



정보처리기사 실기 시험은 한국산업인력공단에서 문제를 공개하지 않아 문제 복원에 많은 어려움이 있습니다. 다음에 제시된 문제는 시험을 치른 학생들의 기억을 토대로 복원한 것이므로, 일부 내용이나 문제별 배점이 실제 시험과 다를 수 있음을 알립니다.

저작권 안내

이 자료는 시나공 카페 회원을 대상으로 하는 자료로서 개인적인 용도로만 사용할 수 있습니다. 허락 없이 복제하거나 다른 매체에 옮겨 실을 수 없으며, 상업적 용도로 사용할 수 없습니다.

*** 수험자 유의사항 ***

1. 시험 문제지를 받는 즉시 응시하고자 하는 종목의 문제지가 맞는지를 확인하여야 합니다.
2. 시험 문제지 총면수·문제번호 순서·인쇄상태 등을 확인하고, 수험번호 및 성명을 답안지에 기재하여야 합니다.
3. 문제 및 답안(지), 채점기준은 일절 공개하지 않으며 자신이 작성한 답안, 문제 내용 등을 수험표 등에 이기 (옮겨 적는 행위) 등은 관련 법 등에 의거 불이익 조치 될 수 있으니 유의하시기 바랍니다.
4. 수험자 인적사항 및 답안작성(계산식 포함)은 흑색 기구만 사용하되, 동일한 한 가지 색의 필기구만 사용하여 하며 흑색을 제외한 유색 필기구 또는 연필류를 사용하거나 2가지 이상의 색을 혼합 사용하였을 경우 그 문항은 0점 처리됩니다.
5. 답란(답안 기재란)에는 문제와 관련 없는 불필요한 낙서나 특이한 기록사항 등을 기재하여서는 안되며 부정의 목적으로 특이한 표식을 하였다고 판단될 경우에는 모든 문항이 0점 처리됩니다.
6. 답안을 정정할 때에는 반드시 정정부분을 두 줄(=)로 그어 표시하여야 하며, 두 줄로 굵지 않은 답안은 정정하지 않은 것으로 간주합니다. (수정테이프, 수정액 사용불가)
7. 답안의 한글 또는 영문의 오타자는 오답으로 처리됩니다. 단, 답안에서 영문의 대·소문자 구분, 띄어쓰기는 여부에 관계 없이 채점합니다.
8. 계산 또는 디버깅 등 계산 연습이 필요한 경우는 <문 제> 아래의 연습란을 사용하시기 바라며, 연습란은 채점대상이 아닙니다.
9. 문제에서 요구한 가지 수(항수) 이상을 답란에 표기한 경우에는 답안기재 순으로 요구한 가지 수(항수)만 채점하고 한 항에 여러 가지를 기재하더라도 한 가지로 보며 그 중 정답과 오답이 함께 기재란에 있을 경우 오답으로 처리됩니다.
10. 한 문제에서 소문제로 파생되는 문제나, 가지수를 요구하는 문제는 대부분의 경우 부분채점을 적용합니다. 그러나 소문제로 파생되는 문제 내에서의 부분 배점은 적용하지 않습니다.
11. 답안은 문제의 마지막에 있는 답란에 작성하여야 합니다.
12. 부정 또는 불공정한 방법(시험문제 내용과 관련된 메모지사용 등)으로 시험을 치른 자는 부정행위자로 처리되어 당해 시험을 중지 또는 무효로 하고, 2년간 국가기술자격검정의 응시자격이 정지됩니다.
13. 시험위원이 시험 중 신분확인을 위하여 신분증과 수험표를 요구할 경우 반드시 제시하여야 합니다.
14. 시험 중에는 통신기기 및 전자기기(휴대용 전화기 등)를 지참하거나 사용할 수 없습니다.
15. 국가기술자격 시험문제는 일부 또는 전부가 저작권법상 보호되는 저작물이고, 저작권자는 한국산업인력공단입니다. 문제의 일부 또는 전부를 무단 복제, 배포, 출판, 전자출판 하는 등 저작권을 침해하는 일체의 행위를 금합니다.

