1 소프트웨어 공학의 개념

1.1 소프트웨어와 시스템

(1) 소프트웨어(Software)

1) 소프트웨어는 하드웨어를 동작시켜 사용자가 작업을 편리하게 수행하도록 하는 프로그램과 자료 구조 등을 총칭함

2) 프로그램들의 사용과 운영을 기술하는 문서까지 포함

(2) 소프트웨어의 특성

|오|답|쪽|지|
공학적으로 잘 작성된 소프트웨어의 특성 오답
✗ 소프트웨어는 편리성이나 유지보수성에 점차 비중을 적게 두는 경향이 있음
✗ 프로그램이 독창적이어야 함

1) 공학적으로 잘 작성된 소프트웨어(=좋은 소프트웨어)의 특성 9904 0405 0703 0709
   ① 사용자가 원하는 대로 동작해야 함
   ② 일정 시간 내에 주어진 조건하에서 원하는 기능을 실행할 수 있어야 함
   ③ 신뢰성이 높아야 하며 효율적이어야 함
   ④ 잠재적인 에러가 가능한 적어야 하며, 유지보수가 용이해야 함
   ⑤ 적당한 사용자 인터페이스 제공으로 사용하기가 편리해야 함
   ⑥ 남이 알아보기 쉬워야 함
   ⑦ 경제적이어야 함
   ⑧ 문서화가 잘 되어 있어야 함
   ⑨ 사용자의 기능 변경의 필요성을 만족하기 위하여 소프트웨어를 진화하는 것이 가능해야 함(유지보수성)
   ⑩ 소프트웨어가 자원을 쓸데없이 낭비하지 않아야 함(효율성)
   ⑪ 소프트웨어는 적절한 사용자 인터페이스와 문서를 가지고 있어야 함 (사용 용이성)

2) 소프트웨어의 생산성 (Productivity)
   ① 소프트웨어의 생산성은 투입된 비용, 노력 등에 대한 생산량을 의미
   ② 소프트웨어 생산성에 영향을 미치는 요소 0409 0709
      • 개발자(프로그래머)의 능력
      • 원활한 팀 의사 전달
      • 제품의 복잡도
   ③ 소프트웨어의 문서(Document) 표준
      • 문서가 표준화되었다는 것은 누구나 알아볼 수 있도록 쉽게 작성되었다는 것을 의미함
      • 소프트웨어의 문서 표준이 되었을 때, 개발자가 얻는 이득 0503 0703
        - 시스템 개발을 위한 분석과 설계가 용이함
        - 프로그램 유지보수가 용이
        - 프로그램의 확장성이 있음

