	data object
11. Labeled arrows	context diagram
12. Circle	transformation (process)
13. Rectangle	external entity

14. DFD의 최하위 프로세스에 대한 명세 표현은?
① activity diagram ② Process Specification (PSPEC)
③ Control flow model ④ Sequence diagram ⑤ Swimlane diagram

15. Class-based modeling과 거리가 먼 것은?
① aggregate ② multiplicity ③ dependencies
④ package ⑤ state transition

16. S/W 설계에 대한 설명 중 적절치 못한 것은?
① Architecture 설계는 S/W의 구조적 요소들 간의 관계를 정의한다.
② S/W 설계는 분석 모델의 상세화인 바, 반복적이어서는 안된다.
③ 읽기 쉽고 이해 가능해야 한다.
④ 분석모델에서 명시한 사항들이 빠져서는 안된다.
⑤ S/W의 완전한 밑그림이 제시되어야 한다.

17. 'S/W는 독립적 이름을 갖는 컴포넌트들로 나누어져야 한다.'는 설계 개념은?
① Abstraction ② Architecture ③ Patterns
④ Modularity ⑤ Refactoring

18. 모듈의 기능적 강도를 나타내는 설계 개념은?
① cohesion ② coupling ③ functional independence
④ information hiding ⑤ refinement

19. 에러의 파급효과를 효과적으로 줄이기 위한 설계 개념은?
① cohesion ② coupling ③ functional independence
④ information hiding ⑤ refinement

20. Abstraction 개념과 상대적이면서도, 상호 보완적인 설계 개념은?
① cohesion ② coupling ③ functional independence
④ information hiding ⑤ refinement

21. Main program과 subprogram들이 제어계층을 이루는 설계 아키텍처는?
① Layed architecture ② Call and return architecture
③ Data flow architecture ④ Data-centered architecture
⑤ Client-server architecture

22. 분석 모델 DFD로부터 아키텍처 설계모델을 이끌어내기 위해서는 Transform flow와 Transaction flow를 검토해야 한다. 이 중 여러 개의 선택적인 action path를 갖는 것은 _____이다.

23. 다음 coupling 유형들을 낮은 것부터 높은 순서대로 나열하시오.

_____ --> _____ --> _____ --> _____
content coupling, common coupling, control coupling, data coupling

24. Component-level design과 거리가 먼 것은?
① Flowchart ② Decision table ③ Pseudo-code
④ Activity diagram ⑤ DFD

25. 다음 user interface 설계모델 중 사용자 입장에서 예상하는 모델은?
① user model ② design model ③ mental model
④ implementation model ⑤ use-case model

26. User interface 분석 및 설계 과정은 보통 spiral model에 의해 진행되는데, 이 과정을 바른 순서대로 나열하시오.
_____ --> _____ --> _____ --> _____
user/task/environment analysis, validation, interface construction, interface design

27. Testing strategy에 대한 설명 중 적절치 못한 것은?
① Formal Technical Review가 필요하다.
② 시스템 레벨에서 컴포넌트 레벨까지 “inward"로 진행된다.
③ 테스팅 과정의 시간대별로 다른 testing techniques가 적용되어야 한다.
④ 소프트웨어 개발자와 독립적 테스트 그룹이 모두 간여해야 한다.
⑤ 디버깅은 테스팅의 결과로 진행되는 별도의 작업이다.

V&V 개념 중 소프트웨어가 주어진 기능들을 올바르게 구현하였는지에 대한 확인 작업을 28. _____라 하고, 소프트웨어가 고객의 요구사항들을 빠짐없이 반영하였는지에 대한 확인 작업을 29. _____라 한다.

다음의 개발단계와 테스팅단계 간의 연관 관계를 완성하시오.
30. System Engineering — unit testing
31. Requirement — integration testing
32. Design — validation testing
33. Code — system testing

Unit testing시 컴포넌트는 독립적 프로그램이 아니므로 상위의 컴포넌트를 대신할 34. _____와 하위의 컴포넌트들을 대신할 35. _____을 필요로 한다. 따라서 Top-down integration testing은 36. _____을 필요로 하고, Bottom-up integration testing에서는 37. _____을 필요로 한다.

38. 시스템 다운시 초기화, 체킹, 데이터복구, 재가동 등의 적절한 복구여부의 확인을 위한 테스트 방법은?

① Recovery testing ② Security testing ③ Stress testing
④ Performance testing ⑤ Gamma testing

39. 에러의 증상으로부터 원인을 발견할 때까지 역추적하는 디버깅 전략은?

① Brute force ② Backtracking ③ Cause elimination
④ automated debugging ⑤ configuration review

40. Basis path testing을 위해 flow graph를 분석하였더니, edge가 11개, node가 9개였다. cyclomatic complexity(독립적 실행경로)는 얼마인가?

① 11 ② 9 ③ 2 ④ 4 ⑤ 5

관련 있는 항목끼리 연결하시오.

