lakedata.tistory.com

[정보처리기사 실기] 2025년 1회 기출문제 풀이

12-15 minutes

2025년 1회 정보처리기사 실기 기출 풀이과정을 정리해보았습니다. 2025년 1회 실기 합격률은 15.1%이였습니다.

01. 다음은 네트워크 보안과 관련된 공격 기법에 대한 설명이다. 다음 설명을 읽고 물음에 답하시오.

> 네트워크상에서 클라이언트와 서버 간에 인증된 통신이 이루어지고 있을 때, 공격자가 중간에서 인증에 필요한 특정 정보를 가로채어 사용자의 권한을 탈취하는 공격을 ( )(이)라고 한다. 이 공격에 성공하면 공격자는 정상 사용자의 권한을 얻어 불법적으로 서버에 접근하거나 정보를 유출할 수 있다. 대표적인 공격 기법으로는 네트워크 패킷을 스니핑하여 인증 정보를 탈취하거나, 웹 브라우저에 저장된 인증용 쿠키를 탈취하여 이용하는 방식이 있다.

더보기

정답: 세션 하이재킹

세션을 가로채다

세션 하이재킹 (Session Hijacking)

서버에 접속하고 있는 클라이언트들의 세션 정보를 가로채는 공격

Reset 패킷을 통해 강제 종료시킨 후 재연결 시 침임자에게 연결

02. 다음은 관계형 데이터베이스의 무결성 제약 조건에 관한 설명이다. 각 설명에 알맞은 무결성 제약 조건을 쓰시오.

> (①) 기본키(Primary Key)를 구성하는 모든 속성은 절대로 NULL 값이나 중복된 값을 가질 수 없다는 규칙이다.
> (②) 외래키(Foreign Key)의 값은 반드시 참조하고 있는 테이블의 기본키 값으로 존재하거나 NULL이어야 한다는 규칙이다.
> (③) 특정 속성(컬럼)에 대해 미리 정의된 형식과 범위 내에서만 데이터가 입력되도록 제한하는 규칙이다.

더보기

정답: 개체 무결성 제약조건, 참조 무결성 제약조건, 도메인 무결성 제약조건

보기) 개체 무결성 제약조건, 참조 무결성 제약조건, 도메인 무결성 제약조건

null값 허용여부 O(속성개수) X O(0~N)

■ 무결성 (Integrity)

개체 무결성	. 기본키를 구성하는 어떤 속성도 NULL/중복 값 가질 수 없음 . 기본키의 속성 값이 NULL값이 아닌 원자 값을 갖는 성질
도메인/속성 무결성	. 릴레이션 내 튜플들이 각 속성의 도메인에 지정된 값만 가짐
참조 무결성	. 외래키는 NULL 또는 참조 릴레이션의 기본키 값과 동일 . 릴레이션은 참조할 수 없는 외래키 값을 가질 수 없음
사용자 정의 무결성	. 속성 값들의 사용자가 정의한 제약 조건에 만족
데이터 무결성 강화	. 데이터 특성에 맞는 적절한 무결성을 정의하고 강화 . 강화 방법 : 제약조건, 어플리케이션, 데이터베이스 트리거

출처:꿈꾸는라이언

03. 다음 설명을 읽고, 이에 해당하는 오류 검출 방식을 쓰시오.

네트워크 환경에서 데이터를 전송할 때, 송신 측은 전송하려는 데이터에 특정 다항식을 적용하여 얻은 나머지 값을 데이터와 함께 전송하고, 수신 측에서는 동일한 다항식으로 나누어 나머지를 비교하여 데이터의 손상 여부를 판단한다. 이러한 방식으로 데이터 전송 중 발생할 수 있는 오류를 효과적으로 검출하는 방법을 ( ) (이)라고 한다. 이 방식은 이더넷과 같은 네트워크 통신 프로토콜에서 널리 사용된다.