※ 수험자 유의사항 미준수로 인한 채점상의 불이익은 수험자 본인에게 전적으로 책임이 있음

문제 1 소프트웨어 공학에서 리팩토링(Refactoring)을 하는 목적에 대해 간략히 서술하시오. (5점)

답 :

문제 2 다음 Java로 구현된 프로그램을 분석하여 그 실행 결과를 쓰시오. (단, 출력문의 출력 서식을 준수하시오.) (5점)

```
public class Test {  
    public static void main(String[] args) {  
        int i = 0, c = 0;  
        while (i < 10) {  
            i++;  
            c *= i;  
        }  
        System.out.println(c);  
    }  
}
```

답 :

문제 3 <학생> 테이블에서 '이름'이 "민수"인 튜플을 삭제하고자 한다. 다음 <처리조건>을 참고하여 SQL문을 작성하시오. (5점)

<처리조건>

- 명령문 마지막의 세미콜론(;)은 생략이 가능하다.
- 인용 부호가 필요한 경우 작은 따옴표(' ')를 사용한다.

답 :

연 습 란

※ 다음 여백은 연습란으로 사용하시기 바랍니다.

문제 4 다음 지문의 괄호에 들어갈 알맞은 용어를 영문(Full name 또는 약어)으로 쓰시오. (5점)

()는 TCP/IP 기반의 인터넷 통신 서비스에서 인터넷 프로토콜(IP)과 조합하여 통신 중에 발생하는 오류의 처리와 전송 경로의 변경 등을 위한 제어 메시지를 취급하는 무연결 전송용 프로토콜로, OSI 기본 참조 모델의 네트워크 계층에 속한다.

답 :

문제 5 데이터베이스의 스키마(Schema)에 대해 간략히 서술하시오. (5점)

답 :

문제 6 다음 지문의 괄호에 들어갈 알맞은 용어를 쓰시오. (5점)

심리학자 톰 마릴은 컴퓨터가 메시지를 전달하고, 메시지가 제대로 도착했는지 확인하며, 도착하지 않았을 경우 메시지를 재전송하는 일련의 방법을 ‘기술적 은어’를 뜻하는 ()이라는 용어로 정의하였다.

답 :

문제 7 다음이 설명하고 있는 관계 대수 연산자의 기호를 쓰시오. (5점)

릴레이션 A, B가 있을 때 릴레이션 B의 조건에 맞는 것들만 릴레이션 A에서 분리하여 프로젝션을 하는 연산이다.

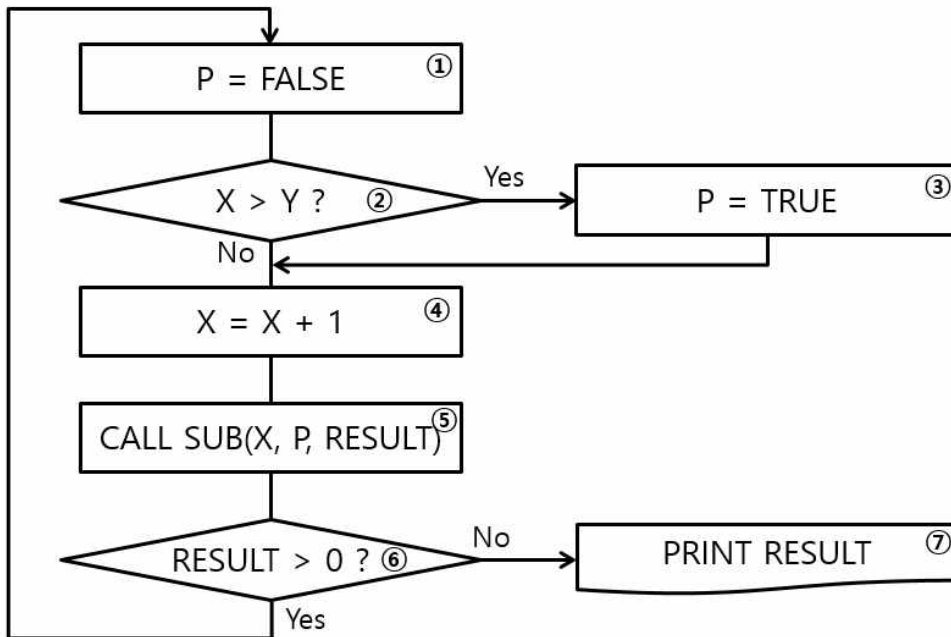
답 :

연 습 란

※ 다음 여백은 연습란으로 사용하시기 바랍니다.

