[3. 연산자]

1. 다음 코드를 실행했을 때 출력 결과를 예상하고 작성해보시오.

| int x = 10;  int y = 20;  int z = (++x) + (y--);  System.out.println(z); |
| --- |

정답: 31

1. 다음 코드를 실행했을 때 출력 결과를 예상하고 작성해보시오.

| int score = 85;  String result = (!(score>90))? "가":"나";  System.out.println(result); |
| --- |

정답: 가

1. 534자루의 연필을 30명의 학생들에게 똑같은 개수로 나누어 줄 때 1인당 몇 개를 가질 수 있고, 마지막에 몇 개가 남는지를 구하는 코드입니다. ( )에 들어갈 알맞은 코드를 차례대로 작성해보세요.

| int pencils = 534;  int students = 30; //학생 한 명이 가지는 연필 수  int pencilsPerStudent = ( );  System.out.println(pencilsPerStudent); //남은 연필 수    int pencilsLeft = ( );  System.out.println(pencilsLeft); |
| --- |

정답:

pencils / students

pencils % students

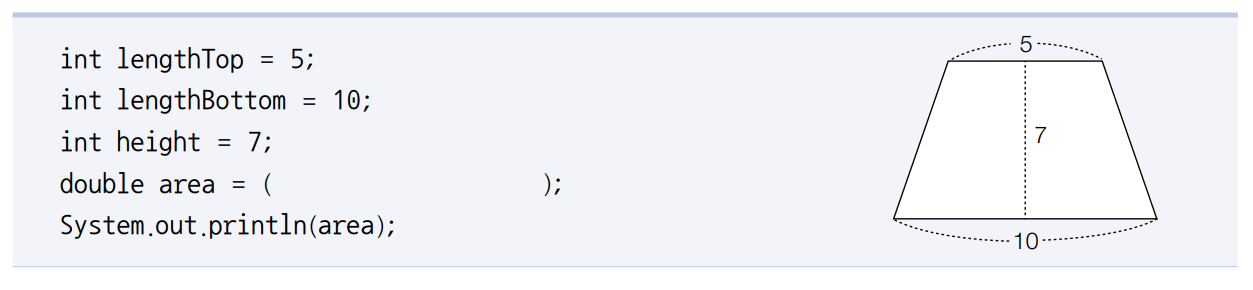
1. 다음은 십의 자리 이하를 버리는 코드입니다. 변수 value의 값이 356이라면 300이 나올 수 있도록 ( )에 알맞은 코드를 작성하세요(산술 연산자만 사용).

| int value = 356;  System.out.println( ); |
| --- |

정답: value / 100 \* 100

\*정수 나눗셈의 특징은 버림이 일어난다.

1. 다음 코드는 사다리꼴의 넓이를 구하는 코드입니다. 정확히 소수 자릿수가 나올 수 있도록 ( )에 들어갈 수 있는 코드를 모두 선택하세요. (정답: 1,2,3,4)



➊ (lengthTop+lengthBottom) \* height / 2.0

➋ (lengthTop+lengthBottom) \* height \* 1.0 / 2

➌ (double)(lengthTop+lengthBottom) \* height / 2

➍ (double)( (lengthTop+lengthBottom) \* height / 2)

1. 다음 코드는 비교 연산자와 논리 연산자의 복합 연산식입니다. 연산식의 출력 결과를 작성해보세요.

| int x = 10;  int y = 5;  System.out.println( (x>7) && (y<= 5) );  System.out.println( (x%3 = = 2) || (y%2 != 1) ); |
| --- |

정답: true, false

1. 다음은 % 연산을 수행한 결과값에 10을 더하는 코드입니다. NaN 값을 검사해서 올바른 결과가 출력될 수 있도록 ( )에 들어갈 코드를 작성해보세요.

| double x = 5.0;  double y = 0.0;  double z = 5 % y;  if ( ) {  System.out.println("0.0으로 나눌 수 없습니다.");  } else {  double result = z + 10;  System.out.println("결과: " + result);  } |
| --- |

정답: Double.isNaN(z)

1. 아래의 코드는 사과를 담는데 필요한 바구니(버켓)의 수를 구하는 코드이다. 만일

사과의 수가 123개이고 하나의 바구니에는 10개의 사과를 담을 수 있다면, 13개의 바구니

가 필요할 것이다. ( )에 알맞은 코드를 넣으시오.

| class Exercise {  public static void main(String[] args) {  int numOfApples = 123; // 사과의 개수  int sizeOfBucket = 10; // 바구니의 크기(바구니에 담을 수 있는 사과의 개수)  int numOfBucket = ( )  System.out.println("필요한 바구니의 수 :"+numOfBucket);  }  } |
| --- |

정답: numOfApples/sizeOfBucket + (numOfApples%sizeOfBucket > 0 ? 1 : 0);

\*사과의 개수(numOfApples)를 바구니의 크기(sizeOfBucket)으로 나눗셈연산(/)을

하면 사과를 담는데 필요한 바구니의 수(numOfBucket)를 구할 수 있다. 정수간의 나눗셈

연산의 특징은 반올림을 하지 않고 버림을 한다는 것이다. 예를 들어 125/10의 결과는 13

이 아니라 12가 된다. 게다가 int와 int간의 이항연산결과는 int이기 때문에, 12.5와 같

은 실수값 결과가 나오지 않는다.