더보기

정답: CRC(Cyclic Redundancy Check, 순환 중복 검사)

※ 오류 검출 방식

해밍 코드	. 수신측에서 오류 검출 후 자동 수정 / 1비트 오류 수정 가능 . 검출 가능 최대 오류 수 = 해밍 거리 – 1
상승 코드	. 순차적/한계값 디코딩 사용 / 여러 비트의 오류 수정 가능
패리티 검사	. 7~8개 비트로 구성된 데이터 블록 끝에 특정 패리티 비트 (짝수/홀수)를 추가하여 오류 검출
순환 중복 검사 (CRC)	. 다항식을 통해 산출된 값으로 오류 검사 (집단 오류 해결) . 데이터 뒤 오류 검출코드 FCS (Frame Check Sequence) 추가
블록 합	. 짝수개 비트 오류 검출 불가한 패리티 검사를 개선한 방법 . 데이터 블록의 수평/수직 패리티 비트 추가

04. 다음 설명을 읽고, 괄호 안에 들어갈 가장 적합한 용어를 쓰시오.

컴퓨터 사용자에게 악성코드에 감염되었다거나, 보안 위협이 존재한다는 허위 경고를 띄워 사용자의 불안을 유발하고, 이를 해결하기 위한 가짜 보안 소프트웨어를 결제하도록 유도하여 금전적 이득을 취하는 악성 소프트웨어를 ( )(이)라고 한다. 사용자는 실제로는 감염되지 않은 상태임에도 불구하고, 이러한 허위 경고에 속아 결제 및 개인 정보를 탈취당할 수 있어 주의가 필요하다

더보기

정답: 스캐어웨어

- 허위로 악성 감염 코드 경고 등을 통해 사용자를 깜짝 놀라게 유도, 자사 백신등을 설치하게 하는 악성코드
- 사용자를 겁먹게 한다는 힌트가 문제에 적혀있었음

05. 다음은 JAVA 코드이다. 출력 결과를 쓰시오.

```
int a= 5, b = 0;
try{
    system.print.ln(a/b);
}
    catch(ArithmeticException e)
{
    system.printf( “출력1 “);
}
    catch(ArrayIndexOutOfBoundsException? e)
{
    system.printf( “출력2” );
}
    catch(NumberFormatException? e)
{
    system.printf( “출력3 “);
}
    catch(Exception e)
{
    system.printf( “출력4” );
}
Finally
{
    printf( “출력5 “);
}
```

더보기

정답: 출력1출력5

5/ 0해서 ArithmeticException 산술 오류

finally구문 실행해야 하고, print()라서 줄바꿈 없음
띄어쓰기도 없었습니다

자바의 주요 예외 클래스

1)IOException

입출력 작업 중 발생하는 예외

예: 파일 읽기 / 쓰기 오류

2) SQLException

데이터베이스 작업 중 발생하는 예외

3) NullPointerException

객체가 null인데 메서드 호출이나 필드 접근을 시도할 때 발생

4) ArrayIndexOutOfBoundsException

배열의 유효 범위를 벗어난 인덱스에 접근할 때 발생

5) ArithmeticException

잘못된 산술 연산(예: 0으로 나누기) 시 발생

06. 다음 네트워크 프로토콜에 대한 설명을 읽고, 빈칸 (1), (2)에 들어갈 올바른 프로토콜의 이름을 각각 쓰 시오.

> (1)(이)란 IP 주소를 가지고 있는 호스트가 네트워크상에서 데이터를 전송할 때, 수신 호스트의 IP 주소 는 알고 있지만, 해당 IP 주소에 대응하는 물리적인 MAC 주소를 모르는 경우 이를 얻기 위해 사용하는 프 로토콜이다.
>
> (2)(이)란 네트워크에 연결된 호스트가 자신의 MAC 주소는 알고 있지만, IP 주소를 모르는 경우 서버에 질의하여 자신의 IP 주소를 얻기 위해 사용하는 프로토콜이다. 이 프로토콜은 주로 디스크가 없는 클라이언 트가 부팅 시 자신의 IP 주소를 얻기 위해 사용되었다.