문제 8 다음은 화이트박스 테스트의 프로그램 제어흐름이다. 다음의 순서도를 참고하여 분기 커버리지로 구성할 테스트 케이스를 작성하시오. (5점)

<순서도>



<작성예시>

(1) → (2) → (4)

답 :

() → () → () → () → () → () → ()
 () → () → () → () → () → ()

연 습 란

※ 다음 여백은 연습란으로 사용하시기 바랍니다.

문제 9 다음의 <성적> 테이블에서 과목별 점수의 평균이 90점 이상인 '과목이름', '최소점수', '최대점수'를 검색하고자 한다. <처리조건>을 참고하여 적합한 SQL문을 작성하시오. (5점)

<성적>

학번	과목번호	과목이름	학점	점수
a2001	101	컴퓨터구조	6	95
a2002	101	컴퓨터구조	6	84
a2003	302	데이터베이스	5	89
a2004	201	인공지능	5	92
a2005	302	데이터베이스	5	100
a2006	302	데이터베이스	5	88
a2007	201	인공지능	5	93

<결과>

과목이름	최소점수	최대점수
데이터베이스	88	100
인공지능	92	93

<처리조건>

- WHERE문은 사용하지 않는다.
- GROUP BY와 HAVING을 이용한다.
- 집계함수(Aggregation Function)를 사용하여 명령문을 구성한다.
- '최소점수', '최대점수'는 별칭(Alias)을 위한 AS문을 이용한다.
- 명령문 마지막의 세미콜론(;)은 생략이 가능하다.
- 인용 부호가 필요한 경우 작은 따옴표(' ')를 사용한다.

답 :

연 습 란

※ 다음 여백은 연습란으로 사용하시기 바랍니다.

문제 10 다음 C언어로 구현된 프로그램을 분석하여 그 실행 결과를 쓰시오. (단, 출력문의 출력 서식을 준수하시오.) (5점)

```
#include <stdio.h>
int r1() {
    return 4;
}
int r10() {
    return (30 + r1());
}
int r100() {
    return (200 + r10());
}
int main() {
    printf("%d\n", r100());
    return 0;
}
```

답 :

문제 11 소프트웨어 개발에서의 작업 중 형상 통제에 대해 간략히 서술하시오. (5점)

답 :

문제 12 소프트웨어가 수행할 특정 기능을 알기 위해서 각 기능이 완전히 작동되는 것을 입증하는 테스트로, 동치 클래스 분해 및 경계값 분석을 이용하는 테스트 기법을 쓰시오. (5점)

답 :

연 습 란

※ 다음 여백은 연습란으로 사용하시기 바랍니다.

문제 13 다음 Java로 구현된 프로그램을 분석하여 그 실행 결과를 쓰시오. (단, 출력문의 출력 서식을 준수 하시오.) (5점)

```
abstract class Vehicle {
    String name;
    abstract public String getName(String val);
    public String getName() {
        return "Vehicle name : " + name;
    }
}
class Car extends Vehicle {
    private String name;
    public Car(String val) {
        name = super.name = val;
    }
    public String getName(String val) {
        return "Car name : " + val;
    }
    public String getName(byte[] val) {
        return "Car name : " + val;
    }
}
public class Test {
    public static void main(String[] args) {
        Vehicle obj = new Car("Spark");
        System.out.print(obj.getName());
    }
}
```

답 :

문제 14 헝가리안 표기법(Hungarian Notation)에 대해 간략히 서술하시오. (5점)

답 :

연 습 란

※ 다음 여백은 연습란으로 사용하시기 바랍니다.

문제 15 다음 Java로 구현된 프로그램을 분석하여 그 실행 결과를 쓰시오. (단, 출력문의 출력 서식을 준수 하시오.) (5점)

```
public class Test{
    public static void main(String[] args){
        int a = 0, sum = 0;
        while (a < 10) {
            a++;
            if (a%2 == 1)
                continue;
            sum +=a;
        }
        System.out.println(sum);
    }
}
```

답 :

문제 16 다음 설명에 해당하는 라우팅 프로토콜(Routing Protocol)을 쓰시오. (5점)

- RIP의 단점을 해결하여 새로운 기능을 지원하는 인터넷 프로토콜이다.
- 인터넷 망에서 이용자가 최단 경로를 선정할 수 있도록 라우팅 정보에 노드 간의 거리 정보, 링크 상태 정보를 실시간으로 반영하여 최단 경로로 라우팅을 지원한다.
- 대규모 네트워크에서 많이 사용된다.
- 최단 경로 탐색에 Dijkstra 알고리즘을 사용한다.
- 라우팅 정보에 변화가 생길 경우 변화된 정보만 네트워크 내의 모든 라우터에 알린다.
- 링크 스테이트 라우팅 알고리즘을 사용하며, 하나의 자율 시스템(AS)에서 동작하면서 내부 라우팅 프로토콜의 그룹에 도달한다.