|오|답|쪽|지|
소프트웨어 생산성에 영향을 미치는 요소 오답
✗ 소프트웨어 사용자의 능력

|오|답|쪽|지|
소프트웨어의 문서(Document) 표준이 되었을 때 개발자가 얻는 이득 오답
✗ 프로그램 개발 인력이 감소 됨

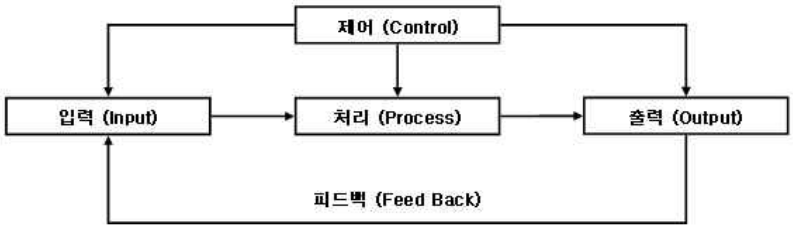
(3) 시스템

1) 시스템은 공통의 목적이나 목표를 달성하기 위하여 여러 가지 상호 관련된 요소들을 유기적으로 결합한 것

2) 소프트웨어는 독립적으로 존재할 수 없으므로 컴퓨터를 기반으로 하는 여러 시스템과 관련을 맺어 상호 동작함

3) 시스템의 구성 요소 0109 0305

|오|답|쪽|지|
시스템의 구성 요소 오답
✗ 상태



① 입력 (Input) : 처리 방법, 처리할 데이터, 조건을 시스템에 투입하는 것
② 처리 (Process) : 입력된 데이터를 처리 방법과 조건에 따라 처리하는 것
③ 출력 (Output) : 처리된 결과를 시스템에서 산출하는 것
④ 제어 (Control) : 자료를 입력하여 출력될 때까지의 처리 과정이 올바르게 진행되는지 감독하는 것
⑤ 피드백 (Feed Back) : 출력된 결과가 예정된 목표를 만족시키지 못할 경우 목표 달성을 위해 반복 처리하는 것

|기|출|문|제|

9904

1. 공학적으로 잘 작성된 소프트웨어의 특성이 아닌 것은?
가. 소프트웨어는 신뢰성이 높아야 하며 효율적이어야 한다.
나. 소프트웨어는 사용자가 원하는 대로 동작해야 한다.
다. 소프트웨어는 편리성이나 유지보수성에 점차 비중을 적게 두는 경향이 있다.
라. 소프트웨어는 잠재적인 에러가 가능한 적어야 하며 유지보수가 용이해야 한다.

0405

2. 좋은 소프트웨어의 조건이라고 할 수 없는 항목은?
가. 남이 알아보기 쉬워야 한다.
나. 경제적이어야 한다.
다. 문서화가 잘 되어 있어야 한다.
라. 프로그램이 독창적이어야 한다.

0409 0709

3. 소프트웨어 개발의 생산성에 영향을 미치는 요소가 아닌 것은?
가. 프로그래머의 능력
나. 팀 의사 전달
다. 제품의 복잡도
라. 소프트웨어 사용자의 능력

0503 0703

4. 소프트웨어의 문서(document) 표준이 되었을 때, 개발자가 얻는 이득으로 거리가 먼 것은?
가. 시스템 개발을 위한 분석과 설계가 용이하다.
나. 프로그램 유지보수가 용이하다.
다. 프로그램의 확장성이 있다.
라. 프로그램 개발 인력이 감소된다.

0109 0305

5. 시스템 구성요소에 해당되지 않는 것은?
가. 입력 나. 출력 다. 제어 라. 상태

1.2 소프트웨어 공학

(1) 소프트웨어 공학(SE, Software Engineering)의 정의

|오|답|쪽|지|
소프트웨어 공학 관련 오답
✗소프트웨어 위기(software crisis)를 완전히 해결한 공학적 원리의 체계
✗신뢰성 있는 소프트웨어를 만들기 위한 도구만을 연구하는 학문
✗점차 많은 비용이 소요되는 소프트웨어 개발에서 가장 경제적인 방법을 찾고자 하는 것

1) 가장 경제적으로 신뢰도 높은 소프트웨어를 만들기 위한 방법, 도구와 절차들의 체계 0605

2) 소프트웨어 공학에 대한 여러 형태의 정의
① 소프트웨어의 개발, 운용, 유지보수, 폐기 처분에 대한 체계적인 접근 방안
② 지정된 비용과 기간 내에 소프트웨어를 체계적으로 생산하고 유지보수하는 데 관련된 기술적이고 관리적인 원리
③ 과학적인 지식을 소프트웨어 설계와 제작에 응용하는 것이며, 이를 개발, 운용, 유지 보수하는 데 필요한 문서 작성 과정

|오|답|쪽|지|
소프트웨어 공학의 공학이 가지는 의미 오답
✗예술성

3) 소프트웨어 공학의 공학(Engineering)이 가지는 의미 0603
① 경제성
② 적시성
③ 보편타당성

(2) 소프트웨어 공학이 나타나게 된 배경 0505

|오|답|쪽|지|
소프트웨어 공학이 나타나게 된 배경 오답
✗유지보수 비용의 감소

1) 소프트웨어(S/W) 비용의 증가

2) 소프트웨어(S/W) 품질과 생산성의 재고

3) 특정 개인에 의존한 시스템 개발

(3) 소프트웨어의 위기(Crisis)