더보기

정답: ARP, RARP

07. 다음 SQL 문제를 분석하고, 실행 결과를 정확히 작성하시오.

<emp> <sal >

id	name	id	incentives
1001	김철수	1002	300
1002	홍길동	1004	300
1004	강감찬	1008	1000
1008	이순신	1009	500

SELECT name, incentive

FROM emp, sal

WHERE emp.id == sal.id AND incentives >= 500 로

더보기

정답:

name | incentives

이순신 | 1000

emp.name | sal.incentives

이순신 | 1000

가 답이 아닌가 했으나 속성명이 중요한게 아니라 값이 중요한거 이기 때문에 기사정답에선 저게 정답인 것 같음

참고)

insert into 테이블명 (컬럼2, 컬럼2) values(값1, 값2)

update 테이블명 set 속성명=데이터 where 조건

delete from 테이블명 where 조건

문법순서: 셀프웨구해요

문장실행순서:

5 select

1 from

2 where

3 group by

4 having

5 order by

08. 다음은 관계형 데이터베이스 관련 용어에 대한 설명이다. 각 설명과 올바르게 매칭되는 용어를 보기에서 찾아 기호를 쓰시오.

1. 하나의 테이블에 존재하는 속성(Attribute)의 개수
2. 하나의 테이블에 존재하는 튜플(Tuple)의 개수
3. 다른 테이블의 기본 키(Primary Key)를 참조하는 키
4. 하나의 속성이 가질 수 있는 값의 범위

<보기>

ㄱ. degree ㄹ. foreign key ㅅ. candidate key ㄴ. cardinality ㅁ. attribute ㅇ. tuple ㄷ. primary ㅂ. domain

더보기

정답:ㄱ, ㄴ, ㄷ, ㄹ, ㅁ

관계형DB 관련용어 매칭하는 문제

* 속디차: 속성(열) = 디그리 = 차수

* 행튜카: 튜플(행) = 카디널리티

09. 다음 네트워크와 서브넷 마스크가 주어졌을 때, 브로드캐스트로 데이터를 전송할 경우 수신 가능한 IP 주소를 보기에서 모두 고르시오.

네트워크 주소 : 192.168.35.10

서브넷 마스크 : 255.255.252.0

<보기>

ㄱ. 192.168.32.100

ㄴ. 192.168.34.50

ㄷ. 192.168.35.200

ㄹ. 192.168.33.25

ㅁ. 192.168.35.99

ㅂ. 192.168.36.10

ㅅ. 192.168.31.150

ㅇ. 192.168.38.200

ㅈ. 192.168.39.10

더보기

정답: ㄱ, ㄴ, ㄷ, ㄹ, ㅁ

255.255.252

서브넷 계산 - 브로드캐스트하면 받을 수 있는 IP들

IP: 192.168.35.10 → 192.168.0010 0011.00001010

서브넷: 255.255.252.0 → 255.255.1111 1100.00000000

- 세 번째 옥텟(35)에 주목:
    - 35 = 00100011
    - 252 = 11111100

255는 11111111이기 때문에 252인 세 번째 옥텟부터 보면 된다

세 번째 옥텟 고정(128~4), 나머지 구간(2~1)의 최소 00 ~ 최대 11가 된다.

네 번째 옥텍 최소 0, 최대 255

	128	64	32	16	8	4	2	1
35	0	0	1	0	0	0	1	1
252	1	1	1	1	1	1	0	0
AND	0	0	1	0	0	0	0	0

그래서 **IP 대역:** 192.168.32.0 ~ 192.168.35.255 인데

사용 가능한 호스트 IP: 192.168.32.1 ~ 192.168.35.254

총 호스트 수: 2^{10} - 2 = **1022개**

네트워크 주소는: 192.168.32.0

브로드캐스트 주소는: 192.168.35.255 는 빼준다.