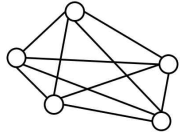
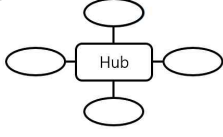
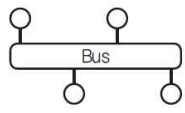
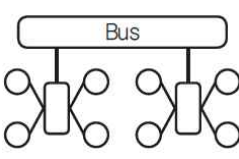
답 :

연 습 란

※ 다음 여백은 연습란으로 사용하시기 바랍니다.

문제 17 EAI에 대한 다음 설명에서 괄호에 들어갈 알맞은 답을 쓰시오. (5점)

EAI(Enterprise Application Integration)는 기업 내 각종 애플리케이션 및 플랫폼 간의 정보 전달, 연계, 통합 등 상호 연동이 가능하게 해주는 솔루션이다. 비즈니스 간 통합 및 연계성을 증대시켜 효율성 및 각 시스템 간의 확정성(Determinacy)을 높여 준다. EAI의 구축 유형은 다음과 같다.

(①)	<ul style="list-style-type: none"> - 가장 기본적인 애플리케이션 통합 방식으로, 애플리케이션을 1 : 1로 연결한다. - 변경 및 재사용이 어렵다. 	
(②)	<ul style="list-style-type: none"> - 단일 접점인 허브 시스템을 통해 데이터를 전송하는 중앙 집중형 방식이다. - 확장 및 유지 보수가 용이하다. - 허브 장애 발생 시 시스템 전체에 영향을 미친다. 	
Message Bus	<ul style="list-style-type: none"> - 애플리케이션 사이에 미들웨어를 두어 처리하는 방식이다. - 확장성이 뛰어나며 대용량 처리가 가능하다. 	
Hybrid	<ul style="list-style-type: none"> - 그룹 내에서는 (②) 방식을, 그룹 간에는 Message Bus 방식을 사용한다. - 필요한 경우 한 가지 방식으로 EAI 구현이 가능하다. - 데이터 병목 현상을 최소화할 수 있다. 	

답

- ① :
- ② :

문제 18 UI(User Interface)의 설계 원칙 중 직관성에 대해 간략히 서술하시오. (5점)

답 :

문제 19 C++에서 생성자(Constructor)에 대해 간략히 서술하시오. (5점)

답 :

연 습 란

※ 다음 여백은 연습란으로 사용하시기 바랍니다.

문제 20 다음 <속성 정의서>를 참고하여 <학생> 테이블에 대해 20자의 가변 길이를 가진 '주소' 속성을 추가하는 <SQL문>을 완성하십시오. (단, SQL문은 ISO/IEC 9075 표준을 기반으로 작성하십시오.) (5점)

<속성 정의서>

속성명	데이터타입	제약조건	테이블명
학번	CHAR(10)	UNIQUE	학생
이름	VARCAHR(8)	NOT NULL	학생
주민번호	CHAR(13)		학생
학과	VARCAHR(16)	FOREIGN KEY	학생
학년	INT		학생

<SQL문>

(①) TABLE 학생 (②) 주소 VARCHAR(20);

답

- ① :
- ② :

연 습 란

※ 다음 여백은 연습란으로 사용하시기 바랍니다.

기출문제 정답 및 해설

[문제 1]

※ 다음 중 밑줄이 표시된 내용은 반드시 포함되어야 합니다.

소프트웨어의 기능 변경 없이, 내부 구조만을 개선하여 소프트웨어를 보다 이해하기 쉽고, 수정하기 쉽도록 만들기 위함

[문제 2]

0

[해설]

```
public class Test {  
    public static void main(String[] args) {  
        ❶      int i = 0, c = 0;  
        ❷      while (i < 10) {  
        ❸          i++;  
        ❹          c *= i;  
        }  
        ❺      System.out.println(c);  
    }  
}
```

모든 Java 프로그램은 반드시 main() 메소드에서 시작한다.

