1. 소프트웨어 개발 방법론

: 과거 경험을 토대로 성공적으로 평가되는 소프트웨어를 분석 및 설계방법들을 모아 하나의 개발 방법으로 정형화 한 것
- 구조적 개발 방법론, 객체 지향 개발 방법론

1. 구조적 개발 방법론

: 개발 순서 : 요구사항 분석 -> 설계 -> 구현 -> 검사 -> 디버깅 -> 유지보수

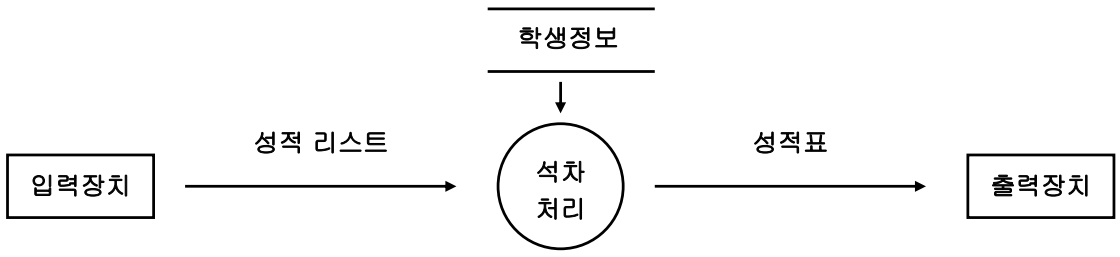
1. 요구사항 분석 ★★☆☆☆

1) 요구사항 분석 기법 : 사용자 면접, 현재 사용 중인 문서 검토, 설문 조사를 통한 의견 수렴

2) 분석가가 갖추어야 할 가장 중요한 능력
- 거시적 관점에서 세부적인 요소를 관찰할 수 있는 능력 (가장 중요)

3) 구조적 분석 기법(도구)
- 자료의 흐름과 처리를 중심으로 하는 요구사항 분석 방법
- 종류 : 자료 흐름도, 자료 사전, 소단위 명세서, 개체 관계도, 상태 전이도

2. 자료흐름도 (DFD : Data Flow Diagram) ★



1) 기호와 의미

① 단말 : 자료의 출처, 도착지

② 자료 이름 → : 자료의 흐름

③ 처리 : 프로세스, 기능, 변환, 버블

④ 자료 저장소

2) 특징
- 시스템내의 모든 자료 흐름은 4가지의 기본 기호로 표시된다.
- 각 각의 변환(처리)에 대하여 개별적인 상세화가 가능하다
- 자료는 처리를 거쳐 변환될 때마다 새로운 명칭을 부여해야 한다
- 자료흐름도의 최하위 처리(process)는 소단위명세서를 갖는다.
- 어떤 처리(process)가 출력자료를 산출하기 위해서는 필요한 자료가 반드시 입력되어야 한다.
- 상위단계의 처리(Process)와 하위 자료흐름도의 자료 흐름은 서로 일치돼야 한다.
- Bubble Chart 라고도 부른다

3. 자료사전 (DD : Data Dictionary) ★★☆☆☆

1) 특징
- DFD에 있는 자료를 더 자세히 정의하고 기록한 것
- 데이터를 설명하는 데이터 (메타 데이터)

기출) 고객명세는 고객성명, 고객번호, 고객주소로 구성되어 있으며, 고객성명과 고객번호는 둘 중 하나만 선택이 가능함 => 고객명세 = [고객성명 | 고객번호] + 고객주소

2) 기호와 의미

기 호	의 미	기 호	의 미	기 호	의 미
=	정의	+	연결	[|]	선택
{ }	반복	* *	주석, 설명	( )	생략

정보처리기사 필기
합격보장!! 기사자격증 전문 최강! 최고! 사이트
[SE 3강]-구조적 개발 방법론

4. HIPO (Hierarchy Input Process Output) ★★★☆☆

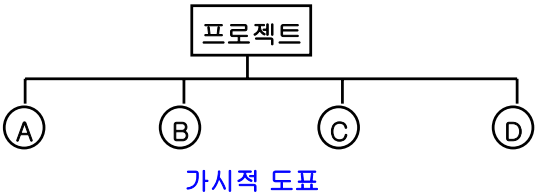
1) 특징
- 분석, 설계, 문서화에 사용되는 도구이며, 기본 시스템 모델은 입력, 처리, 출력으로 구성됨
- 하향식 소프트웨어 개발을 위한 문서화 도구로서 이해하기 쉬움
- 변경, 유지보수 용이