41. basis path testing — white-box testing
42. equivalence partitioning — black-box testing
43. boundary value analysis

다음 소프트웨어공학의 신개념들을 바르게 연결하시오.

44. Formal methods — High-quality S/W 개발시 적용
45. Cleanroom S/W Engineering — 수학적 명세를 통한 방법
46. Component-based development — 재사용성을 강조한 방법
47. Reengineering — 보수유지성 향상을 위한 방법

48. 다음 정보의 스펙트럼을 바른 순서대로 나열하시오.

_____ --> _____ --> _____ --> _____

Information, Wisdom, Knowledge, Data

49. System 전문가로서 갖추어야 할 덕목과 거리가 먼 것은?

① Philosophy ② Management ③ S/W Engineering
④ Domain knowledge ⑤ Stock market

50. 소프트웨어공학은 나에게 있어서 _____ 이다.

왜냐하면_____

_____

_____ 이기 때문이다.

2010학년도 1학기 소프트웨어공학 기말고사 문제지 및 답안지

학번: **이름:**

1. 시스템 모델링에 필요한 주요 세 가지 모델과 거리가 먼 것은? [5점]

㉮ performance model
㉯ functional model
㉰ behavioral model
㉱ data model

2. 분석 모델링에는 data element, scenario-based element, flow-oriented element, class-based element, behavioral element 등 다섯 가지를 들 수 있다. 각 element 별로 두 개씩의 모델 (diagram 또는 table 등)을 나열하시오. (분석 모델링 과제를 상기할 것!) [10점]

3. 다음 설계 개념들과 주요 특성에 관한 관련성을 연결하시오. [10점]

㉮ abstraction	divide & conquer
㉯ architecture	organization of components
㉰ modularity	suppress low-level detail
㉱ information hiding	re-organization
㉲ refinement	elaboration
㉳ functional independence	cohesion/coupling
㉴ refactoring	reduce error propagation

4. 설계는 data 설계, architecture 설계, component 설계 그리고 GUI 설계 등 네 가지로 정리된다. cohesion과 coupling개념은 _____ 설계에 적용되며, transform mapping과 transaction mapping은 _____ 설계에 활용되며, user analysis와 task analysis는 _____ 설계에 필요하며, data mining과 data warehouse 개념은 _____ 설계에 사용된다. [5점]

5. GUI 설계를 위한 황금규칙에 해당되지 않는 것은? [5점]
㉮ Place the user in control
㉯ Use the user's image model
㉰ Reduce the user's memory load
㉱ Make the interface consistent

6. Stub와 driver의 용도에 대하여 각각 설명하시오. [10점]

7. 테스트 용이성을 나타내는 testability의 항목으로 적합하지 않은 것은? [5점]
㉮ operability
㉯ controllability
㉰ invulnerability
㉱ decomposability

8. Debugging에 관한 설명 중 적절치 못한 것은? [5점]
㉮ debugging은 testing에 앞서 시행된다.
㉯ 먼저 가설을 세운 뒤, 확인하는 방법을 cause elimination이라 한다.
㉰ symptom(증상)에서 원인을 추적하는 방법을 backtracking이라 한다.
㉱ debugging 솜씨는 타고나는 것으로 알려져 있다.

9. Testing strategy 네 가지를 나열하고 각각 간략히 설명하시오. [10점]

10. Testing tactics에는 black-box testing과 white-box testing이 있다. _____에는 equivalence partitioning기법과 boundary value analysis기법 등이 있고, _____에는 basis path testing기법이 대표적이다. _____는 일명 glass-box testing이라고도 하며, _____는 일명 behavioral testing이라고 한다. [5점]

11. 고품질 소프트웨어를 개발하기 위하여 black box, state box, clear box개념 등을 도입한 functional specification기법을 기반으로 하는 방법을 _____ software engineering이라 한다. [5점]

12. 컴포넌트기반 개발 방법론에서, 기존 컴포넌트에서 불필요한 특성들에 대한 약간의 수정을 가한 컴포넌트를 무엇이라고 하는가? [5점]

㉮ Qualified component ㉯ Adapted component
㉰ Assembled component ㉱ Updated component

13. 다음의 S/W reengineering cycle에 들어갈 적절한 용어를 쓰시오. [5점]

inventory analysis ==> document restructuring ==> _____ ==> code restructuring ==> data restructuring ==> _____ ==> inventory analysis

14. 다음 약자들의 full name을 쓰시오. [10점]

㉮ CASE: _____
㉯ UML: _____
㉰ SQA: _____
㉱ ERD: _____
㉲ DFD: _____
㉳ CRC: _____

15. 소프트웨어 공학의 미래 발전 동향에 관한 설명 중 적절한 것은? [5점]

㉮ soft science에서 hard science로
㉯ evolutionary model에서 linear thinking으로
㉰ knowledge 기반에서 information 기반으로
㉱ 독립적 IT에서 융합형 IT로