- ❶ 정수형 변수 i와 c를 선언하고 각각 0으로 초기화한다.
- ❷ i가 10보다 작은 동안 ❸, ❹번을 반복 수행한다.
- ❸ 'i = i + 1'과 동일하다. i의 값을 1 증가시킨다.
- ❹ 'c = c * i'와 동일하다. c * i의 값을 c에 저장한다.
- ❺ c의 값을 화면에 출력하고 커서를 다음 줄 처음으로 옮긴다.

결과 0

반복문 실행에 따른 변수들의 값의 변화는 다음과 같다.

i	c
0	0
1	0
2	0
3	0
4	0
5	0
6	0
7	0
8	0
9	0
10	0

[문제 3]

DELETE FROM 학생 WHERE 이름 = '민수';

[풀이]

DELETE	삭제하라
FROM 학생	<학생> 테이블을 대상으로 하라.
WHERE 이름 = '민수';	'이름'이 "민수"인 자료만을 대상으로 한다.

[문제 4]

※ 다음 중 하나를 쓰면 됩니다.

ICMP, Internet Control Message Protocol

[답안 작성 시 주의 사항]

한글 또는 영문을 Full-name이나 약어로 쓰라는 지시사항이 없을 경우 한글이나 영문 약어로 쓰는 것이 유리합니다. 영문을 Full-name으로 풀어쓰다가 스펠링을 틀리면 오답으로 처리되니까요.

[문제 5]

※ 다음 중 밑줄이 표시된 내용은 반드시 포함되어야 합니다.

데이터베이스의 구조와 제약 조건에 관한 전반적인 명세를 기술한 것

[문제 6]

※ 다음 중 하나를 쓰면 됩니다.

프로토콜, Protocol

[문제 7]

÷

[문제 8]

(1) → (2) → (3) → (4) → (5) → (6) → (7)

(1) → (2) → (4) → (5) → (6) → (1)

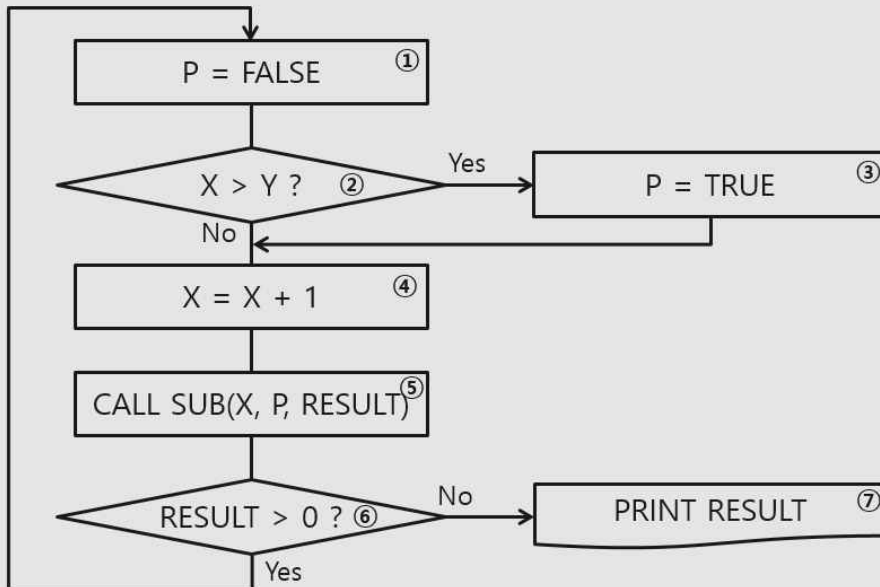
또는

(1) → (2) → (3) → (4) → (5) → (6) → (1)

(1) → (2) → (4) → (5) → (6) → (7)

[해설]

- 화이트박스 테스트의 검증 기준(Coverage) 중 분기 검증 기준(Branch Coverage)은 소스 코드의 모든 조건문이 한 번 이상 수행되도록 테스트 케이스를 설계하는 방법입니다.



- 위의 순서도를 기반으로 한 테스트 케이스는 ①번에서 시작한 프로세스가 조건문인 ②번과 ⑥번에 도달했을 때 반드시 한 번은 Yes로 한 번은 No로 진행되도록 설계되어야 합니다. 또한 문제지의 답란에 7칸의 괄호와 6칸의 괄호가 제시되어 있으므로, 두 번의 프로세스로 모든 코드가 수행되도록 설계해야 합니다.