2) HIPO 종류

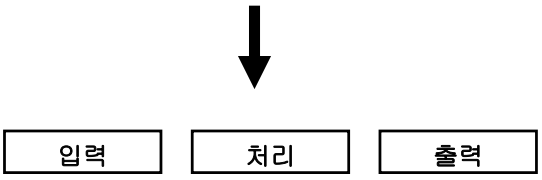
① 가시적 도표 (Visual Table of Contents) = 구조도
- 시스템 전체적인 기능과 흐름을 보여주는 계층 구조도

② 총체적 다이어그램 (Overview Diagram) = 개요 도표 집합
- 입력, 처리, 출력에 대한 전반적인 정보를 제공하는 도표

③ 세부적 다이어그램 (Detail Diagram) = 상세 도표 집합
- 총체적 다이어그램를 상세 기술하는 도표



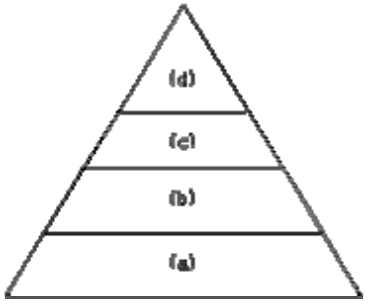
가시적 도표



총체적 다이어그램

1. 구조적 설계

1) 소프트웨어 설계 모형



(d) 절차 설계 : 모듈이 수행할 기능을 절차적 기술로 바꾸는 것

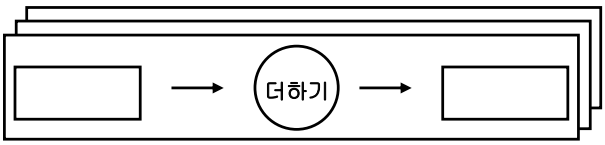
(c) 인터페이스 설계 : 시스템과 사용자가 어떻게 통신하는가

(b) 아키텍처(구조) 설계 : 모듈간의 관계와 프로그램 구조 정의

(a) 데이터 설계 : 요구사항분석단계에서 생성된 정보를 소프트웨어 구현하는데 필요한 자료구조로 변환하는 것

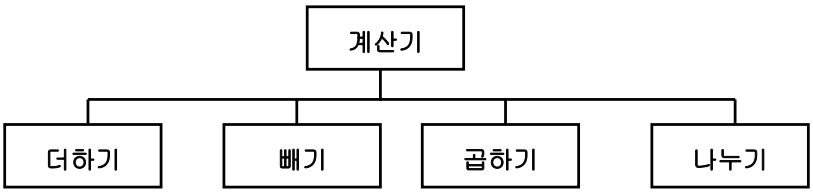
2) 분석 -> 설계 -> 구현 개념 이해하기 (ex. 계산기)

① 분석 자료 (DFD)

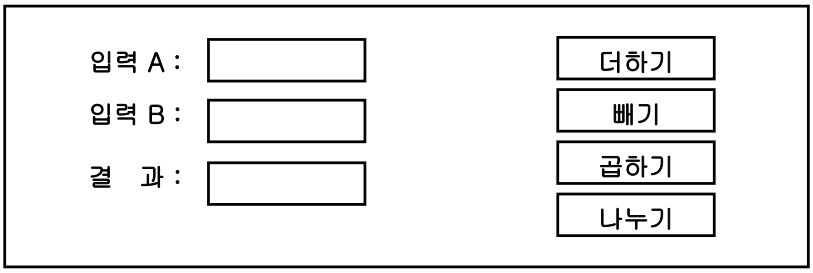


② 데이터 설계
- 입력 A, 입력 B, 결과,....

③ 구조 설계



④ 인터페이스 설계



⑤ 절차(프로시져) 설계

a = Text1	a = Text1	a = Text1	a = Text1
b = Text2	b = Text2	b = Text2	b = Text2
Text3 = a + b	Text3 = a - b	Text3 = a * b	Text3 = a / b

⑥ 구현

3) 설계의 기본 원리

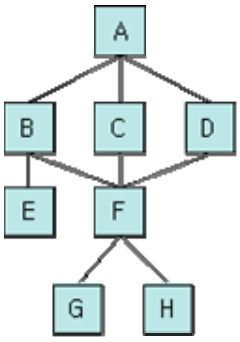
① 모듈화
- 소프트웨어를 모듈 단위로 나누는 것 (작업 단위, 소프트웨어 내의 프로그램, 부 시스템, 서브루틴)