1) 소프트웨어의 위기 0209 0605
컴퓨터의 발달 과정에서 소프트웨어의 개발속도가 하드웨어의 개발속도를 따라가지 못해 사용자들의 요구사항을 감당할 수 없는 문제가 발생함을 의미

2) 소프트웨어 위기의 원인
① 소프트웨어의 특징에 대한 이해 부족
② 소프트웨어의 관리 부재
③ 프로그래밍에만 치중

|오|답|쪽|지|
소프트웨어 위기현상 관련 오답
✗개발인력의 급증
✗소프트웨어의 개발도구 부족

3) 소프트웨어 위기의 현상 9908 0007 0205 0705
① 개발 인력의 부족과 그로 인한 인건비 상승
② 개발 기간의 지연
③ 하드웨어 비용을 초과하는 개발 비용 증가
④ 성능 및 신뢰성의 부족
⑤ 유지 보수의 어려움에 따른 엄청난 비용

(4) 소프트웨어 공학의 기본 원칙 0003 0403 0509

|오|답|쪽|지|
소프트웨어 공학의 기본 원칙 오답
✗충분한 인력 투입

1) 현대적인 프로그래밍 기술 적용

2) 지속적인 검증 시행

3) 결과에 대한 명확한 기록 유지

(5) 소프트웨어 공학의 발전을 위한 소프트웨어 사용자(Software User)로서의 자세 0503 0609

1) 프로그래밍 언어와 알고리즘의 최근 동향을 주기적으로 파악
2) 컴퓨터의 이용 효율이나 워크스테이션에 관한 정보들을 체계적으로 데이터베이스화함
3) 바이러스에 대한 예방에 만전을 기하여 시스템의 안전을 확보함

|오|답|쪽|지|

소프트웨어 공학의 발전을 위한 소프트웨어 사용자(Software User)로서의 자세 오답

✗ 타 기업의 시스템에 몰래 접속하여 새로운 소프트웨어 개발에 관한 정보를 획득함

|기|출|문|제|

0503 0605

1. 소프트웨어 공학에 대한 가장 적절한 설명은?
가. 소프트웨어 위기(software crisis)를 완전히 해결한 공학적 원리의 체계이다.
나. 신뢰성 있는 소프트웨어를 만들기 위한 도구만을 연구하는 학문이다.
다. 가장 경제적으로 신뢰도 높은 소프트웨어를 만들기 위한 방법, 도구와 절차들의 체계이다.
라. 점차 많은 비용이 소요되는 소프트웨어 개발에서 가장 경제적인 방법을 찾고자 하는 것이다.

0603

2. 소프트웨어공학의 공학(Engineering)이 가지는 의미와 어울리지 않는 것은?
가. 예술성 나. 경제성
다. 보편타당성 라. 적시성

0505

3. 소프트웨어공학이 나타나게 된 배경과 관계가 먼 것은?
가. S/W 비용의 증가
나. 유지보수 비용의 감소
다. S/W 품질과 생산성의 재고
라. 특정 개인에 의존한 시스템 개발

0209 0605

4. 컴퓨터의 발달 과정에서 소프트웨어의 개발속도가 하드웨어의 개발속도를 따라가지 못해 사용자들의 요구사항을 감당할 수 없는 문제가 발생함을 의미하는 것은?
가. 소프트웨어의 위기(Crisis)
나. 소프트웨어의 오류(Error)
다. 소프트웨어의 버그(Bug)
라. 소프트웨어의 유지보수(Maintenance)