2011학년도 1학기 소프트웨어공학 기말고사 문제지

1. 소프트웨어공학과정의 에센스(정수)를 순서대로 나열하시오. [3점]

㉮ Carry out the plan　　㉯ Examine the result for accuracy
㉰ Plan a solution　　㉱ Understand the problem

2. 시스템에 대한 일반적 정의에 적합지 않은 것은? [3점]

㉮ Something with input and output　　㉯ Source of data
㉰ Set of facts, principles, rules, etc.　　㉱ Descriptive information

3. 컴퓨터기반 시스템의 요소들과 거리가 먼 것들을 모두 고르시오. [3점]

㉮ H/W　㉯ People　㉰ Product　㉱ Subsystem　㉲ Database
㉳ Documentation　㉴ Procedures

4. 시스템 모델링의 세 가지 주요 관점이 아닌 것은? [3점]

㉮ Functional modeling　　㉯ Behavioral modeling
㉰ Transactional modeling　　㉱ Data modeling

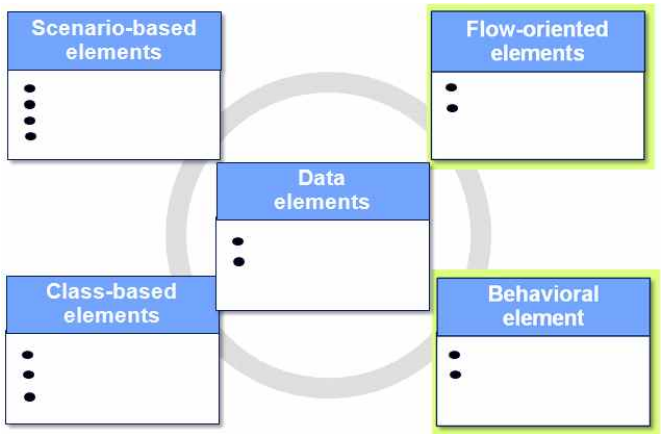
5. 요구사항공학(Requirement Engineering)은 다음 중 일곱 가지 기능들의 실행을 통해 달성될 수 있다. 해당 되지 않는 것은? [3점]

㉮ Inception　㉯ Transition　㉰ Elicitation　㉱ Elaboration　㉲ Negotiation
㉳ Specification　㉴ Validation　㉵ Requirement Management

6. Use-case 명세시 불필요한 것은? [3점]

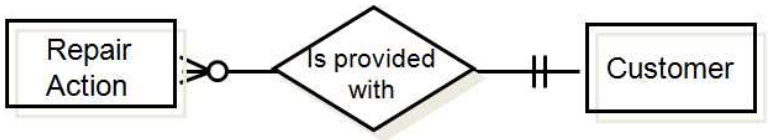
㉮ 주요 액터는 누구인가?　　㉯ 액터의 목적(goal)은 무엇인가?
㉰ 선행조건은 무엇인가?　　㉱ 구현환경은 무엇인가?

7. 아래 그림에 있는 분석 모델 요소들을 보기를 참조하여 채우시오. [10점]



[보기] Data Flow Diagram, State Diagram, Class Diagram, Use-case Templete, Entity Relationship Diagram, Data object Table, Sequence Diagram, Processing Narratives, Swimlane Diagram, Activity Diagram, CRC Model, Collaboration Diagram, Use-case Diagram

8. 다음 ERD그림의 의미를 설명하시오. [6점]



9. CRC model이란? [3점]
㉮ Case-Reaction-Conclusion
㉯ Collaboration-Representation-Cohesion
㉰ Code-Retrieve-Correction
㉱ Class-Responsibility-Collaborator

다음 보기에서 답을 고르시오 (10번 - 14번 문제)
[보기] Modularity, Information hiding, Abstraction, Architecture, Functional Independence, Pattern, Refinement, Refactoring

10. 하위레벨의 상세한 솔루션 중심에서 상위레벨의 개략적 문제 중심으로의 표현 [3점]

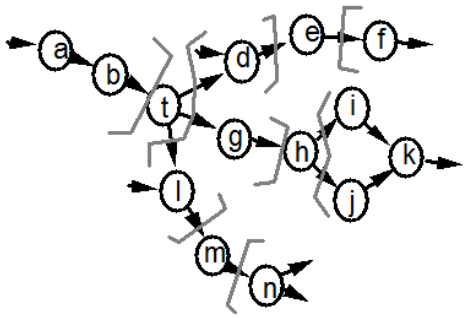
11. 프로그램 컴포넌트의 구조 또는 조직 [3점]

12. 소프트웨어를 독립적 컴포넌트로 나누는 것 [3점]

13. 오류파급효과의 감소를 위해 타 모듈로부터의 정보사용을 제한하는 것 [3점]