[첫 번째 테스트 케이스 설계 방안]

- 7칸 괄호 : ①②③④⑤⑥⑦
 - 6칸 괄호 : ①②④⑤⑥①
- ※ 7칸 괄호에 맞는 테스트 케이스를 설계할 때 ②번 조건문에서 Yes로, ⑥번 조건문에서 No로 진행되도록 설계했으므로, 6칸 괄호에 맞는 테스트 케이스는 ②번 조건문에서 No로, ⑥번 조건문에서 Yes로 진행되도록 설계해야 합니다.

[두 번째 테스트 케이스 설계 방안]

- 7칸 괄호 : ①②③④⑤⑥①
 - 6칸 괄호 : ①②④⑤⑥⑦
- ※ 7칸 괄호에 맞는 테스트 케이스를 설계할 때 ②번 조건문에서 Yes로, ⑥번 조건문에서도 Yes로 진행되도록 설계했으므로, 6칸 괄호에 맞는 테스트 케이스는 ②번 조건문에서 No로, ⑥번 조건문에서도 No로 진행되도록 설계해야 합니다.

[문제 9]

SELECT 과목이름, MIN(점수) AS 최소점수, MAX(점수) AS 최대점수 FROM 성적 GROUP BY 과목이름 HAVING AVG(점수) >= 90;

[풀이]

- ① SELECT 과목이름, MIN(점수) AS 최소점수, MAX(점수) AS 최대점수
- ② FROM 성적
- ③ GROUP BY 과목이름
- ④ HAVING AVG(점수) >= 90;

- ① ‘과목이름’, ‘점수’의 최소값, ‘점수’의 최대값을 표시하되, ‘점수’의 최소값은 ‘최소점수’로, ‘점수’의 최대값은 ‘최대점수’로 표시한다.
- ② <성적> 테이블을 대상으로 검색한다.
- ③ ‘과목이름’을 기준으로 그룹을 지정한다.
- ④ 각 그룹의 ‘점수’의 평균이 90보다 크거나 같은 그룹만을 표시한다.

[문제 10]

234

[해설]

```
#include <stdio.h>
int r1() {
    ④ return 4;
}
int r10() {
    ③ return (30 + r1());
}
int r100() {
    ② return (200 + r10());
}
int main() {
    ① printf("%d\\n", r100());
    return 0;
}
```

4 반환

34(30 + 4) 반환

234(200 + 34) 반환

결과 **234**

main() 함수에서의 return 0은 프로그램의 종료를 의미한다.

- 모든 C 프로그램은 반드시 main() 함수부터 시작한다.
- ① 인수 없이 r100() 함수를 호출한 다음 반환받은 값을 화면에 출력한다. ②번으로 이동한다.
 - ② r10() 함수를 호출하여 반환받은 값에 200을 더한 후 r100() 함수를 호출한 ①번으로 반환한다. r10() 함수를 호출하기 위해 ③번으로 이동한다.
 - ③ r1() 함수를 호출하여 반환받은 값에 30을 더한 후 r100() 함수를 호출한 ②번으로 반환한다. r1() 함수를 호출하기 위해 ④번으로 이동한다.
 - ④ 반환값 4를 가지고 r1() 함수를 호출했던 ③번으로 이동한다.

[문제 11]

※ 다음 중 밑줄이 표시된 내용은 반드시 포함되어야 합니다.

식별된 형상 항목에 대한 변경 요구를 검토하여 현재의 기준선이 잘 반영될 수 있도록 조정하는 작업

[문제 12]

※ 다음 중 하나를 쓰면 됩니다.

블랙박스 테스트, Black Box Test

[문제 13]

Vehicle name : Spark

[답안 작성 시 주의 사항]

프로그램의 실행 결과는 부분 점수가 없으므로 정확하게 작성해야 합니다. 예를 들어, 출력값들을 줄을 나눠

Vehicle name :

Spark 와 같이 썼을 경우 부분 점수 없이 완전히 틀린 것으로 간주됩니다.

