

Java 실습

지능 물류 빅데이터 연구소 이상현

02 데이터 활용

키보드 입력 (KeyboardInput)

- 키보드에서 데이터를 입력 받아서 화면에 출력하세요.

- 데이터를 입력 받을 때는 Scanner 객체를 사용합니다.

- 실행 예

- Scanner scanner = new Scanner(**System.in**);

- ➔ System.in : 표준 입력 장치 (키보드)

- **int** inum = scanner.**nextInt**();

- ➔ 정수 (4byte)

- **double** dnum = scanner.**nextDouble**();

- ➔ 배정도 실수 (8byte)

- **String** str = scanner.**next**();

- ➔ 문자열

- **String** strline = scanner.**nextLine**();

- ➔ 문자열 (with WhiteSpace : " " or tab)

자기소개 (IntroduceMyself)

- 변수를 활용하여 자기 소개를 출력하세요.
 - println
- 실행 예
 - 이름 : 홍길동
 - 생년월일 : 2001년 10월 1일
 - 학교 : 부산대학교
 - 학과 : 컴퓨터공학과
 - 입학 : 2020년

일주일간 수입 계산 (CalcIncome)

- 임의의 변수를 만들어서 일주일간의 수입을 입력하고 총 수입과 일일 평균수입을 출력하세요.
 - printf
- 입력 예
 - 월요일 : 10000
 - 화요일 : 20000
 - 수요일 : 10000
 - 목요일 : 20000
 - 금요일 : 10000
- 출력 예
 - 총 수입 : 70000원
 - 일 평균 : 14000원

환율 계산 (CalcExchange)

- 환율이 아래와 같이 주어졌을 때 임의의 달러를 원화로 계산해서 출력하세요.
 - 1 달러 : 1,350원
- 실행 예
 - 입력 : 10 달러 → 13,500원

두 수의 곱 & 몫과 나머지 (Calculator)

- 두 수를 입력 받아서 곱을 출력하세요.
- 앞의 수를 뒤의 수로 나누어서 몫과 나머지를 출력하세요.
- 실행 예
 - 입력 값 : 15, 4
 - 곱 : $15 * 4 = 60$
 - 몫 : $15 / 4 = 3$
 - 나머지 : $15 - (4 * 3) = 3$

데이터 단위 변환 (TransformUnit)

- 광속을 분속, 시속, 마하로 출력하세요.
- 임의의 속도를 시속으로 입력 받아서 초속, 분속, 마하로 출력하세요.
- 실행 예
 - 입력 : 300,000km/s
 - 분속 : km/m
 - 시속 : km/h
 - 마하 : mach

자유 낙하 물체의 위치 구하기 (FreeFall)

- 등가속 운동 물체의 위치 공식은 아래와 같습니다.

- $x(t) = \frac{1}{2}at^2 + v_0t + x_0$ ($x(t)$: 위치, a : 가속도, t : 이동 시간, v_0 : 초기 속도, x_0 : 초기 위치)

- 임의의 상공에서 정지 상태의 물체를 가만히 놓아 자유 낙하 시키려 한다. 높이를 입력하면 지면 도착 시간을 아래 실행 예와 같이 출력하세요. (저항은 없고, 중력 가속도는 9.81m/s로 한다.)
- 실행 예
 - 높이(m) : 1000
 - 지면 도착 시간 : 14.3초

최소 지폐 수 계산 (MinBillCount)

- 상품 가격 167,000원을 지불하기 위해 필요한 최소 지폐 장수는 아래와 같다.
 - 5만원 * 3장
 - 1만원 * 1장
 - 5천원 * 1장
 - 1천원 * 2장
- 천원 미만은 할인하고 임의의 금액을 지불하기 위해 필요한 지폐 장수를 구하는 프로그램을 작성하세요.

03 제어문

수치 합 (NumericalSum)

- 0 보다 큰 하나의 정수를 입력 받아서 전체 합, 홀수 합, 짝수 합을 출력하세요.
- 실행 예
 - 입력 : n
 - 전체 합 : $1+2+3+ \dots + (n-1)+n = \text{sum}$
 - 홀수 합 : $1+3+\dots = \text{sum}$
 - 짝수 합 : $2+4+\dots = \text{sum}$

도형 면적 계산 (Figure)

- 삼각형 또는 사각형을 선택 받는다.
- 만약 (if)
 - 삼각형이면 밑변과 높이를 입력 받아서 면적을 계산해서 출력한다.
 - 사각형이면 너비와 높이를 입력 받아서 면적을 계산해서 출력한다.
- 실행 예
 - 사각형 : 너비 - 10, 높이 - 10, 면적 : 100
 - 삼각형 : 밑변 - 10, 높이 - 10, 면적 : 50