10. 다음은 C언어 코드이다. 출력 결과를 쓰시오.

```
#include <stdio.h>

int main() {
    char arr[5] = {'B', 'A', 'D', 'E'};
    char c = 'C';
    int i;

    printf("%d₩n", arr[3] - arr[1]);

    for (i = 0; i < 4; i++) {
        if (arr[i] > c)
            break;
    }
```

```
        char temp = arr[i];
        arr[i] = c;
        i++;

        for (; i < 5; i++) {
            char temp2 = arr[i];
            arr[i] = temp;
            temp = temp2;
        }

        for (i = 0; i < 5; i++) {
            printf("%c", arr[i]);
        }

        return 0;
    }
```

더보기

정답:

4

BACDE

```
        printf("%d₩n", arr[3] - arr[1]);
```

아스키코드

C D

arr B A Ð Ɛ E

i ~~2~~ ~~3~~ 4

t ~~Ð~~ E

t2 E

t2 \0

*temp2는 지역변수라 for문 할때마다 새로 생성

11. 다음은 C언어 코드이다. 출력 결과를 쓰시오.

```
#include <stdio.h>
#include <stdlib.h>

int A[9] = {8, 9, 3, 7, 2, 8, 3, 1, 4};

void func(int **res, int cols, int rows) {
    for (int i = 0; i < (cols * rows); i++) {
        int r = (i + 1) % (rows * cols);
        res[r / cols][r % cols] = A[i];
    }
}

int main() {
    int **res;
    int cols = 3;
    int rows = 3;
    int sum = 0;

    res = (int **)malloc(sizeof(int *) * rows);
    for (int i = 0; i < rows; i++) {
        res[i] = (int *)malloc(sizeof(int) * cols);
    }

    func(res, cols, rows);

    for (int i = 0; i < rows * cols; i++) {
        sum += res[i / cols][i % cols] * (i % 2 == 0 ? 1 : -1);
    }

    printf("%d", sum);

    for (int i = 0; i < rows; i++) {
        free(res[i]);
    }
```

```
    free(res);


    return 0;
}
```

더보기

정답: 13

1000번지 res

4
8 9
(100번지)

3
7 2
(200번지)

8
3 1
(300번지)

r i

res[0][1]=8 1 0

res[0][2] = 9 2 1

res[1][0] = 3 3 2

res[1][1] = 7 4 3

res[1][2] = 2 5 4

res[2][0] = 8 6 5

res[2][1] = 3 7 6

re[2][2] = 1 8 7

res[0][0] = 4 9 8

i

0 res[0][0] 4 1 4

1 res[0][1] 8 -1 -8

2 res[0][2] 9 1 9

3 res[1][0] 3 -1 -3

4 res[1][1] 7 1 7

5 res[1][2] 2 -1 -2

6 res[2][0] 8 1 8

7 res[2][1] 3 -1 -1

8 res[2][2] 1 1 1

12. 아래의 설명을 읽고 빈칸에 들어갈 결합도를 쓰시오.

(1) 하나의 모듈이 다른 모듈 내부에 있는 변수나 기능을 직접 접근하거나 사용하는 경우를 의미하며, 가장 강한 결합도로 모듈 간 독립성을 크게 해친다.
(2) 모듈 간의 인터페이스로 단순 자료형이 아니라 배열(Array), 객체(Object), 구조체와 같은 복합적인 자료 구조가 전달되는 경우를 의미한다. 호출하는 모듈이 필요하지 않은 정보까지 함께 전달받아 모듈의 독 립성을 저하시킬 수 있다.
(3) 두 개 이상의 모듈이 파라미터로 데이터를 전달하지 않고, 외부에 선언된 전역 변수를 참조하거나 수정하면서 서로 상호작용하는 경우이다. 모듈 간 결합도가 강하고 모듈 독립성이 저하된다.