② 추상화 : 전체적이고 포괄적인 개념을 설계한 후 세분화 구체화 시켜나가는 방법
- 추상화의 종류 : 기능 추상화, 제어 추상화, 자료 추상화

③ 정보 은닉 : 모듈 내부에 포함된 절차와 자료들의 정보를 숨겨서 다른 모듈이 접근하거나 변경하지 못하도록 하는 기법

④ 구조화
- 공유도(Fan-In) : 어떤 모듈을 제어(호출)하는 상위 모듈의 개수
- 제어도(Fan-out) : 어떤 모듈에 의해 제어(호출)되는 하위 모듈의 개수

기출) 다음은 프로그램 구조를 나타낸다.
모듈 F에서의 Fan-In과 Fan-Out의 수는 얼마인가?
(정답) Fan-In : 3, Fan-Out : 2



4) 좋은 설계 기준
- 설계는 모듈적이어야 함
- 설계는 자료와 프로시저에 대한 분명하고 분리된 표현을 포함
- 소프트웨어는 논리적으로 특별한 기능과 부기능을 수행하는 요소들로 나누어져야 한다.
- 소프트웨어 요소들 간의 효과적인 제어를 위해 설계에서 계층적 조직이 제시되어야 함

5) 자료 흐름 중심 설계

① 정보 흐름의 유형 설정 (데이터 설계)
② 흐름의 경계를 표시
③ 자료흐름도를 프로그램 구조로 사상 (구조 설계)
④ 제어 계층을 분해시켜서 정의 (절차 설계)
⑤ 경험적 방법으로 구체화

2. N-S 차트 (Nassi-Schneiderman Chart) ★★☆☆☆

- 절차 설계 기법
- 논리의 기술에 중점을 둔 도형을 이용한 표현 방법으로 박스 다이어그램이라고 함
- 순차(Sequence), 선택 및 다중 선택(If ~ then ~ else, Case), 반복(Repeat ~ until, While, for) 등의 제어 논리 구조를 표현

3. 모듈화 ★

1) 모듈화 목적
- 소프트웨어 복잡도가 감소하고, 변경이 쉬우며 프로그램 구현이 용이
- 개념 이해하기 : OSI 7계층 설명에서 자동화 부품화에 대한 개념
(각 단계는 독립적이며 상호 의존도는 낮아야 한다.)

2) 결합도 (Coupling) : 모듈 간에 상호 의존도
- 독립적인 모듈이 되기 위해서는 결합도가 약해야 함
- 종류 : 데이터 < 스탬프 < 제어 < 외부 < 공통 < 내용

① 데이터 결합도(Data) : 데이터 요소(파라미터,인수,매개변수)로만 구성된 경우
② 스탬프 결합도(Stamp) : 배열이나 레코드 등의 자료구조가 전달될 경우
③ 제어 결합도(Control) : 제어 요소가 전달된 경우
④ 외부 결합도(External) : 외부로 선언한 데이터(변수)를 참조할 경우
⑤ 공통 결합도(Common) : 공통 데이터 영역을 사용할 경우
⑥ 내용 결합도(Content) : 내부 기능 및 내부 자료를 참조할 경우

정보처리기사 필기
합격보장!! 기사자격증 전문 최강! 최고! 사이트
[SE 3강]-구조적 개발 방법론

3) 응집도 (Cohesion) : 모듈 안의 요소들이 서로 관련되어 있는 정도
- 모듈이 독립적인 기능으로 잘 정의되어 있는 정도
- 독립적인 모듈이 되기 위해서는 응집도가 강해야 함
- 종류 : 우연적 < 논리적 < 시간적 < 절차적 < 교환적 < 순차적 < 기능적

① 우연적 응집도(Coincidental) : 서로 관련 없는 요소로만 구성
② 논리적 응집도(Logical) : 유사한 성격 또는 처리 요소들로 구성
③ 시간적 응집도(Temporal) : 특정 시간에 처리되는 몇 개의 기능을 모아 구성
④ 절차적 응집도(Procedural) : 구성 요소들이 그 기능을 순차적으로 수행할 경우
⑤ 교환적 응집도(Communication) : 동일한 입력과 출력을 사용하여 서로 다른 기능을 수행하는 구성 요소들이 모였을 경우
⑥ 순환적,순차적 응집도(Sequential) : 출력 데이터를 그 다음 활동의 입력 데이터로 사용할 경우
⑦ 기능적 응집도(Functional) : 단일 문제와 연관되어 수행될 경우

