

1. Java 에서의 상속에 대한 특징 중 틀린 것을 고르시오. (②)
 - ① 생성자와 초기화 블록은 상속되지 않는다.
 - ② 자식클래스의 멤버 개수는 부모 클래스와 항상 같아야만 한다.
 - ③ 보다 적은 양의 코드로도 새로운 클래스를 만들 수 있다.
 - ④ 코드의 중복을 제거하여 프로그램의 생산성과 유지보수가 좋아진다.
2. 다음 중 오버로딩(Overloading)이 성립하기 위한 조건이 아닌 것을 2 개 고르시오. (③, ④)
 - ① 메서드의 이름이 같아야 한다.
 - ② 매개변수의 개수나 타입이 달라야 한다.
 - ③ 리턴 타입이 달라야 한다.
 - ④ 매개변수의 이름이 달라야 한다.
3. Java 에서의 오버라이딩(Overriding)의 성립 조건 중 틀린 것을 모두 고르시오. (④, ⑤)
 - ① 조상의 메서드와 이름이 같아야 한다.
 - ② 매개변수의 수와 타입이 모두 같아야 한다.
 - ③ 리턴 타입이 같아야 한다.
 - ④ 접근 제어자는 조상의 메서드보다 좁은 범위로만 변경할 수 있다.
 - ⑤ 조상의 메서드보다 더 많은 수의 예외를 선언할 수 있다.
4. 추상(abstract) 클래스와 인터페이스(interface)의 용도에 대해 기술하시오
 추상클래스 : (자식 클래스에 오버라이딩에 대한 강제성을 부여하여 일관된 인터페이스를 제공한다.)
 인터페이스 : (상수를 기반으로 모든 기능을 공통화 해서 강제성을 부여하여 공통된 인터페이스를 제공하고 자바의 단일 상속의 단점을 극복하기 위해 쓴다.)
5. 자바에서 사용되는 상속의 유형으로 틀린 것을 모두 고르시오. (④, ⑤)
 - ① class 클래스명 extends 클래스명 {}
 - ② class 클래스명 implements 인터페이스명 1, 인터페이스명 2{}
 - ③ interface 인터페이스명 extends 인터페이스명 1, 인터페이스명 2{}
 - ④ interface 인터페이스명 extends 클래스명 {}
 - ⑤ class 클래스명 extends 클래스명 1, 클래스명 2
 - ⑥ class 클래스명 extends 클래스명 implements 인터페이스명 1, 인터페이스명 2 {}
6. 서브클래스에서 슈퍼클래스의 메소드 오버라이딩시 사용하는 어노테이션(Annotation)은 ?
 (@Override)
7. 아래의 소스 18 번 줄에 추가할 메소드로 적당한 것을 2 개 고르시오. (②, ④)

```

10. public abstract class Employee {
11.     protected abstract double getSalesAmount();
12.
13.     public double getCommision() {
14.         return getSalesAmount() * 0.15;
15.     }
16. }
17. public class Sales extends Employee {
18.     // insert method here
19.
20. }

```

- ① double getSalesAmount() { return 1230.45; }
- ② public double getSalesAmount() { return 1230.45; }
- ③ private double getSalesAmount() { return 1230.45; }
- ④ protected double getSalesAmount() { return 1230.45; }
- ⑤ protected abstract double getSalesAmount() { return 1230.45; }

8. 접근 제어자의 조합에 대한 설명 중 틀린 것을 고르시오. (④)

- ① 메소드에 static 과 abstract 를 함께 사용할 수 없다.
- ② 클래스에 abstract 와 static 을 동시에 사용할 수 없다.
- ③ abstract 메소드의 접근 제어자가 private 일 수 없다.
- ④ 메소드에 public 과 final 을 같이 사용할 수 없다.