[해설]

```

abstract class Vehicle {
    String name;
    abstract public String getName(String val);
    ⑤ public String getName() {
    ⑥     return "Vehicle name : " + name;
    }
}

class Car extends Vehicle {
    private String name;
    ② public Car(String val) {
    ③     name = super.name = val;
    }
    public String getName(String val) {
        return "Car name : " + val;
    }
    public String getName(byte[] val) {
        return "Car name : " + val;
    }
}

public class Test {
    public static void main(String[] args) {
    ①     Vehicle obj = new Car("Spark");
    ④⑦     System.out.print(obj.getName());
    }
}

```

추상 클래스 Vehicle을 정의한다.

- **abstract [클래스 정의부]** : 추상 클래스 정의 시 추가하는 예약어

※ 추상 클래스는 내부에 실행 코드가 없는 추상 메소드를 포함하기 때문에 객체 변수의 생성자로 사용할 수 없다.

예) Vehicle obj = new Vehicle(); - X

추상 메소드 getName(String val)을 정의한다.

- **abstract [메소드 정의부]** : 추상 메소드 정의 시 추가하는 예약어

※ 추상 메소드는 선언만 있고 내부에 실행 코드가 없는 메소드로, 이후 상속 관계가 설정된 자식 클래스에 의해 재정의된 후 사용된다.

클래스 Car를 정의하고 부모 클래스로 Vehicle을 지정하면서 Vehicle에 속한 변수와 메소드를 상속받는다.

- **extends [클래스명]** : 클래스 정의 시 상속받을 클래스를 추가하는 예약어

※ Car 클래스에 있는 getName(String val) 메소드는 Vehicle 클래스에 있는 getName(String val) 메소드와 이름은 같지만 실행 코드는 다르다. 이와 같이 부모 클래스에서 정의한 메소드를 자식 클래스에서 다시 정의해서 사용하는 것을 메소드 오버라이딩 또는 메소드 재정의 라고 한다.

모든 Java 프로그램은 반드시 main() 메소드에서 시작한다.

① Vehicle obj = new Car("Spark");

클래스 Car로 형 변환이 수행된 클래스 Vehicle의 객체 변수 obj를 선언하고, "Spark"를 인수로 클래스 Car의 생성자를 호출한다.

• **[부모클래스명] [객체변수명] = new [자식클래스생성자()]** : 자식 클래스 생성자로 인스턴스를 생성할 때 자료형을 부모 클래스로 지정하면 생성된 인스턴스는 부모 클래스로 묵시적인 클래스 형 변환이 발생한다. 이렇게 형 변환이 발생했을 때 부모 클래스와 자식 클래스에 같은 이름의 메소드가 존재하면 호출되는 메소드는 생성되는 인스턴스에 따라 결정된다. 즉 선언한 클래스 자료형이 아닌 생성된 자식 클래스 인스턴스의 메소드를 호출하며, 이런 기술을 가상 메소드라고 한다.

• 클래스에 속하는 멤버 변수는 인스턴스가 생성될 때마다 새로 생성되지만 메소드는 실행해야 할 코드의 집합이기 때문에 클래스의 인스턴스가 여러 개 생성된다고 해서 메소드가 여러 개 생성되지는 않는다.

• 일반적으로 프로그램에서 메소드를 호출한다는 것은 그 메소드의 명령 집합이 있는 메모리 위치를 참조하여 명령을 실행하는 것인데, 가상 메소드의 경우에는 가상 메소드의 이름과 실제 메모리 주소가 짝을 이루는 '가상 메소드 테이블'이 만들어진다. 어떤 메소드가 호출되면 이 테이블에서 주소 값을 찾아 해당 메소드의 명령을 수행하는 것이다. 다음은 Vehicle 클래스와 Car 클래스의 가상 메소드 테이블이다.

※ 객체 변수 obj는 Vehicle 클래스의 객체 변수이다. Vehicle 클래스는 실행 코드가 없는 추상 메소드 getName(String val)로 인해 객체 변수의 생성이 불가능해야 하지만, 형 변환으로 인해 getName(String val) 메소드가 Car 클래스에서 재정의 되었으므로 객체 변수의 생성이 가능해진다.