9908 0205 0705

5. 소프트웨어의 위기현상과 거리가 먼 것은 어느 것인가?
가. 유지보수의 어려움
나. 개발인력의 급증
다. 성능 및 신뢰성의 부족
라. 개발기간의 지연 및 개발비용의 증가

0007

6. 소프트웨어 위기(crisis)의 현상과 관계가 먼 것은?
가. 개발 기간의 지연 및 하드웨어 비용을 초과하는 개발비용의 증가
나. 성능 및 신뢰성 부족
다. 소프트웨어의 개발도구 부족
라. 유지 보수의 어려움에 따른 엄청난 비용

0003 0403

7. 소프트웨어 공학의 기본 원칙이라고 볼 수 없는 것은?
가. 현대적인 프로그래밍 기술 적용
나. 지속적인 검증 시행
다. 결과에 대한 명확한 기록 유지
라. 충분한 인력 투입

0509

8. 소프트웨어 개발 프로젝트를 성공적으로 수행하기 위한 기본 원칙과 거리가 먼 것은?
가. 현대적인 프로그래밍 기술 적용
나. 결과에 대한 명확한 기록 유지
다. 소프트웨어 수명주기에 관련된 계획 및 시행
라. 충분한 예비인력의 보유

0503 0609

9. 소프트웨어 공학의 발전을 위한 소프트웨어 사용자(Software User)로서의 자세로 옳지 않은 것은?
가. 프로그래밍 언어와 알고리즘의 최근 동향을 주기적으로 파악한다.
나. 컴퓨터의 이용 효율이나 워크스테이션에 관한 정보들을 체계적으로 데이터베이스화한다.
다. 타 기업의 시스템에 몰래 접속하여 새로운 소프트웨어 개발에 관한 정보를 획득한다.
라. 바이러스에 대한 예방에 만전을 기하여 시스템의 안전을 확보한다.

1.3 소프트웨어 생명 주기

(1) 소프트웨어 생명 주기(Software Life Cycle) 개념

1) 소프트웨어 생명 주기는 소프트웨어 개발 방법론의 바탕이 되는 것으로 소프트웨어를 개발하기 위해 정의하고 운용, 유지보수 등의 과정을 각 단계별로 나눈 것

2) 소프트웨어 생명 주기는 소프트웨어 개발 단계와 각 단계별 주요 활동, 활동의 결과에 대한 산출물로 표현하며, 소프트웨어 수명 주기라고도 함

|오|답|쪽|지|
소프트웨어 생명주기의 역할 오답
✗단계별 종료 시점을 명확하게 함

3) 소프트웨어 생명주기의 역할 9910
① 프로젝트 비용 산정과 개발 계획을 수립할 수 있는 기본 골격이 됨
② 용어 및 기술의 표준화를 가능하게 함
③ 문서화가 충실한 프로젝트 관리를 가능하게 함

(2) 일반적인 소프트웨어 생명 주기

1) 정의 단계
① '무엇(What)'을 처리하는 소프트웨어를 개발할 것인지 정의하는 단계로, 관리자와 사용자가 가장 많이 참여하는 단계
② 타당성 검토 단계, 개발 계획 단계, 요구사항 분석 단계로 나누어짐

2) 개발 단계
① '어떻게(How)'에 초점을 두고 실제적으로 소프트웨어를 개발하는 단계
② 설계 단계, 구현 단계, 테스트 단계로 나누어짐

|오|답|쪽|지|
유지보수 단계 오답
✗기본설계 단계
✗실행단계

3) 유지보수 단계 9904 9908 0203 0403 0605
① 소프트웨어를 직접 운용하며, '변경(Change)'에 초점을 두고 여러 환경 변화에 따라 소프트웨어를 적응 및 유지시키는 단계
② 소프트웨어 생명 주기 단계 중에서 시간과 비용이 가장 많이 요구되는 단계