9. 아래의 변수를 상수필드로 처리하고자 한다. 빈칸에 적당한 키워드를 채우시오
(public) (static) (final) String MAKER = "KOREA";

10. interface 의 특징에 대한 설명 중 맞는 것을 고르시오. (①)

- ① 모든 메소드가 추상 메소드인 클래스이다.
- ② 모든 인터페이스의 메소드는 묵시적으로 private 이며 abstract 이다.
- ③ 변수는 가질 수 없다.
- ④ 객체 생성도 안되고 reference 변수로도 사용이 불가하다.

11. 메소드 작성시 사용할 수 없는 예약어는? (④)

- ① static
- ② final
- ③ abstract
- ④ transient

12. 다음 중 연산 결과가 true 가 아닌 것을 고르시오. (⑤)

```

class Car { }
class FireEngine extends Car implements Movable { }
class Ambulance extends Car { }
FireEngine fe = new FireEngine();

```

- ① fe instanceof FireEngine
- ② fe instanceof Movable
- ③ fe instanceof Object
- ④ fe instanceof Car
- ⑤ fe instanceof Ambulance

13. 다음 프로그램의 실행결과를 쓰시오. (ABC123)

```

class Exercise29 {

    public static void change(String str) {
        str += "456";
    }

    public static void main(String[] args) {
        String str = "ABC123";
        System.out.println(str);
        change(str);
        System.out.println("After change:"+str);
    }

}

```

14. 다음과 같이 class 들이 정의되어 있다
compile 시에 Error 를 발생시키는 것을 모두 고르시오. (①, ④, ⑤)

```

abstract class Mammal { }
class Dog extends Mammal { }
class Cat extends Mammal { }

```

- ① Mammal m1 = new Mammal ();
- ② Mammal m2 = new Dog ();
- ③ Mammal m3 = new Cat ();

④ Dog d1 = new Mammal ();

⑤ Dog d2 = new Cat ();

15. 예외(Exception)를 해결하는 방법을 모두 기술하시오.

① : (throws로 위임)

② : (try-catch(-finally))

③ : (try-with-resource)

16. Checked Exception 에 해당하는 클래스는 ? (②)

① RuntimeException

② IOException

③ ArithmeticException

④ ArrayStoreException

17. 조건 상황에 따라 예외를 발생시키는 키워드는 ? (throw)

18. 예외처리(Exception Handling)에서 catch 구문을 여러 번 사용할 경우 틀린 것은? (

④)

① 상속 관계에서 같은 레벨(형제 관계)에 해당하는 예외클래스간에는 사용 순서가 상관없다.

② 최하위 예외클래스를 가장 먼저 제시해야 한다.

③ 상위(부모) 예외클래스를 후손 클래스보다 아래에 두어야 한다.

④ Exception 클래스를 가장 먼저 제시해야 한다.

19. 상속 관계에 있는 클래스 간의 생성자 호출시 에러가 발생하는 것은? (③)

```
class A {  
    private int no;  
    public A() { }  
    public A(int no) { this.no = no; }  
}  
class B extends A {  
    private String name;  
}
```