더보기

정답: 내용, 스탬프, 공통

(1) 하나의 모듈이 다른 모듈 내부에 있는 변수나 기능을 직접 접근하거나 사용하는 경우를 의미하며, 가장 강한 결합도로 모듈 간 독립성을 크게 해친다.
(2) 모듈 간의 인터페이스로 단순 자료형이 아니라 배열(Array), 객체(Object), 구조체와 같은 복합적인 자료 구조가 전달되는 경우를 의미한다. 호출하는 모듈이 필요하지 않은 정보까지 함께 전달받아 모듈의 독 립성을 저하시킬 수 있다.
(3) 두 개 이상의 모듈이 파라미터로 데이터를 전달하지 않고, 외부에 선언된 전역 변수를 참조하거나 수정하면서 서로 상호작용하는 경우이다. 모듈 간 결합도가 강하고 모듈 독립성이 저하된다.

13. 다음은 JAVA 코드이다. 출력 결과를 쓰시오.

```
class parent{
    static int total = 0;
    int v = 1;
    parent(){
        total += (++v);
        show();
```

```
        }
        public void show(){
            total += total;
        }
    }
    class child extends parent{
        int v = 10;
        child(){
            v += 2;
            total += (v++);
            show();
        }
        @override
        public void show(){
            total += total*2;
        }
    }

    class Main {
        public static void main(String[] args) {
            new child();
            System.out.println(parent.total);
        }
    }
```

더보기

정답: 54

pt ~~0~~ ~~2~~ ~~6~~ ~~18~~ 54

new Child(); 에서

Child생성자 전에 부모생성자를 명시하지 않아 부모 기본생성자를 호출

하고 showValue();오버라이딩 되어서 자식꺼 사용해고 다시 Child생성자로 간다.

14. 다음 설명을 읽고, 어떤 디자인 패턴인지 보기에서 골라 쓰시오.

- 이 패턴은 Wrapper(래퍼)라고도 불린다.
- 서로 호환되지 않는 두 개의 클래스 사이에서 한 클래스의 인터페이스를 다른 클래스가 기대하는 형태로 변환하여, 기존에 호환되지 않았던 클래스들이 함께 작동할 수 있도록 해주는 구조적 패턴이다.
- 이 패턴을 활용하면 이미 존재하는 클래스를 수정하지 않고도 기존 코드와 새로운 코드의 통합을 용이하게 만들 수 있다.

<보기>

ㄱ. Singleton ㄴ. Factory Method ㄹ. Adapter ㅁ. Decorator ㅅ. Visitor ㄷ. Abstract Factory

ㅂ. Observer

더보기

정답: Adapter

호환성

디자인 패턴

- Wrapper라고도 불림

- 다른 클래스가 이용할 수 있도록 인터페이스 변환해주는 패턴

<참고>

생성 패턴 - 생빌 프로 팩앱싱

: 생성 - 빌더 / 프로토타입 / 팩토리 메서드 / 앱스트랙 팩토리 / 싱글톤

구조 패턴 - 구 브데 퍼플 프록컴어

: 구조 - 브리지 / 데코레이터 / 퍼사이드 / 플라이 웨이트 / 프록시 / 컴포지트 / 어댑터

행위 패턴 - 행 미인이 템옵 스터 비커 스트메체

: 행위 - 미디에이터 / 인터프리터 / 이터레이터 / 템플릿 / 옵져버 / 스테이트 / 비지터 / 커맨트 / 스트레티지 / 메멘토 / 체인 오브 리스판서빌리티

15. 구문(문장) 커버리지를 수행하려고 한다. 아래 제어 흐름도의 빈칸에 맞는 소스코드를 쓰고, 구문 커버리지의 경로를 쓰시오.