1. 구현 ★★★☆☆

1) 정의 : 설계단계에서 생성된 내용을 컴퓨터가 알 수 있는 형태로 변환하는 과정 (코딩)

2) 프로그램 언어 선택 기준
- 대상 업무 성격, 개발 담당자의 경험과 지식, 과거의 개발 실적 등, 4세대 언어 여부 (X)

3) 구조적 프로그래밍 : 컴퓨터 프로그램을 여러 갈래로 분기하여 복잡하게 하지 않고, 순서대로, 선택적으로 반복 문장을 사용하는 제어구조만을 사용한 프로그램 (Dijkstra 제안)

1. 검사 (Test) ★

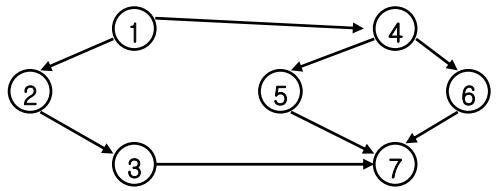
1) 소프트웨어 품질 보증 활동의 하나로써 오류를 발견하기 위하여 프로그램 수행하는 과정

2) 검사 기법
① 화이트 박스 테스트 : 구조 테스트
- 모듈 안의 작동을 자세히 관찰할 수 있으며, 프로그램 원시 코드의 논리적인 구조를 커버하도록 테스트 케이스를 설계하는 프로그램 테스트 방법
- 프로그램의 제어 구조에 따라 선택, 반복 등의 부분들을 수행함으로써 논리적 경로를 제어
- 모듈 안의 작동을 직접 관찰
- 원시 코드의 모든 문장을 한 번 이상 수행함
- 종류 : 기초 경로 검사(Basic Path Testing, McCabe 제안), 조건 검사(Condition Testing), 루프 검사(Loop Testing), 데이터 흐름 검사(Data Flow Testing)

② 블랙 박스 테스트 : 기능 테스트
- 소프트웨어가 수행할 특정 기능을 알기 위해서 각 기능이 완전히 작동되는 것을 입증하기 위한 검사
- 발견할 수 있는 오류 : 성능, 부정확한 기능, 인터페이스 오류, 논리 구조상의 오류 (X)
- 종류 : 동치분할검사(Equivalence Partitioning), 경계값 분석(Boundary Value Analysis), 원인-효과 그래프 검사(Cause-Effect Graphing Testing), 오류예측검사(Fault Based Testing), 비교검사(Comparison Testing)

정보처리기사 필기
합격보장!! 기사자격증 전문 최강! 최고! 사이트
[SE 3강]-구조적 개발 방법론

* McCabe 의 소프트웨어 복잡도 측정



기출) McCabe 방법에 의한 다음 그래프의 V(G)의 크기는?
- 정답 : 3

2. 검사 전략 ★★★★★

1) 검사 순서 : 단위(코드) -> 통합(설계) -> 검증(요구사항) -> 시스템

2) 단위 검사 : 모듈에 대한 검사 (화이트 박스 테스트 기법 사용)

3) 통합 검사 : 모듈들을 결합하여 검사
① 하향식 : 상위 모듈에서 하위 모듈 방향으로 통합하면서 검사하는 기법
- Stub 필요 : 모듈 간에 통합 시험을 하기 위해 일시적으로 제공되는 시험용 모듈
② 상향식 : 하위 모듈에서 상위 모듈 방향으로 통합하면서 검사하는 기법
- 절차 : 하위 모듈을 클러스터로 결합 -> 드라이버라는 제어 프로그램 작성 -> 클러스터 검사 -> 드라이버 제거하고 클러스터를 상위로 결합

4) 검증 검사 : 요구사항을 충족하는지 검사 (블랙 박스 테스트 기법 사용)
① 형상 검사
② 알파 검사 : 개발자의 장소에서 사용자가 시험하고 개발자는 뒤에서 결과를 지켜보는 검사
③ 베타 검사 : 실업무를 가지고 사용자가 직접 시험하는 검사

5) 시스템 검사 : 해당 컴퓨터 시스템에서 수행 되는지를 검사

3. 디버깅

: 오류 수정 과정 (검사 기법 X)
- 성공적인 테스팅의 결과로 발생
- 징후로부터 원인을 찾아 수정하는 과정
- 심리적인 요소가 많이 관여하기 때문에 힘듦
- 접근법 : 맹목적 강요, 역추적, 원인 제거