그리고 사과의 개수(numOfApples)를 바구니의 크기(sizeOfBucket)으로 나눴을 때 나머지

가 있으면 하나의 바구니가 더 필요하다. 그래서 나머지 연산자(%)를 이용해서 나눗셈연

산에서 나머지가 발생하는지 확인해서, 나머지가 발생하면 바구니의 개수(numOfBucket)에

1을 더해줘야 한다.

1. 아래는 변수 num의 값에 따라 ‘양수’, ‘음수’, ‘0’을 출력하는 코드이다. 삼항 연산

자를 이용해서 알맞은 코드를 작성하시오.

정답:

class Exercise {

public static void main(String[] args) {

int num = 10;

System.out.println(num > 0 ? "양수":(num < 0 ? "음수" : "0"));

}

}

1. 아래는 변수 num의 값 중에서 일의 자리를 1로 바꾸는 코드이다. 만일 변수 num의

값이 333이라면 331이 되고, 777이라면 771이 된다. ( )에 알맞은 코드를 넣으시오.

| class Exercise {  public static void main(String[] args) {  int num = 333;  System.out.println( );  }  } |
| --- |

정답: num/10\*10+1

1. 아래는 화씨(Fahrenheit)를 섭씨(Celcius)로 변환하는 코드이다. 변환공식이 'C =

5/9 ×(F - 32)'라고 할 때, ( )에 알맞은 코드를 넣으시오. 단, 변환 결과값은 소수점

셋째자리에서 반올림해야한다.(Math.round()를 사용하지 않고 처리할 것)

| class Exercise {  public static void main(String[] args) {  int fahrenheit = 100;  float celcius = ( )  System.out.println("Fahrenheit:"+fahrenheit);  System.out.println("Celcius:"+celcius);  }  } |
| --- |

정답: (int)((5/9f \* (fahrenheit - 32))\*100 + 0.5) / 100f;

\*먼저 화씨를 섭씨로 바꾸는 공식은 '5/9f \* (fahrenheit - 32)'이다. 5/9의 결과

는 0이기 때문에 두 피연산자 중 어느 한 쪽을 반드시 float나 double로 해야만 실수형태

의 결과를 얻을 수 있다. 그래서 정수형 리터럴인 9대신 float타입의 리터럴인 9f를 사용

하였다. 소수점 셋째자리에서 반올림을 하려면 다음의 과정을 거쳐야한다.

1. 값에 100을 곱한다.

37.77778 \* 100

2. 1의 결과에 0.5를 더한다.

3777.778 + 0.5 → 3778.278

3. 2의 결과를 int타입으로 변환한다.

(int)3778,278 → 3778

4. 3의 결과를 100f로 나눈다.(100으로 나누면 소수점 아래의 값을 잃는다.)

3778 / 100f → 37.78

1. 아래 코드의 문제점을 수정 후 실행 결과를 얻도록 하시오.

| class Exercise3\_8 {  public static void main(String[] args) {  byte a = 10;  byte b = 20;  byte c = a + b;  char ch = 'A';  ch = ch + 2;  float f = 3 / 2;  long l = 3000 \* 3000 \* 3000;  float f2 = 0.1f;  double d = 0.1;  boolean result = d==f2;  System.out.println("c="+c);  System.out.println("ch="+ch);  System.out.println("f="+f);  System.out.println("l="+l);  System.out.println("result="+result);  }  } |
| --- |

실행결과

c=30

ch=C

f=1.5

l=27000000000

result=true

정답:

| class Exercise3\_8 {  public static void main(String[] args) {  byte a = 10;  byte b = 20;  byte c = (byte)(a + b);  char ch = 'A';  ch = (char)(ch + 2);  float f = 3 / 2f;  long l = 3000 \* 3000 \* 3000L;  float f2 = 0.1f;  double d = 0.1;  boolean result = (float)d==f2;  System.out.println("c="+c);  System.out.println("ch="+ch);  System.out.println("f="+f);  System.out.println("l="+l);  System.out.println("result="+result);  }  } |
| --- |

1. 다음은 문자형 변수 ch가 영문자(대문자 또는 소문자)이거나 숫자일 때만 변수 b

의 값이 true가 되도록 하는 코드이다. ( )에 알맞은 코드를 넣으시오.

| class Exercise {  public static void main(String[] args) {  char ch = 'z';  boolean b = ( )  System.out.println(b);  }  } |
| --- |

정답: ('a' <= ch && ch <= 'z') || ('A' <= ch && ch <= 'Z') || ('0' <= ch && ch <= '9')

1. 다음은 대문자를 소문자로 변경하는 코드인데, 문자 ch에 저장된 문자가 대문자

인 경우에만 소문자로 변경한다. 문자코드는 소문자가 대문자보다 32만큼 더 크다. 예를

들어 'A'의 코드는 65이고 'a'의 코드는 97이다. ( )에 알맞은 코드를 넣으시오. [힌트: 삼항연산자 이용]

| class Exercise {  public static void main(String[] args) {  char ch = 'A';  char lowerCase = ( );  System.out.println("ch:"+ch);  System.out.println("ch to lowerCase:"+lowerCase);  }  } |
| --- |

실행결과:

ch:A

ch to lowerCase:a

정답: ('A' <= ch && ch <= 'Z') ? (char)(ch+32): ch;

\*대문자인 경우에만 문자코드의 값을 32만큼 증가시키면 소문자가 된다. 문자 ch가

대문자인지를 확인하는 조건식은 'A' <= ch && ch <= 'Z'이고, 문자 ch의 문자코드를

32증가시키기 위해서는 덧셈연산을 해야하는데, 이 때 덧셈연산의 결과가 int이므로

char타입으로의 형변환이 필요하다.