더보기

정답:

1 total = 0;

2 a >= 0

3 a % 2 == 0

4 total = total + a;

5 a = a 2;

6 return total;

구문 커버리지 : 1 → 2 → 3 → 4 → 5 → 2 → 6

16. 다음은 JAVA 코드이다. 출력 결과를 쓰시오.

```
static int func(int[] a, int st, int end) {
        if(st>=end) return 0;
        int mid = (st+end)/2;
```

```
                return a[mid] + Math.max(func(a,st, mid), func(a, mid+1,end));
        }

        public static void main(String[] args) {
                int[] a=  new int[] {3, 5, 8, 12, 15};
                int result = func(a, 0, a.length-1);
                System.out.println(result);
        }
```

17. 다음은 파이썬 코드이다. 출력 결과를 쓰시오.

```
class Node:
    def __init__(self, value):
        self.value = value
        self.children = []

def tree(li):
    nodes = [Node(i) for i in li]
    for i in range(1, len(li)):
        parent_index = (i - 1)
        nodes[parent_index].children.append(nodes[i])
    return nodes[0]

def s(node, level=0):
    if not node:
        return 0

    total = node.value if level % 2 else 0

    for child in node.children:
        total += s(child, level + 1)

    return total

li = [3, 5, 8, 12, 15, 18, 20]
root = tree(li)
```

```
print(s(root))
```

더보기

정답: 13

node.value if level % 2 else 0

3항연산자: 참 if 조건 else 거짓

//는 몫

파이썬은 /하면 정수/정수여도 실수가 반환되는데 //는 정수 반환

18. 다음은 C언어 코드이다. 출력 결과를 쓰시오.

```
#include <stdio.h>
#include <stdlib.h>

struct node {
    int value;
    struct node* next;
};
```

```
struct node* add(struct node* head, int value) {
    struct node* new_node = (struct node*)malloc(sizeof(struct node));
    new_node->value = value;
    new_node->next = head;
    return new_node;
}

struct node *func(struct node *head, int x){

    struct node *prev;
    struct node * cur = head;

    for(cur; cur->vaule != x; cur = cur->next){
       prev = cur;
       prev->next = cur->next

   }
   if(){}
cur->next = head

    return cur;
}

int main() {
    struct node* head = NULL;
    for (int i = 1; i <= 5; i++) head = add(head, i);

    struct node* cur = func(head, 3);
    for (cur=head; cur; cur = cur->next) {
        printf("%d", cur->value);
    }

    return 0;
}
```

19. 다음은 C언어 코드이다. 출력 결과를 쓰시오.

```
#include <stdio.h>
#include <stdlib.h>

typedef struct {
    char name[10];
    unsigned char score[3];
} Node;

int f(unsigned char score) {
    return score & 0xA5;
}

int main() {
    Node *nodes = (Node *)malloc(2 * sizeof(Node));

    nodes[0] = (Node){"Kim", {0xF0, 0xF5, 0xDB}};
    nodes[1] = (Node){"Lee", {0xED, 0xD3, 0xF2}};

    int result = 0;

    for (int i = 0; i < 2; i++) {
        for (int j = 0; j < 3; j++) {
            result += f(nodes[i].score[j]);
        }
    }

    printf("%dWn", result);

    free(nodes);
    return 0;
}
```

더보기

정답: 908

2진수 : 0b(binary)

8진수 : 00(octal)

16진수 : 0x(hexadecimal

10진수 : Decimal

20. 다음은 JAVA 코드이다. 출력 결과를 쓰시오.

```
class Main {
    static int add(String s) {
        int t = Integer.valueOf(s);
        if (t <= 1) return t;
        return add(t - 1) + add(t - 3);
    }

    static int add(int i) {
        if (i <= 1) return i;
        return add(i - 1) + add(i - 2);
    }

    public static void main(String[] args) {
        System.out.println(add("5"));
    }
}
```

더보기

정답: 4

재귀함수

문자열 "5" 입력 → add(String s) 호출

그 다음 재귀는 Integer.valueOf(s)로 int형으로 변환되어 add(int i) 호출