14. Cohesion과 Coupling에 의해 정의되는 모듈의 척도 [3점]

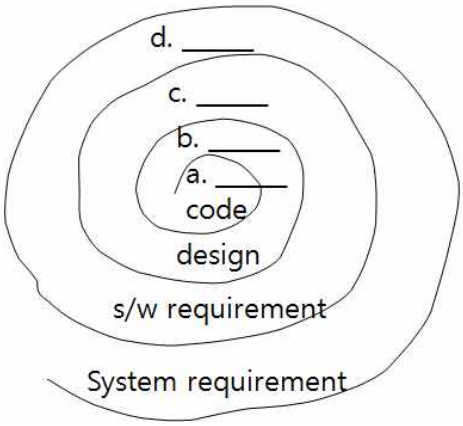
15. 다음 Transaction flow를 갖는 DFD에 대한 소프트웨어 구조를 그리시오. [10점]

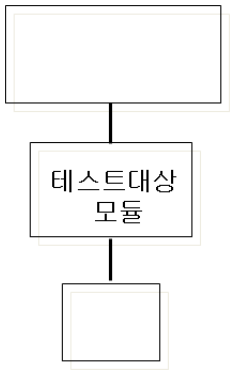


16. User Interface 설계를 위한 황금 규칙이 아닌 것은? [3점]
㉮ Make the interface consistent. ㉯ educe the user's memory load.
㉰ Speed up the execution time. ㉱ Place the user in control.

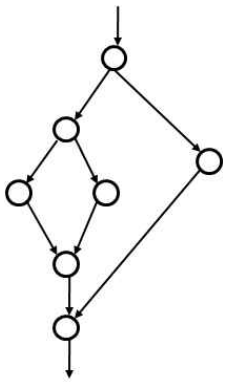
17. 아래의 그림에 들어갈 적합한 테스팅 전략을 작성하시오. [6점]

18. 아래의 단위 테스트에 필요한 상/하위 모듈의 이름을 각각 쓰시오. [5점]





19. 다음의 Flow graph에서 Basis path testing을 위한 독립적 실행경로는 몇 개인가? [6점]



20. 재공학의 절차로 적절한 것은? [3점]
㉮ Inventory Analysis - Reverse Engr. - S/W Restructuring - Forward Engr.
㉯ Reverse Engr. - Inventory Analysis - S/W Restructuring - Forward Engr.
㉰ Reverse Engr. - Forward Engr. - Inventory Analysis - S/W Restructuring
㉱ Inventory Analysis - S/W Restructuring - Reverse Engr. - Forward Engr.

다음 용어의 의미를 설명하시오.

21. Kow the user, know the tasks. [5점]

22. Keep it simple, stupid! [5점]

23. To see the forest for the tree [5점]

2012학년도 2학기 소프트웨어공학 기말고사 문제지 및 답안지

학번: 이름:

1. 소프트웨어 구성원의 세부 절차 설계 ==>

2. 구성원간의 관계성 설계 ==>

3. 소프트웨어와 시스템 그리고 사람간의 통신방법 설계 ==>

4. 구현중심의 하위계층에서 문제표현 중심의 상위계층으로의 단계적 설계 개념 ==>

5. 프로그램 구성원간의 구조화 개념 ==>

6. 복잡한 문제를 작은 문제들로 나누어 해결한 뒤, 통합하는 설계 전략 ==>

7. 소프트웨어를 하나의 통합된 모듈로 구성하는 설계 개념 ==>

8. 모듈의 주요 사항들을 가급적 외부로부터 감추려는 설계 기법 ==>

9. Cohesion과 coupling으로 규정될 수 있는 설계 개념 ==>

10. 하향식으로 상세화하는 설계 전략 ==>

11. 설계 검토를 통해 설계 구조를 재구성하는 설계 개념 ==>

12. 아키텍처 설계와 동의어 ==>

13. 데이터 저장소를 중심에 두고 여러 모듈들이 이를 활용하는 방식의 아키텍처 ==>

15. 주프로그램이 부프로그램들을 제어하는 방식의 아키텍처 ==>

16. DFD를 분석하여 incoming flow와 outgoing flow의 경계를 나누는 것을 통해 아키텍처를 설계해나가는 설계 단계 ==>

17. Functional, Layer, Communicational 등등으로 세분화되는 컴포넌트 설계의 척도 ==>

18. Content, Common, Control 등등으로 세분화되는 컴포넌트 설계의 척도 ==>

19. 테이블, 수도코드, 플로우차트 등을 이용하는 설계 ==>

20. 사용자 인터페이스의 설계과정에 가장 적합한 소프트웨어공학 모델
==>

21. 소프트웨어개발 단계별 과정에서 에러의 발생 및 발견을 도식화한 모델
==>

22. 소프트웨어 품질 보증을 위한 점검 회의 ==>

23. V&V에서 논리성 검증을 의미하는 것 ==>

24. 에러의 증상을 발견하는 것 ==>

25. 에러의 증상에 대한 원인을 규명하는 것 ==>

26. 소프트웨어 아키텍처에 초점을 맞춘 테스팅 ==>

27. 요구사항과의 일치성 여부에 초점을 맞춘 테스팅 ==>

28. 소프트웨어와 타 시스템 요소들과의 통합 테스팅 ==>

29. 소프트웨어 테스팅시 하위 모듈들을 대체하는 임시 모듈 ==>

30. 소프트웨어 테스팅시 상위의 메인 모듈의 역할을 담당하는 임시 모듈
==>

31. Stub를 필요로 하는 Integration testing 기법 ==>

32. 독립적 클래스부터 시작하여, 의존적 클래스로 확장해 가는 객체지향 Integration testing 기법 ==>

33. 사용자의 사이트에서 사용자에 의해 테스트되는 Validation test
==>

34. 시스템의 비정상적 운용에 대해 감내할 수 있는 정도를 측정하는 테스팅
==>

35. 에러의 원인에 대한 가설을 세운 뒤, 이를 확인해 나가는 디거깅 방식
==>

36. 소프트웨어 인터페이스상에서 시행되며, 기능적 측면을 중심으로 테스팅하는 기법 ==>

37. White-box 테스팅 방법의 하나로 flow graph를 활용하는 기법
==>

38. Black-box 테스팅 방법의 하나로 입력데이터를 몇 개의 주요 클래스들로 분류하여 테스트하는 기법 ==>

39. Box structure 명세를 이용한 엄격한 소프트웨어 개발 방식 ==>

40. 철저한 인증 절차를 통해 확인하고 보관하는 소프트웨어 형상 관리 개념
==>

보기

transform mapping, top-down integration testing, bottom-up integration testing, regression testing, transaction mapping, data-centered architecture, component-level design, milestone, system testing, validation testing, integration testing, unit testing, architecture, information hiding, waterfall model, incremental model, prototyping model, defect amplification model, cause elimination, refactoring, security testing, stress testing, performance testing, thread-based testing, use-based testing, cluster testing, alpha testing, beta testing, recovery testing, class testing, spiral model, pattern, data-flow architecture, baseline, formal method, cleanroom software engineering, call-and-return architecture, divide-and-conquer strategy, testing, debugging, formal technical review, modularity, stub, data-flow architecture, abstraction, boundary value analysis, monolithic, layered architecture, equivalence partitioning, brute force, basis path testing, driver, verification, validation, white-box testing, data/class design, functional independence, black-box testing, backtracking, coupling, cohesion, interface design, architecture design, refinement, preliminary design

2013학년도 1학기 소프트웨어공학 기말고사 문제지 및 답안지

학번: **이름:**

(객관식 2점, 주관식 3점, 총 95점 만점)

1. main focus of requirement modeling is on _____ not _____
① what ② how ③ where ④ who

2. 요구사항 분석의 경험 법칙이 아닌 것은?
① visible해야 한다
② information, function, behavior 영역에 대한 통찰을 제공해야 한다.
③ 가급적 세밀하고 정교한 모델을 구축해야 한다.
④ coupling을 최소화해야 한다.
⑤ 구현관점은 배제해야 한다.

3. 5가지 요구사항 분석 모델 (Elements of analysis model)을 나열하고, 구체적 model이나 diagram의 명칭을 제시하시오. (과제 제출물 참조)

4. Use Case Template 에 들어갈 항목과 거리가 먼 것은?
① level of abstraction ② exceptions ③ priority
④ goal ⑤ scenario

5. UML은 무엇의 약자인가?
① Uniformed Model Library
② Uniformed Modeling Language
③ Unified Model Library
④ Unified Modeling Language

6. Swimlane diagram은 다음 어느 diagram의 변형인가?
① activity diagram ② collaboration diagram

③ sequence diagram ④ deployment diagram
⑤ state transition diagram

7. ERD는 무엇의 약자인가?
① Element Requirement diagram ② Element Relationship diagram
③ Entity Requirement diagram ④ Entity Relationship diagram

8. CRC Model index card는 무엇의 약자인가?
① Class-Requirement-Component ② Class-Requirement-Collaborator
③ Class-Responsibility-Component ④ Class-Responsibility-Collaborator

9. DFD는 무엇의 약자인가?
① data flow diagram ② data function diagram
③ direct flow diagram ④ direct function diagram

10. DFD상에 나타나지 않는 도형은?
① 화살표 ② 직사각형 ③ 원 ④ 마름모 ⑤ 두줄 평행선

11. DFD상의 최하위 버블들에 대한 처리과정을 설명하는 모델은?
① CSPEC ② PSPEC ③ Component ④ Module

12. State diagram과 Sequence diagram과의 차이점을 논하시오.

13. Design 모델 네 종류를 나열하시오. (과제 제출물 참조)

다음 질문들에 적합한 설계 개념들을 보기에서 찾아 쓰시오. (문제14 ~ 19)