(3) 소프트웨어 생명 주기 모형의 종류

1) 폭포수 모형(Waterfall Model) 0709

|오|답|쪽|지|
폭포수 모형에 대한 설명 오답
✗앞 단계에서 발견하지 못한 오류를 다음 단계에서 발견했을 때 오류 수정이 용이함
✗요구분석 단계에서 프로토타입을 사용하는 것이 특징임
✗각 단계의 병렬 수행이 가능함
✗요구사항의 변경이 용이함

① 폭포수 모형 9908 0010 0209
- 소프트웨어 개발 과정의 앞 단계가 끝나야만 다음 단계로 넘어갈 수 있는 선형 순차적 모형
- 제품의 일부가 될 매뉴얼을 작성해야 함
- 각 단계가 끝난 후 결과물이 명확히 나와야 함
- 폭포수 모형은 소프트웨어 공학에서 가장 오래되고 가장 폭넓게 사용된 전통적인 소프트웨어 생명 주기 모형으로, 고전적 생명 주기 모형이라고도 함

② 폭포수 모형(Waterfall Model) 개발 순서 0103 0106 0109 0205 0303 0308 0503 0609

타당성 검토 ⇨ 계획 ⇨ 요구 분석 ⇨ 설계 ⇨ 구현(코딩) ⇨ 시험(검사, 테스트) ⇨ 유지보수

③ 폭포수 모형의 장점 9904 0305 0308
- 모형의 적용 경험과 성공 사례가 많음
- 단계별 정의가 분명하고, 전체 공조의 이해가 용이
- 단계별 산출물이 정확하여 개발 공정의 기준점을 잘 제시

④ 폭포수 모형의 단점 0010
- 개발 과정 중에 발생하는 새로운 요구나 경험을 설계에 반영하기 어려움
- 처음부터 사용자들이 모든 요구사항들을 명확하게 제시해야 함
- 단계별로 오류 없이 다음 단계로 진행해야 하는데 현실적으로 오류 없이 다음 단계로 진행하기는 어려움

2) 프로토타입 모형 (Prototype Model)

① 프로토타입 모형 9908 0003 0103 0106 0403 0409 0509 0609 0703
- 시스템의 일부 혹은 시스템의 모형을 만드는 과정으로서, 요구된 소프트웨어의 일부를 구현하여, 추후 구현단계서 사용될 골격코드가 되는 모형
- 실제 상황이 나오기 전에 가상으로 시뮬레이션을 통해 최종 결과물에 대한 예측을 할 수 있는 소프트웨어 수명 주기 모형
- 최종 결과물이 만들어지기 전에 의뢰자가 최종 결과물의 일부 또는 모형을 볼 수 있음
- 프로토타입은 발주자나 개발자 모두에게 공동의 참조 모델을 제공
- 프로토타입은 구현 단계의 구현 골격이 됨
- 구축하고자 하는 시스템의 요구사항이 불명확한 경우 가장 적절하게 적용될 수 있음 9910

|오|답|쪽|지|
프로토타이핑 모형(Prototyping Model) 설명 오답
✗ 개발 단계에서 오류 수정이 불가능하므로 유지보수 비용이 많이 발생함
✗ 계획수립 모형
✗ 코코모 모형 (cocomo model)

② 프로토타이핑(원형) 모형의 개발 작업 순서 0010

요구 수집 ⇨ 빠른 설계 ⇨ 프로토타입 구축 ⇨ 고객평가 ⇨ 프로토타입 조정 ⇨ 구현

③ 프로토타입 모형의 장점 0007 0109 0303 0503 0603 0605 0709
- 사용자 요구사항의 정확한 파악 및 충실한 반영
- 요구사항의 변경이 용이
- 정보문제의 본질에 대한 불확실성과 그 정보문제를 해결하기 위해 사용자가 제시하는 요구의 불확실성을 줄일 수 있음