정보처리기사 필기
합격보장!! 기사자격증 전문 최강! 최고! 사이트
[SE 3강]-구조적 개발 방법론

1. 유지 보수 ★

: 가장 많은 비용이 투입되는 단계로써 인수, 설치된 후 발생하는 모든 공학적 작업

1) 유지보수 활동
- 수정 보수 (Corrective) : 오류 수정
- 적응 보수 (Adaptive) : 환경 변화(하드웨어, 운영체제 등) 반영
- 완전화 보수, 기능 보수 (Perfective) : 기능 개선, 가장 큰 비중 차지(Win98 -> Win 2000 -> Win XP)
- 예방 보수 (Preventive)

2) 유지보수 비용 계산식 ($M = P + Ke^{(c-d)}$)
- P : 생산적인 활동에 드는 비용, K : 통계값에서 구한 상수, c : 복잡도, d : 지식의 정도

3) 외계인 코드
- 아주 오래되어(15년 이상) 유지보수 작업이 어려운 프로그램 (방지법 : 문서화)

4) 유지보수 부작용
- 코딩 부작용 : 코딩 내용 변경에 따른 문제
- 자료 부작용 : 자료 구조 변경에 따른 문제
- 문서화 부작용 : 변경에 대한 내용이 문서에 적용되지 않을 경우

[SE 3강]-구조적 개발 방법론

1. 구조적 분석 도구와 거리가 먼 것은?

가. 자료 사전
나. 자료 흐름도
다. 프로그램 명세서
라. 소단위 명세서

2. 시스템 개발을 위한 첫 단계는 사용자의 요구나 시스템에 대한 분석이라고 할 수 있다. 이 중 사용자의 요구 분석을 위해 주로 사용하는 기법이 아닌 것은?

가. 사용자 면접
나. 현재 사용 중인 각종 문서 검토
다. 설문 조사를 통한 의견 수렴
라. 통제 및 보안 분석

3. 분석가(Analyst)가 갖추어야 할 능력 중 가장 중요한 것은?

가. 추상적인 개념을 파악하여 논리적인 구성요소로 분해할 수 있는 능력
나. 서로 상반되고 모호한 정보로부터 필요한 사항을 수렴할 수 있는 능력
다. 관련된 하드웨어와 소프트웨어에 관한 최신 기술
라. 거시적 관점에서 세부적인 요소를 관찰할 수 있는 능력

[정답] 1.다 2.라 3.라

정보처리기사 필기
합격보장!! 기사자격증 전문 최강! 최고! 사이트

[SE 3강]-구조적 개발 방법론

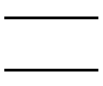
4. 자료 흐름도의 구성 요소가 아닌 것은?

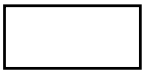
가. 소단위 명세서 나. 단말
다. 프로세스 라. 자료 저장소

5. 자료 흐름도(DFD)를 작성하는데 지침이 될 수 없는 항목은?

가. 자료흐름은 처리(Process)를 거쳐 변환 될 때마다 새로운 이름을 부여한다.
나. 어떤 처리(Process)가 출력자료를 산출하기 위해서는 반드시 입력 자료가 발생해야 한다.
다. 자료저장소에 입력 화살표가 있으면 반드시 출력 화살표도 표시되어야 한다.
라. 처리(Process)와 하위 자료흐름도의 자료 흐름은 서로 일치 되어야 한다.

6. 자료 흐름도(DFD, Data Flow Diagram)의 구성 요소 중 자료 출처와 도착지를 나타내는 기호는?

가.  나. 

다. 라. 

7. 자료 사전(Data Dictionary)에 사용되는 기호의 의미를 올바르게 나타낸 것으로 짝지어진 것은?

가. { } : 자료의 생략 가능, ( ) : 자료의 선택
나. ( ) : 자료의 설명, * * : 자료의 선택
다. = : 자료의 설명, * * : 자료의 정의
라. + : 자료의 연결, ( ) : 자료의 생략 가능

[정답] 4.가 5.다 6.라 7.라

[SE 3강]-구조적 개발 방법론

8. HIPO(Hierarchy Input Process Output)에 대한 설명으로 옳지 않은 것은?

가. HIPO 다이어그램에는 가시적 도표(Visual Table of Contents), 총체적 다이어그램(Overview Diagram), 세부적 다이어그램(Detail Diagram)의 세 종류가 있다.
나. 가시적 도표(Visual Table of Contents)는 시스템에 있는 어떤 특별한 기능을 담당하는 부분의 입력, 처리, 출력에 대한 전반적인 정보를 제공한다.
다. HIPO다이어그램은 분석 및 설계 도구로서 사용된다.
라. HIPO는 시스템의 설계나 시스템 문서화용으로 사용되고 있는 기법이며, 기본 시스템 모델은 입력, 처리,출력으로 구성된다.