14. 문제 중심의 개략적 표현에서 해결방안 중심의 상세한 표현 간의 관계성

15. Structure 또는 organization of program module

16. S/W를 단일 속성을 갖는 컴포넌트로 나누는 개념

17. 다음 모듈들로부터 해당모듈의 주요 사항들을 은폐시키는 개념

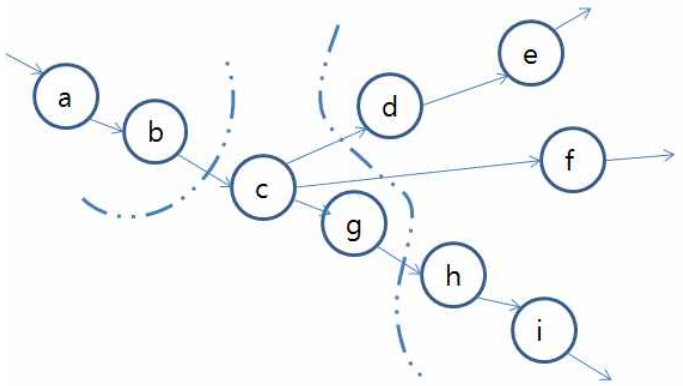
18. Cohesion과 coupling에 의해 정의되는 모듈성의 개념

19. Elaboration이라고도 하며 Abstraction과 보완 관계를 갖는 개념

보기: Architecture, Modularity, Pattern, Information hiding, Functional Independence, Refactoring, Refinement, Abstraction

20. Architecture style과 거리가 먼 것은?
① Call-and-return Arch. ② Data-flow Arch. ③ Data-centered Arch.
④ Layered Arch. ⑤ Spiral Arch.

21. 다음의 transition mapping을 통한 first-iteration architecture를 그리시오.



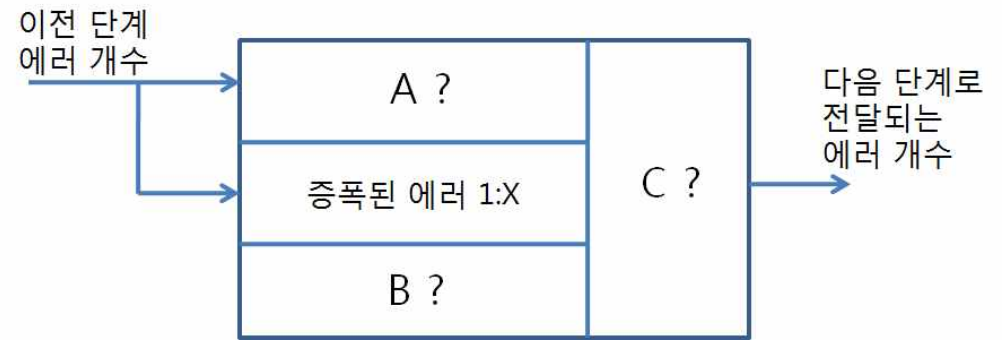
22. 가장 바람직한 coupling 수준은?
① common coupling ② content coupling ③ type use coupling

23. 가장 바람직한 cohesion 수준은?
① layer cohesion ② functional cohesion ③ communicational cohesion

24. PDL은 무엇의 약자인가?
① Project design library ② Program design library
③ Project design language ④ Program design language

25. User Interface 설계를 위한 황금 법칙과 거리가 먼 것은?
① Keep it simple ② Place the user in control
③ Reduce user's memory load ④ Make the interface consistent

26. 다음 Defect amplification model에서 A, B, C는 각각 무엇인가?



27. FTR은 무엇의 약자인가?
① Formal Technical Requirement ② Functional Technical Requirement

③ Formal Technical Review ④ Functional Technical Review

28. Testing과 Debugging의 관계에 대하여 논하시오.

29. Verification과 Validation의 차이점에 관해 논하시오.

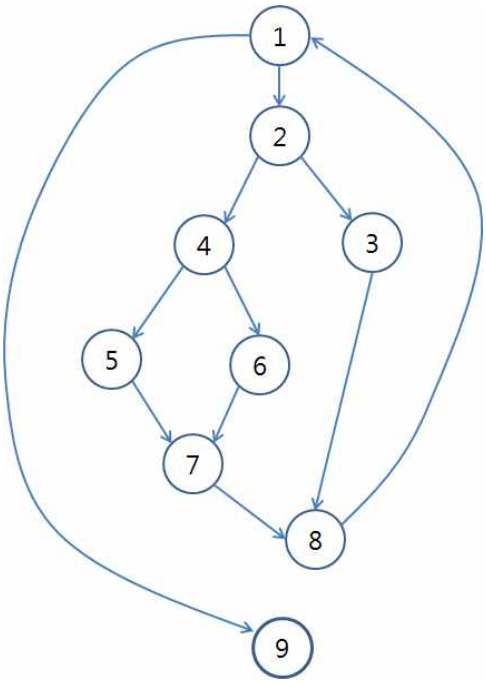
30. Testing 전략 네 가지를 나열하고 각각 간략히 설명하시오.