① public B() { super(); }

② public B(int no) { super(no); }

③ public B(int no, String name) { this(name); super(no); }

④ public B(String name) {

this.name = name; }

20. 다형성(Polymorphism)에 대해 설명하시오.
(부모 클래스를 참조하여 여러 형태의 객체를 가질 수 있는 성질이다.)
21. 다음의 final 키워드 사용 대상에 따른 특징을 설명하시오.
① final class : (상속 불가)
② final method : (오버라이딩 불가)
③ final variable/field : (값 변경 불가)
22. 다형성에 적용되는 기능이 아닌 것은 ? (④)
① Up Casting
② Down Casting
③ Dynamic Binding
④ Auto Boxing
23. 레퍼런스가 참조하는 인스턴스의 클래스 타입을 확인할 때 사용하는 연산자는 ?
(instanceof)
24. 다형성(Polymorphism)을 사용했을 때의 장점을 2 개 이상 기술하시오.
① (유지보수 용이)
② (코드 재사용성 증가)
③ (확장성 증가)
25. 자바 컬렉션 프레임워크가 가지는 특징들을 간단히 기술하시오.
① Set : (순서가 없는 데이터의 집합으로, 데이터 중복을 허용 하지 않는다.)
② List : (순서가 있는 데이터의 집합으로, 데이터 중복을 허용한다.)
③ Map : (키-값 쌍으로 구성되는 순서가 없는 데이터의 집합으로, 키 중복은 허용 하지만, 값 중복은 허용하지 않는다.)
26. 객체 입/출력을 위해서 클래스에 반드시 적용해야 하는 처리내용은?
(Serializable 인터페이스 구현)
27. 아래의 클래스 중 기본 스트림 클래스가 아닌 것을 고르시오. (④)
① FileInputStream
② ByteArrayInputStream
③ CharArrayWriter
④ InputStreamReader
28. BufferedReader 클래스를 사용하여 키보드와 입력 바이트 스트림을 생성하는 구문을 작성하시오.
(`BufferedReader br = new BufferedReader(new InputStreamReader(System.in));`)
29. 아래 소스 코드의 내용을 완성하시오.

```

//예외처리용 클래스를 작성함
public class ZeroException {
    public ZeroException(String message) {
        super(message);
    }
}

public class Calculator {
    public double divide(double a, double b) {
        //나눌 수 b가 0일 경우 ZeroException 발생시키는 소스 작성함
        if (b == 0) {
            throw new ZeroException("0으로 나눌 수 없습니다.");
        }
        return a / b;
    }
}

public class Exam29 {
    public static void main(String[] args) {
        //divide() 메소드 사용과 관련된 예외처리 코드 작성함, throws 하지 말 것
        try {
            System.out.println(new Calculator().divide(12.5, 0));
        } catch (ZeroException e) {
            System.out.println("예외 발생: " + e.getMessage());
        }
    }
}

```

30. 아래의 multi catch 구문을 예외는 구분하되 하나의 catch 구문으로 변경한 코드를 오른쪽 칸에 작성하시오. (단, Exception 으로 처리하지 말 것)

```

try {
    Properties prop = new Properties();
    prop.load(new FileReader("dbSource.txt"));
    // 중간 생략
} catch (IOException e) {

```

```

        e.printStackTrace();
    } catch (SQLException e) {
        e.printStackTrace();
    }
}

```

```

try {
    Properties prop = new Properties();
    prop.load(new FileReader("dbSource.txt"));
    // 중간 생략
} catch (IOException | SQLException e) {
    e.printStackTrace();
}

```

31. 키는 String 이고 값은 Book 클래스 객체만 저장할 수 있는 HashMap 클래스 생성 구문을 Generics 기능을 사용하여 작성하시오.

(HashMap<String, Book> map = new HashMap<>();)

32. 아래 코드를 finally 를 사용하지 않고, 자동 close 처리되는 try with resource 문으로 변경하시오.

```

FileReader fr = null;
try {
    fr = new FileReader("books.dat");
    중간 생략
} catch (IOException e){
    e.printStackTrace();
} finally{
    try {
        fr.close();
    } catch (IOException e){
        e.printStackTrace();
    }
}

```

```

try (FileReader fr = new FileReader("books.dat")) {
    // 중간 생략
}

```

```
} catch (IOException e) {  
    e.printStackTrace();  
}
```

33. 아래 코드에서 맵에 저장된 정보를 연속으로 출력 처리하기 위한 구문을 완성하시오

```
HashMap<String, Book> map;  
//객체 생성과 객체 정보 저장 코드 중간 생략하였음.  
//맵에 저장된 정보를 연속으로 출력 처리하기 위한 소스를 작성함  
//출력 : 키 = 객체 정보 (참고 : Book 클래스에 toString() 메소드 오버라이딩 되어 있음)  
//처리 메소드는 각자 선택할 것  
for (Map.Entry<String, Book> entry : map.entrySet()) {  
    System.out.println(entry.getKey() + " = " + entry.getValue());  
}
```