9. 프로그램을 구성하는 기능을 기술한 것으로 입력, 처리, 출력을 기술하는 HIPO패키지에 해당하는 것은?

가. Overview Diagram
나. Detail Diagram
다. Visual Table of contents
라. Index Diagram

10. 데이터 흐름도(DFD)의 구성요소에 포함되지 않는 것은?

가. 처리 공정(process)
나. 자료 흐름(data flow)
다. 자료 사전(data dictionary)
라. 자료 저장소(data store)

11. 자료 사전(Data Dictionary)에서 반복을 의미하는 기호는?

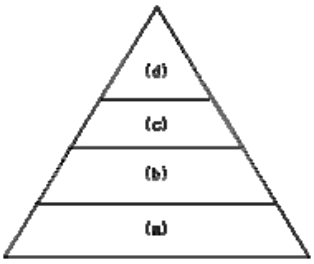
가. =　　나. { }　　다. +　　라. ( )

[정답] 8.나 9.가 10.다 11.나

정보처리기사 필기
합격보장!! 기사자격증 전문 최강! 최고! 사이트

[SE 3강]-구조적 개발 방법론

12. 다음은 소프트웨어 설계 모형의 구조도이다. (a), (b), (c), (d)에 들어갈 항목을 순서대로 나열한 것은?



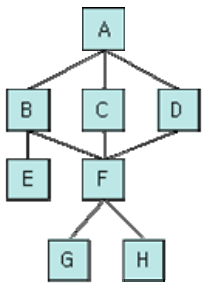
가. 데이터 설계-아키텍처 설계-절차 설계-인터페이스 설계
나. 아키텍처 설계-데이터 설계-절차 설계-인터페이스 설계
다. 아키텍처 설계-데이터 설계-인터페이스 설계-절차 설계
라. 데이터 설계-아키텍처 설계-인터페이스 설계-절차 설계

13. 설계 품질을 평가하기 위해서는 반드시 좋은 설계에 대한 기준을 세워야 한다. 다음 중 좋은 기준이라고 할 수 없는 것은?

가. 설계는 모듈적 이어야 한다.
나. 설계는 자료와 프로시저에 대한 분명하고 분리된 표현을 포함해야 한다.
다. 소프트웨어 요소들 간의 효과적인 제어를 위해 설계에서 계층적 조직이 제시 되어야 한다.
라. 설계는 서브루틴이나 프로시저가 전체적이고 통합적이 될 수 있도록 유도되어야 한다.

14. 다음은 프로그램 구조를 나타낸다. 모듈 F에서의 Fan-In과 Fan-Out의 수는 얼마인가?

가. Fan-In : 2, Fan-Out : 3
나. Fan-In : 3, Fan-Out : 2
다. Fan-In : 1, Fan-Out : 2
라. Fan-In : 2, Fan-Out : 1



[정답] 12.라 13.라 14.나

[SE 3강]-구조적 개발 방법론

15. 결합도(Coupling)가 강한 순서대로 옳게 나열된 것은?

가. 내용 결합도 > 공통 결합도 > 제어 결합도 > 스탬프 결합도 > 데이터 결합도
나. 공통 결합도 > 내용 결합도 > 제어 결합도 > 데이터 결합도 > 스탬프 결합도
다. 데이터 결합도 > 내용 결합도 > 제어 결합도 > 공통 결합도 > 스탬프 결합도
라. 공통 결합도 > 내용 결합도 > 제어 결합도 > 스탬프 결합도 > 데이터 결합도

16. 시스템을 설계할 때 필요한 설계 지침으로 두 모듈 간의 의존도를 나타내는 것은?

가. 결합도 나. 응집도 다. 신뢰도 라. 종합도

17. 데이터 설계에 있어서 응집도(Cohesion)의 의미로 가장 적절한 것은?

가. 데이터 구조들이 시스템 전반에 얼마나 연관 관계를 가지고 있는가 하는 정도
나. 모듈이 개발 단계 별로 얼마나 잘 정의되어 있는가 하는 정도
다. 모듈이 독립적인 기능으로 잘 정의되어 있는 정도
라. 모듈들 간의 상호 연관성의 정도

18. 응집도가 강한 것부터 약한 순서로 옳게 나열된 것은?