31. Driver와 Stub에 대해 각각 설명하시오.

32. Top-down과 Bottom-up integration testing의 장단점을 간단히 논하시오.

33. White-box testing과 black-box testing을 각각 논하시오.

34. White-box testing을 위해 Basis path testing을 하고자 한다. 코드분석 결과 다음의 flow graph를 얻었다. Test case를 위한 최소 독립경로는 무엇인가?



35. Equivalence Partitioning과 Bound value analysis를 통해 Black-box testing을 하고자 한다. 입력데이터는 0 ~ 200 사이의 혈압값이며, 처리결과 80이하는 저혈압, 80 ~ 120은 정상, 120 이상은 고혈압으로 출력된다. 적절한 입력 test case 12개를 쓰시오.

36. Cleanroom Process model에서 box structure specification에 해당되지 않는 것은?

① black-box ② white-box ③ state-box ④ clear-box

37. 소프트웨어 형상 관리를 위한 baseline 개념을 간략히 설명하시오.

38. data, information, knowledge, wisdom 개념을 간략히 설명하시오.

2014학년도 1학기 소프트웨어공학 기말고사 문제지 및 답안지

학번: **이름:**

[기본 문제 각 2점]

1. DFD는 무엇의 약자인가?

① direct function diagram ② data function diagram
③ direct flow diagram ④ data flow diagram

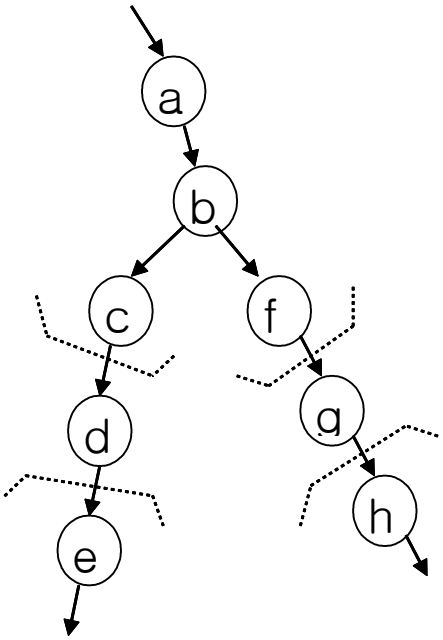
2. DFD상에 나타나지 않는 도형은?

① 화살표 ② 직사각형 ③ 마름모 ④ 원 ⑤ 평행선

3. DFD상의 최상위 레벨은?

① CSPEC ② PSPEC ③ Component ④ Context diagram ⑤ Driver

4. 다음의 Transaction flow mapping을 통한 first-iteration S/W Architecture를 그리시오. [4점]



5. State diagram 상에서 나타나지 않는 정보는?

① State ② Input ③ Output ④ Transition ⑤ External entity

6. SW 설계 모델에 해당되지 않는 것은?

① Data/class design ② Pattern design ③ Architecture design
④ Interface design ⑤ Component-level design

7. 다음 설계 개념들과 주요 특성을 연결하시오. [각 2점]

㉮ Abstraction	divide & conquer
㉯ Architecture	organization of components
㉰ Modularity	suppress low-level detail
㉱ Information hiding	re-organization
㉲ Refinement	elaboration
㉳ Functional independence	cohesion/coupling
㉴ Refactoring	reduce error propagation

8. 정형기술검토회의의 역할과 거리가 먼 것은?

① 오류를 발견한다.
② 설계 모델을 완성한다.
③ 요구사항을 충족시키는지 확인한다.
④ 표준을 따라서 개발되는지 확인한다.

9. 소프트웨어 시험에 관한 설명 중 적합하지 않은 것은?

① 오류의 50~65%는 설계단계에서 발생된다.
② 심리적으로 볼 때 건설적이라기보다는 파괴적인 것으로 인식된다.
③ 성공적인 시험은 잘 알려진 오류들을 찾아내는 것이다.
④ 시험을 통해 오류가 없다는 것을 완전히 보장할 수는 없다.

10. 좋은 시험에 대한 설명 중 잘못된 것은?

① 좋은 시험은 오류를 발견할 확률이 높다.
② 좋은 시험은 가급적 중복적이어야 한다.
③ 좋은 시험은 “best of breed(최고의 품종)”이어야 한다.
④ 좋은 시험은 너무 단순하지도 않고 너무 복잡하지도 않아야 한다.

11. 다음 중 Black-box testing을 통해 발견하기 힘든 오류는?

① 성능 오류들 ② 무결성 오류들
③ 인터페이스 오류들 ④ 실행경로상의 오류들

12. 통합 테스팅에 대한 접근방법이 아닌 것은?

① Top-down ② Bottom-up ③ Sandwich ④ Regression

13. 다음의 시험 중 시스템 실패에 대한 복구 능력 여부를 확인하기 위한 시험은?

① recovery testing ② security testing
③ stress testing ④ performance testing

14. 디버깅에 대한 설명 중 잘못된 것은?
① 디버깅은 성공적인 시험의 결과로 나타난다.
② 디버깅은 원인과 증상을 연결시키는 지적 과정이다.
③ 디버깅이 힘든 이유는 심리적인 요소가 많이 관여하기 때문이다.
④ 디버깅에 대한 체계적인 접근은 아직까지 불가능하다.