④ 프로토타이핑(Prototyping) 접근방법 채용에 따른 불확실성 결정요인 0603
- 지원이 필요한 일로부터의 요구연역(要求演繹)
- 사용자와 분석자의 지식과 경험의 수준
- 커뮤니케이션 문제가 일어날 가능성

|오|답|쪽|지|
프로토타이핑(Prototyping) 접근방법 채용에 따른 불확실성 결정요인 오답
✗ 프로토타이핑 시에 소요되는 비용문제

3) 나선형(Spiral) 모형 0405

① 반복적인 작업을 수행하는 점증적 생명주기 모델

② 고객과의 의사소통(Communication)을 통해 계획수립과 위험분석, 구축, 고객평가의 과정을 거쳐 소프트웨어를 개발

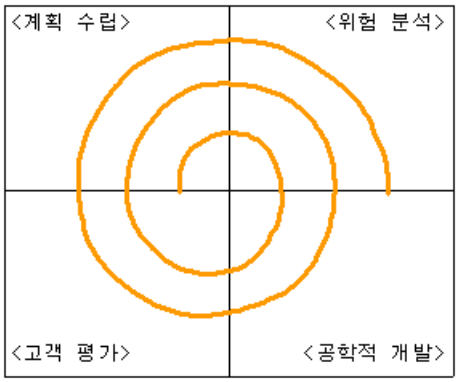
③ 가장 큰 장점인 위험분석 단계에서 기술과 관리의 위험요소 들을 하나씩 제거해 나감으로서 완성도 높은 소프트웨어를 만들 수 있음

④ 비용이 많이 들거나 시간이 많이 소요되는 대규모 프로젝트나 큰 시스템을 구축할 때 유리

⑤ 점진적으로 개발 과정이 반복되므로 누락되거나 추가된 요구사항을 첨가할 수 있고, 정밀하며, 유지보수 과정이 필요 없음

|오|답|쪽|지|
나선형(spiral) 모형의 단계 오답
✗객체 구현 (Object Implementation)
✗시스템 유지보수

2) 나선형 모형의 단계와 순서 9904 9908 0203 0209 0305 0409 0505
(=Boehm이 제안한 나선형 모델의 태스크(Task))

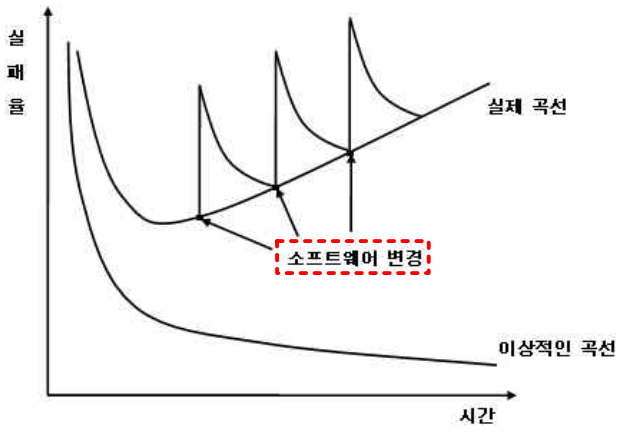


Planning ⇨ Risk Analysis ⇨ Engineering ⇨ Customer Evaluation

① 계획 수립(Planning) : 목적, 다른 방안, 제약 조건을 결정
② 위험 분석(Risk Analysis) : 다른 방안을 분석하고 위험을 식별하고 해결
③ 공학적 개발(Engineering) : 개선된 한 단계 높은 수준의 제품을 개발
④ 고객 평가(Customer Evaluation) : 공학적 결과를 평가

※ 소프트웨어 실패율(오류율) 0503

아래의 그림은 소프트웨어에 대한 사용 시간과 실패율에 대한 관계를 나타내는데, 소프트웨어 실패율(오류율)은 소프트웨어를 변경함으로 예상치 못한 오류(실패)가 발생할 수 있다는 것을 확인할 수 있음



|기|출|문|제|

9910
1. 소프트웨어 생명주기의 역할로 거리가 먼 것은?
가. 프로젝트의 비용 산정과 개발 계획을 수립할 수 있는 기본 골격이 된다.
나. 단계별 종료 시점을 명확하게 한다.
다. 용어의 표준화를 가능하게 한다.
라. 문서화가 충실한 프로젝트 관리를 가능하게 한다.