Vehicle 클래스의 가상 메소드 테이블

메소드 이름	메소드 주소
getName(String val)	마지막 300
getName()	100

Car 클래스의 가상 메소드 테이블

메소드 이름	메소드 주소
Car(String val)	200
getName(String val)	300
getName(byte[] val)	400

주소 메소드 영역

100

200

300

400

```

public String getName() {
    return "Vehicle name : " + name;
}

public Car(String val) {
    name = super.name = val;
}

public String getName(String val) {
    return "Car name : " + val;
}

public String getName(byte[] val) {
    return "Car name : " + val;
}

```

재정의*

※ 객체 변수 obj는 추상 클래스인 Vehicle의 객체 변수이므로 주소가 '미지정' 상태여야 하지만 형 변환으로 인해 Car 클래스의 getName() 메소드가 재정의 되었으므로 여기서는 300을 가리키게 된다.

- ② 클래스 Car의 생성자 Car()의 시작점이다. ①번에서 전달받은 "Spark"를 val에 저장한다.
 - ※ **생성자(Constructor)** : 인스턴스를 생성할 때 자동으로 호출되는 메소드로 반환값이 없으며, 객체의 초기화를 담당한다. 사용 시 생성자의 이름은 반드시 클래스의 이름과 동일해야 하고, 자료형은 생략한다.
- ③ val의 값 "Spark"를 Vehicle 클래스의 변수 name과 Car 클래스의 변수 name에 저장하고, Car()를 호출했던 다음 줄인 ④번으로 이동한다.
 - ※ **super** : 상속 관계에 있는 부모 클래스를 가리키는 예약어로, 여기서는 Vehicle 클래스를 가리킨다.
- ④ 객체 변수 obj의 getName() 메소드를 호출한다.
 - ※ 메소드의 이름이 동일해도 '인수의 자료형과 개수'가 다르면 서로 다른 메소드이다. 때문에 getName() 메소드는 자식 클래스의 getName(String val) 메소드에 의해 재정의(오버라이딩)된 메소드가 아니다.
- ⑤ getName() 메소드의 시작점이다.
- ⑥ 문자열 "Vehicle name : "에 변수 name에 저장된 값 "Spark"를 붙여 메소드를 호출했던 ⑦번으로 반환한다.
- ⑦ ⑥번에서 반환받은 값을 출력하고 프로그램을 종료한다.

결과 Vehicle name : Spark

[문제 14]

※ 다음 중 밑줄이 표시된 내용은 반드시 포함되어야 합니다.

변수나 함수의 이름 앞에 데이터 타입을 명시하는 코딩 규칙

[문제 15]

30

[해설]

```
public class Test{
    public static void main(String[] args){
        ❶      int a = 0, sum = 0;
        ❷      while (a < 10) {
        ❸          a++;
        ❹          if (a%2 == 1)
        ❺              continue;
        ❻          sum +=a;
                }
        ❼      System.out.println(sum);
    }
}
```

모든 Java 프로그램은 반드시 main() 메소드에서 시작한다.

- ❶ 정수형 변수 a와 sum을 선언하고 각각 0으로 초기화한다.
 - ❷ a가 10보다 작은 동안 ❸~❻번을 반복 수행한다.
 - ❸ 'a = a + 1;'과 동일하다. a의 값에 1을 누적시킨다.
 - ❹ a%2, 즉 a를 2로 나눈 나머지가 1이면 ❺번을 수행하고, 아니면 ❻번으로 이동한다.
 - ❺ while문의 시작점인 ❷번으로 제어를 이동시킨다.
 - ❻ sum에 a의 값을 누적시킨다.
- 반복문 실행에 따른 변수들의 값의 변화는 다음과 같다.

a	sum
0	0
1	
2	2
3	
4	6
5	
6	12
7	
8	20
9	
10	30

- ❼ sum의 값을 출력하고 커서를 다음 줄의 처음으로 옮긴다.

결과 30

[문제 16]

※ 다음 중 하나를 쓰면 됩니다.

OSPF(Open Shortest Path First Protocol)

[문제 17]

- ① Point to Point ② Hub & Spoke

[문제 18]

※ 다음 중 밑줄이 표시된 내용은 반드시 포함되어야 합니다.

누구나 쉽게 이해하고 사용할 수 있어야 한다는 설계 원칙

[문제 19]

※ 다음 중 밑줄이 표시된 내용은 반드시 포함되어야 합니다.

객체 생성 시 자동으로 호출되는 메소드로, 주로 객체의 초기화 용도로 사용된다.

[문제 20]

- ① ALTER ② ADD

[풀이]

ALTER TABLE 학생	수정할 테이블의 이름은 <학생>이다.
ADD 주소 VARCHAR(20);	가변 길이의 문자 20자리인 '주소' 속성을 추가한다.