15. 하향식 통합 테스팅에 대한 설명 중 적절한 것은?
① 테스팅이 용이하다.
② stub가 필요하다.
③ 전체 기능을 조기에 시험할 수 없다.
④ 마지막 모듈이 추가될 때까지 전체 프로그램의 모습은 존재하지 않는다.

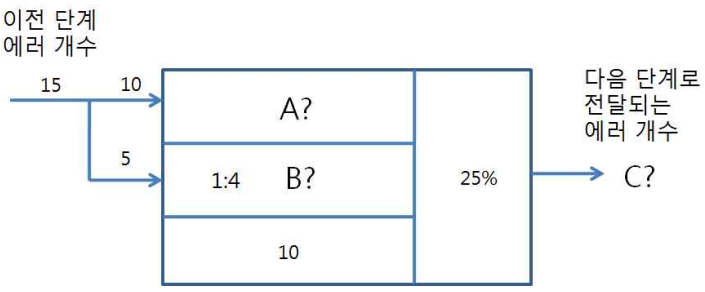
16. 다음의 객체지향 통합 테스팅 전략 중 하나의 입력이나 이벤트 처리에 필요한 클래스들을 엮어가는 방법은?
① class testing ② cluster testing
③ use-based testing ④ thread-based testing

17. 모듈의 독립성 관점에서 최악의 coupling 수준은?
① common coupling ② content coupling ③ type use coupling

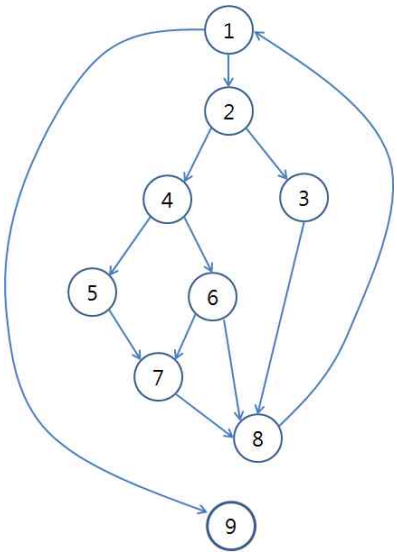
18. 모듈의 독립성 관점에서 최악의 cohesion 수준은?
① layer cohesion ② functional cohesion ③ unity cohesion

19. User Interface 설계를 위한 황금 법칙과 거리가 먼 것은?
① Make the interface consistent ② Place the user in control
③ Reduce user's memory load ④ Provide interaction with user

20. 다음 Defect amplification model에서 A, B, C는 각각 무엇인가? [각 2점]



21. White-box testing을 위해 Basis path testing을 하고자 한다. 소스코드분석 결과 다음의 flow graph를 얻었다. Test case 설계를 위한 최소 독립경로를 각각 나열하시오. [8점]



22. Equivalence Partitioning과 Bound value analysis를 통해 Black-box testing을 하고자 한다. 테스팅 대상 시스템은 15 ~ 25 사이의 적정 실내 온도를 제어하기 위한 SW 시스템이다. 따라서 입력데이터는 기온이다. 처리결과 15이하는 저온, 15 ~ 25은 정상, 25이상은 고온으로 출력된다. 적절한 test case 12개 정도를 나열하시오. [8점]

23. Cleanroom Process model에서 box structure specification에 적합하지 않은 용어는?
① black-box ② setup-box ③ state-box ④ clear-box

24. 소프트웨어 형상 관리와 거리가 먼 용어는?
① baseline ② CRC ③ FTR ④ SCM ⑤ SCIs

25. Debugging에 관한 설명 중 적절치 못한 것은?
① debugging은 testing의 후속작업으로 시행된다.
② 먼저 가설을 세운 뒤, 확인하는 방법을 cause elimination이라 한다.
③ symptom으로부터 원인을 추적하는 방법을 backtracking이라 한다.
④ debugging 솜씨는 후천적이다.

26. 다음 소프트웨어공학의 신개념들을 바르게 연결하시오. [각 2점]

㉮ Formal methods	High-quality S/W 개발 방법
㉯ Cleanroom S/W Engineering	수학적 명세를 통한 방법
㉰ Reengineering	보수유지성 향상을 위한 방법

27. V&V 개념 중 소프트웨어가 주어진 기능들을 올바르게 구현하였는지에 대한 논리적 검증을 ㉮_____라 하고, 소프트웨어가 고객의 요구사항들을 빠짐없이 반영하였는지에 대한 유효성 검증을 ㉯_____라 한다. [각 2점]

28. 다음의 개발단계와 테스팅 단계 간의 연관 관계를 완성하시오. [각 2점]

㉮ System Engineering	unit testing
㉯ Requirement	integration testing
㉰ Design	validation testing
㉱ Code	system testing