프로그래밍언어론

1. 다음 C 프로그램의 실행 결과는?

```
#include <stdio.h>
void main() {
        int x = 1;
        switch (x+1) {
                case 1: x++;
                case 2: x += 2;
                case 3: x += 3;
                default: break;
        }
        printf("%d\n", x);
}
```

① 1 ② 2 ③ 3 ④ 4 ⑤ 6

2. Java 언어의 메소드 재정의(method overriding)에 대한 설명으로 옳지 않은 것은?

① 재정의된 메소드는 자식 클래스에 상속되지 않는다.
② 한 클래스 내에 여러 개의 재정의된 메소드가 있을 수 있다.
③ 메소드 재정의는 상속된 메소드를 다시 정의하는 것이다.
④ final로 선언된 메소드는 재정의 할 수 없다.
⑤ static으로 선언된 메소드는 재정의 할 수 있다.

3. EBNF로 나타낸 다음 문법을 BNF 표기로 바꾸면?

```
<exp> ::= a { +a }
```

① <exp> ::= a
② <exp> ::= a+a | a
③ <exp> ::= <exp>+a
④ <exp> ::= <exp>+a | a
⑤ <exp> ::= <exp>+<exp>

4. 다음 중 정규식(regular expression) $(a|b)^*c(d|e)^*$로 표현할 수 없는 스트링은?

① aac ② abc
③ bcde ④ bacde
⑤ abde

5. 다음 Java 프로그램의 실행 결과는? 메소드 m2()의 if 문에서 cond는 참이라고 가정.

```
class M {
    public static void main(String args[]) {
    try {
      m1();
    } catch (Exception e) { System.out.print(" 1 ");  }
 }

static void  m1() throws Exception {
     try {
          m2();
          System.out.print(" 2 ");
     } catch(Exception e) {
          System.out.print(" 3 ");
          throw e;
     }
 }
static void m2() throws Exception {
     if( cond )
         throw new Exception();
     System.out.print(" 4 ");
 }
}
```

① 4 3 2 1 ② 4 3 1
③ 4 2 1 ④ 3 2 1
⑤ 3 1

6. 다음과 같이 C 배열이 선언되어 있을 때 틀린 것은?

```
int ARR[10][10];
```

① ARR와 &ARR[0]는 같은 의미이다.
② &ARR[0]와 &ARR[0][0]는 같은 의미이다.
③ *(ARR[0]+10)와 ARR[1][0]는 같은 의미이다.
④ *(&ARR[1][0]-1)와 ARR[0][9]는 같은 의미이다.
⑤ *(&ARR[0][0]+1)와 ARR[0][1]는 같은 의미이다.

7. 프로그래밍언어의 구현 방법에 대한 설명으로 틀린 것은?

① 고급언어는 저급언어에 비해 인터프리터 구현이 용이하다.
② 인터프리터 방식은 컴파일러 방식에 비해 이식성이 뛰어나다.
③ 컴파일러 방식은 인터프리터 방식에 비해 실행속도가 빠르다.
④ JIT(Just-In-Time) 컴파일 방식은 인터프리터의 단점인 실행효율을 향상시킨 방식이다.
⑤ 하나의 언어에 대해 컴파일방식과 인터프리터 방식을 함께 사용할 수 있다.

8. 다음 Java 프로그램 실행 결과는?

```
class A {
      static int s = 1;
      int x = 0;
      static void f() { s = s + 3;  }
      void g() { x = x + 2; }
}

class C {
    public static void main(String[] args) {
        A a = new A();
        a.g();
        A b = new A();
        b.g();
        a.f();
        System.out.println(a.x + " " + b.x + " " + a.s + " " + b.s);
    }
}
```

① 2 2 4 0　② 4 4 4 4
③ 2 2 4 4　④ 4 4 4 1
⑤ 2 2 4 1

9. 다음 Java 프로그램의 실행 결과는?

```
class C {
        private int a = 1;
        static private int b = 2;
        public int getA() { return a; }
        public static int getB() { return b; }
}
class D extends C {
        private int a = 4;
        static private int b = 6;
        public int getA() { return a; }
        public static int getB() { return b; }
}
public class Test {
        public static void main(String[] args) {
                C obj = new D();
                System.out.println(obj.getA() + obj.getB());
        }
}
```

① 1　② 3　③ 6　④ 7　⑤ 10

10. Java와 C++에 대한 설명으로 올바른 것은?

① Java의 클래스는 다중 상속을 허용한다.
② Java의 인터페이스는 다중 상속을 허용한다.
③ Java의 메소드는 virtual을 이용하여 정적바인딩을 나타낸다.
④ C++의 클래스는 다중 상속을 허용하지 않는다.
⑤ C++의 멤버함수는 virtual을 이용하여 정적바인딩을 나타낸다.

11. 다음 C++ 프로그램의 실행 결과는?

```
#include <iostream.h>
class A {
    public:
    void p() { cout << "A:p "; }
    virtual void q() { cout << "A:q "; }
    void f() {
        p();
        q();
     }
};
class B: public A {
    public:
    void p() { cout << "B:p "; }
    void q() { cout << "B:q "; }
};
main()  {
    A a;
    B b;
    a.f();
    b.f();
    a = b;
    a.f();
}
```

① A:p A:q B:p B:q A:p A:q
② A:p A:q A:p B:q A:p A:q
③ A:p A:q A:p B:q B:p B:q
④ A:p A:q B:p B:q B:p B:q
⑤ A:p A:q A:p A:q A:p A:q

12. 객체지향프로그램의 특징인 추상화(abstraction)의 장점에 대한 설명으로 틀린 것은?