9908 0203 0403
2. 소프트웨어 라이프사이클 단계 중 가장 오랜 시간이 걸리며 대부분의 비용을 차지하는 단계는 어느 것인가?
가. 타당성 검토 단계
나. 운용 및 유지보수 단계
다. 기본설계 단계
라. 실행단계

9908 0209
3. 폭포수 모델(waterfall model)에 대한 설명으로 옳지 않은 것은 어느 것인가?
가. 앞 단계가 끝나야만 다음 단계로 넘어갈 수 있다.
나. 요구분석 단계에서 프로토타입을 사용하는 것이 특징이다.
다. 제품의 일부가 될 매뉴얼을 작성해야 한다.
라. 각 단계가 끝난 후 결과물이 명확히 나와야 한다.

0103 0106 0303 0503
4. 소프트웨어 수명주기 도형 중 폭포수 모형 (Waterfall Model)의 개발 단계로 옳은 것은?
가. 계획-분석-설계-시험-구현(코딩)-유지보수
나. 계획-분석-설계-구현(코딩)-시험-유지보수
다. 계획-설계-분석-구현(코딩)-시험-유지보수
라. 계획-분석-설계-구현(코딩)-시험-설치

0109 0609
5. 전통적인 소프트웨어 개발 방법론인 폭포수형 (waterfall) 모델에서 개발 순서가 옳은 것은?
가. 타당성 검토-계획-분석-구현-설계
나. 타당성 검토-분석-계획-설계-구현
다. 타당성 검토-계획-분석-설계-구현
라. 타당성 검토-분석-계획-구현-설계

9904 0308
6. 소프트웨어 수명주기 모형 중 폭포수 모형의 장점이 아닌 것은?
가. 적용사례가 많다.
나. 단계별 정의가 분명하다.
다. 단계별 산출물이 명확하다.
라. 요구사항의 변경이 용이하다.

9908 0106 0409 0609
7. 프로토타이핑 모형(Prototyping Model)에 대한 설명으로 옳지 않은 것은?
가. 최종 결과물이 만들어지기 전에 의뢰자가 최종 결과물의 일부 또는 모형을 볼 수 있다.
나. 개발 단계에서 오류 수정이 불가능하므로 유지보수 비용이 많이 발생한다.
다. 프로토타입은 발주자나 개발자 모두에게 공동의 참조 모델을 제공한다.
라. 프로토타입은 구현 단계의 구현 골격이 될 수 있다.

0103 0403
8. 실제 상황이 나오기 전에 가상으로 시뮬레이션을 통해 최종 결과물에 대한 예측을 할 수 있는 소프트웨어 수명 주기 모형은?
가. 집중적 모형(spiral model)
나. 프로토타이핑 모형(prototyping model)
다. 코코모 모형(cocomo model)
라. 폭포수 모형(waterfall model)

0007 0303 0503 0603 0609 0709
9. 프로토타입 모형의 장점으로 가장 적절한 것은?
가. 프로젝트 관리가 용이하다.
나. 노력과 비용이 절감된다.
다. 요구사항을 충실히 반영한다.
라. 관리와 개발이 명백히 구분된다.

0010
10. 다음은 프로토타이핑(원형) 모형의 개발에 필요한 작업을 기술한 것이다. 작업 순서대로 옳게 나열한 것은?

① 빠른 설계	② 프로토타입 구축
③ 프로토타입 조정	④ 요구 수집
⑤ 구현	⑥ 고객평가

가. ④-②-①-③-⑥-⑤
나. ④-①-②-⑤-⑥-③
다. ④-①-②-③-⑥-⑤
라. ④-①-②-⑥-③-⑤