가. Sequential→Functional→Procedural→Coincidental →Logical
나. Procedural→Coincidental→Functional→Sequential →Logical
다. Functional→Sequential→Procedural→Logical →Coincidental
라. Logical→Coincidental→Functional→Sequential →Procedural

[정답] 15.가 16.가 17.다 18.다

정보처리기사 필기
합격보장!! 기사자격증 전문 최강! 최고! 사이트

[SE 3강]-구조적 개발 방법론

19. 효과적인 모듈화 설계 방안이 아닌 것은 어느 것인가?

가. 응집도를 높인다.
나. 결합도를 낮춘다.
다. 복잡도와 중복을 피한다.
라. 모듈의 기능은 예측 불가능하도록 정의한다

20. 자료 흐름 중심 설계 절차를 올바른 순서로 나열한 것은?

1. 자료 흐름도를 프로그램 구조로 사상한다.
2. 흐름의 경계를 표시한다.
3. 정보 흐름의 유형을 설정한다.
4. 제어 계층을 분해(Factoring)시켜서 정의한다.
5. 경험적 방법으로 구체화시킨다.

가. 1-2-3-4-5 나. 3-2-1-4-5
다. 4-5-3-2-1 라. 4-5-1-2-3

21. N-S(Nassi-Schneiderman) Chart에 대한 설명으로 거리가 먼 것은?

가. 논리의 기술에 중점을 둔 도형식 표현 방법이다.
나. 연속, 선택 및 다중 선택, 반복 등의 제어 논리 구조로 표현한다.
다. 주로 화살표를 사용하여 논리적인 제어 구조로 흐름을 표현한다.
라. 조건이 복합되어 있는 곳의 처리를 시각적으로 명확히 식별하는데 적합하다.

22. 나씨-슈나이더만(Nassi-Schneiderman)도표는 구조적 프로그램을 표현하기 위해 고안되었다. 이 방법에서 알고리즘의 제어구조는 3가지로 충분히 표현될 수 있는데 이에 속하지 않는 것은 어느 것인가?

가. 선택, 다중 선택(If~then~else, Case)
나. 반복(Repeat~until, While, for)
다. 분기(Goto, Return)
라. 순차(Sequence)

[정답] 19.라 20.나 21.다 22.다

[SE 3강]-구조적 개발 방법론

23. 소프트웨어 개발 방법론에서 구현(Implementation)에 대한 설명으로 가장 적절한 것은?

가. 요구사항 분석 과정 중 모아진 요구사항을 옮기는 것
나. 시스템이 무슨 기능을 수행하는지에 대한 시스템의 목표 기술
다. 프로그래밍 또는 코딩이라고 불리며 설계 명세서가 컴퓨터가 알 수 있는 모습으로 변환되는 과정
라. 시스템이나 소프트웨어 요구사항을 정의하는 과정

24. 구조적 프로그래밍에서 사용하는 기본적인 제어 구조에 해당하지 않는 것은?

가. 순차(Sequence) 나. 반복(Iteration)
다. 호출(Call) 라. 선택(Selection)

25. 블랙 박스 검사에 해당하지 않는 것은?

가. 데이터 흐름 검사(Data Flow Testing)
나. 동치 분할 검사(Equivalence Partitioning Testing)
다. 원인 효과 그래픽 기법(Cause-effect Graphing-testing)
라. 비교 검사(Comparison Testing)

26. 화이트 박스 시험(White Box Text)의 설명으로 옳지 않은 것은 어느 것인가?

가. 프로그램의 제어 구조에 따라 선택, 반복 등의 부분들을 수행함으로써 논리적 경로를 제어한다.
나. 모듈 안의 작동을 직접 관찰할 수 있다.
다. 소프트웨어 산물의 각 기능별로 적절한 정보 영역을 정하여 적합한 입력에 대한 출력의 정확성을 점검한다.
라. 원시 코드의 모든 문장을 한 번 이상 수행함으로써 수행된다.

27. 제품이 수행할 특정 기능을 알기 위해서 각 기능이 완전히 작동되는 것을 입증하는 검사로서, 기능 검사라고도 하는 것은?

가. 블랙 박스 검사 나. 그린 박스 검사
다. 블루 박스 검사 라. 화이트 박스 검사

28. 소프트웨어의 시험 중 화이트 박스 시험의 과정이 아닌 것은?