① 객체 사이의 결합도(coupling)를 증가시킨다.
② 객체 표현의 세부 사항 노출을 조절 할 수 있다.
③ 프로그램의 신뢰성(reliability)이 증가한다.
④ 객체의 무결성(integrity)이 증가된다.
⑤ 객체를 사용하는 코드는 객체 표현에 독립적이다.

13. 부프로그램에 전달될 실매개변수(actual parameter)가 식(expression)일 때 실매개변수의 값이 가장 늦게 계산되는 매개변수 전달 방법은?

① 값 전달 방법(call by value)
② 결과 전달 방법(call by result)
③ 참조 전달 방법(call by reference)
④ 이름 전달 방법(call by name)
⑤ 값-결과 전달(call by value-result)

14. 다음 코드에서 x는 4바이트의 길이를 갖는 int 형의 배열이며 주소 4530부터 저장된다고 가정하자. 다음 코드가 실행한 후에 각 식의 값으로 적당하지 않은 것은?

```
for(i=0;i<10;i++)
  x[i] = i+1;
```

① x의 값은 4530이다.
② x+2의 값은 3이다.
③ *(&x[2] +1)의 값은 4이다.
④ *(x+2)의 값은 3이다.
⑤ &x[1]의 값은 4534이다.

15. 바인딩 시간은 언어설계, 언어구현, 컴파일, 링크, 적재, 실행 시간으로 세분화할 수 있다. 다음 설명 중 틀린 것은?

① C 언어에서 선언된 변수의 타입은 실행시간에 바인딩된다.
② Java 언어에서 int 타입의 크기는 언어설계시간에 바인딩된다.
③ C 언어에서 int 타입의 크기는 언어구현시간에 바인딩된다.
④ Java 언어에서 선언된 배열의 크기는 실행시간에 바인딩된다.
⑤ C 언어에서 선언된 배열의 크기는 컴파일 시간에 바인딩 된다.

16. 다음 C 프로그램에 대한 설명으로 올바른 것은?

```
01    void foo(int x) {
02        int y = x;
03    }
04    void main() {
05        int a = 10;
06        int *p, *q;
07        p = (int *) malloc(sizeof(int));
08        q = &a;
09        p = q;
10        foo(*q);
11    }
```

① 02행에서 x와 y는 이명(alias)이다.
② 07행에서 쓰레기(garbage)가 발생한다.
③ 08행에서 허상참조(dangling reference)가 발생한다.
④ 09행에서 쓰레기가 발생한다.
⑤ 10행의 함수 호출에 의해 q와 x는 이명이다.

17. 다음 Java 접근 수정자(access modifier)에 대한 설명으로 옳지 않은 것은?

① private로 선언된 필드는 선언된 클래스 내에서만 사용 가능하다.
② public으로 선언된 필드는 프로그램 내 어디서나 사용 가능하다.
③ protected로 선언된 필드는 선언된 클래스의 자식 클래스에서 사용 가능하다.
④ protected로 선언된 필드는 같은 패키지 내의 다른 클래스에서 사용될 수 있다.
⑤ 접근 수정자 없이 선언된 필드는 선언된 클래스 내에서만 사용 가능하다.

18. 다음 문법에 대한 설명으로 올바른 것은?

```
<expr> ::= <expr>+<term>|<term>
<term> ::= <factor>*<term>|<factor>
<factor> ::= (<expr>)|<id>
```

① 이 문법은 모호하다(ambiguous).
② 연산자 +의 우선순위가 *보다 높다.
③ 연산자 +는 좌우선 결합(left-associative)을 한다.
④ 연산자 *는 좌우선 결합을 한다.
⑤ 이 문법은 정규 문법(regular grammar)이다.

19. 블록구조 언어의 활성레코드(activation record)에 대한 설명으로 틀린 것은?

① 블록이 시작될 때 활성레코드가 스택에 push 된다.
② 블록이 끝나면 해당 블록의 활성레코드가 스택에서 pop 된다.
③ 정적링크(static link)는 스택의 이전 활성레코드의 주소를 저장한다.
④ EP(environment pointer)는 스택의 탑에 있는 활성레코드의 주소를 저장한다.
⑤ 블록 내에 선언된 스택변수의 기억장소는 활성레코드에 할당된다.

20. 다음 C 프로그램의 실행 결과는?

```
#include <stdio.h>
int a[4] = {2, 4, 6, 8};
void swap1(int x, int y) {
    int t = x;
    x = y;
    y = t;
}
void swap2(int *x, int *y) {
    int t = *x;
    *x = *y;
    *y = t;
}
main() {
   swap1(a[0], a[1]);
   printf("%d %d ", a[0], a[1]);
   swap2(a+2, a+3);
   printf("%d %d ", a[2], a[3]);
}
```

① 2 4 6 8　② 4 2 6 8
③ 2 4 8 6　④ 4 2 8 6
⑤ 2 4 8 8