가. 조건 테스트 나. 모든 실행문 테스트
다. 경계 값 분석 라. 분기점 테스트

[정답] 23.다 24.다 25.가 26.다 27.가 28.다

정보처리기사 필기
합격보장!! 기사자격증 전문 최강! 최고! 사이트

[SE 3강]-구조적 개발 방법론

29. 소프트웨어 검사 단계를 올바른 순서로 나열한 것은?

㉠ 설계 검사	㉡ 요구사항 검사
㉢ 코드 검사	㉣ 시스템 검사

가. ㉠ → ㉡ → ㉢ → ㉣
나. ㉢ → ㉠ → ㉡ → ㉣
다. ㉡ → ㉢ → ㉣ → ㉠
라. ㉡ → ㉣ → ㉢ → ㉠

30. 하향식 통합에 있어서 모듈 간의 통합 시험을 하기 위해 일시적으로 필요한 조건만을 가지고 임시로 제공되는 시험용 모듈을 무엇이라고 하는가?

가. Driver 나. Stub
다. Sub-Program 라. Dummy-Program

31. 화이트 박스(WHITE BOX) 테스트 기법이 아닌 것은?

가. 데이터 흐름 검사(DATA FLOW TEST)
나. 루프 검사(LOOP TEST)
다. 기초 경로 검사(BASIC PATH TEST)
라. 동치 분할 검사(EQUIVALENCE PARTITIONING TEST)

32. 상향식 통합 테스트(Bottom - up Integration Test)의 과정이 옳게 나열된 것은?

① 드라이버라는 제어 프로그램의 작성
② 낮은 수준의 모듈들을 클러스터로 결합
③ 클러스터의 검사
④ 드라이버를 제거하고 클러스터를 상위로 결합

가. ① → ② → ③ → ④
나. ② → ① → ③ → ④
다. ② → ③ → ① → ④
라. ① → ② → ④ → ③

33. 디버깅에 대한 설명 중 거리가 먼 것은?

가. 디버깅은 성공적인 테스팅의 결과로 발생한다.
나. 디버깅은 징후로부터 원인을 찾아 수정하는 과정이다.
다. 디버깅이 힘든 이유는 심리적인 요소가 많이 관여하기 때문이다.
라. 디버깅에 대한 체계적인 접근은 아직까지 제안되고 있지 않다.

[정답] 29.나 30.나 31.라 32.나 33.라

[SE 3강]-구조적 개발 방법론

34. 소프트웨어 유지보수에 관련된 설명으로 옳지 않은 것은?

가. 유지보수는 소프트웨어가 인수, 설치된 후 발생하는 모든 공학적 작업을 말한다.
나. 유지보수는 원인에 따라 수리(Corrective) 보수, 적응(Adaptive)보수, 완전화(Perfective) 보수, 예방(Preventive)보수 등이 있다.
다. 소프트웨어에 가해지는 변경을 제어 관리하는 것을 형상 관리(Configuration Management)라고 한다.
라. 소프트웨어 비용 중 유지보수 비용은 개발비용보다 적다.

35. 소프트웨어 유지보수의 유형에 해당하지 않는 것은?

가. 수정 보수(Corrective Maintenance)
나. 기능 보수(Functional Maintenance)
다. 완전화 보수(Perfective Maintenance)
라. 예방 보수(Preventive Maintenance)

36. 유지보수(Maintenance) 작업의 분류상 가장 큰 비중(업무량 및 비용)을 차지하는 부분은?

가. 교정 정비(Corrective Maintenance)
나. 조정 정비(Adaptive Maintenance)
다. 예방 정비(Preventive Maintenance)
라. 완전 정비(Perfective Maintenance)

37. 외계인 코드(Alien Code)에 대한 설명으로 옳은 것은?

가. 프로그램의 로직이 복잡하여 이해하기 어려운 프로그램을 의미한다.
나. 아주 오래되어(15년 정도 이상) 유지보수 작업이 어려운 프로그램을 의미한다.
다. 오류(Error)가 없이 완벽하게 수정된 프로그램을 의미한다.
라. 4세대 언어로 사용자가 직접 작성한 프로그램을 의미한다.

38. 소프트웨어 유지보수의 부작용 중 자료코드에 대한 변경이 설계문서나 사용자가 사용하는 매뉴얼에 적용되지 않을 때에 발생하는 부작용은 무엇인가?

가. 코딩 부작용　　나. 자료 부작용
다. 문서화 부작용　　라. 유지보수 부작용

[정답] 34.라 35.나 36.라 37.나 38.다