소수 판단 (PrimeNumber1)

- 임의의 수 N을 입력 받아서 해당 수가 소수인지 확인하는 프로그램을 작성하세요.
- 방법
 - 2 ~ N-1 까지 확인
 - 2 ~ N/2 까지 확인
 - 2 ~ Math.sqrt(N) 까지 확인
- 실행 예
 - 입력 : 7 → 7 is a Prime.
 - 입력 : 9 → 9 is not a Prime.

```
public class PrimeNumber1 {  
  
    public static void main(String[] args) {  
        Scanner sc = new Scanner(System.in);  
        while(true) {  
            System.out.print("Number[0:exit]:");  
            int num = sc.nextInt();  
            if (num == 0) break;  
            if(isPrime(num) == true)  
                System.out.println(num + " is a Prime.");  
            else  
                System.out.println(num + " is not a Prime.");  
        }  
        sc.close();  
        System.out.println("Done!");  
    }  
    // num이 소수면 true, 그렇지 않으면 false를 리턴  
    public static boolean isPrime(int num) {  
  
        // 코드 작성  
  
    }  
}
```

소수 찾기 (PrimeNumber2)

- 정수의 자릿수를 입력 받아서 해당 자릿수에 속하는 소수를 찾아서 출력하는 프로그램을 작성하세요.

```
int num = 3;  
int s = (int)Math.pow(10.0, (double)(num-1));  
int e = (int)Math.pow(10.0, (double)(num))-1;
```

- 실행 예
 - 입력 자릿수가 3이라면 3자릿수 정수 (100 ~ 999) 중에서 소수에 해당하는 숫자를 찾아서 출력한다.

```
public class PrimeNumber2 {  
  
    public static void main(String[] args) {  
        Scanner sc = new Scanner(System.in);  
        while(true) {  
            System.out.print("Number[0:exit]:");  
            int num = sc.nextInt();  
            if (num == 0) break;  
  
            int s = (int)Math.pow(10.0, (double)(num-1));  
            int e = (int)Math.pow(10.0, (double)(num))-1;  
  
            // 코드 작성  
  
            System.out.println("Number of Prime :" + count);  
        }  
    }  
    System.out.println("Done!");  
}
```

소수 찾기 (PrimeNumber3)

- 정수의 자릿수를 입력 받아서 아래와 같은 형태의 소수를 찾아서 출력하는 프로그램을 작성하세요.
- 실행 예
 - 입력 자릿수 : 4 (1000 ~ 9999)
 - 7331 ➔ 7, 73, 733, 7331 모두 소수

7331



이진수 (BinaryNumber)

- 임의의 수를 입력 받아서 이진수로 변환해서 출력하세요.
- 실행 예
 - 입력 : 2 ➔ 10
 - 입력 : 13 ➔ 1101

```
public class BinaryNumber {  
  
    public static void main(String[] args) {  
        Scanner sc = new Scanner(System.in);  
        while(true) {  
            System.out.print("십진수[0:exit]:");  
            int num = scanner.nextInt();  
            if (num == 0) break;  
  
            System.out.printf("==>%s\n",  
                             decimalToBinary(num));  
        }  
        System.out.print("Done!");  
    }  
  
    private static String decimalToBinary(int num) {  
  
        // 코드 작성  
  
    }  
}
```

최대공약수 & 최소공배수 (GcdLcm)

- 두 개의 정수를 입력 받아서 최대공약수와 최소공배수를 구해서 출력하세요.

- 유클리드 호제법

1. 입력 받은 두 개의 정수 중 큰 정수를 max, 작은 정수를 min
2. $rem = max \% min$
3. rem이 0이면 ➔ 최대공약수 = 작은 정수, 최소공배수 = 두 정수의 곱 / 최대공약수
4. rem이 0이 아니면 ➔ $max = min, min = rem$, 2번 반복

- 실행 예

- 입력 : 12, 18

➔ 최대공약수 : 6, 최소공배수 : 36

예금 복리 계산 (DepositInterest)

- 은행에 목돈 a 를 예금하려 한다. 만기는 n 년, 이율은 연 복리로 $r\%$ 이다. 아래 공식을 참고하여 만기 시 수령 금액을 출력하세요. (소수점 이하 금액은 버릴 것)
 - $S=a(1+r)^n$ (S : 만기 금액, a : 원금, r : 이율, n : 만기 년수)
 - `int s = a * Math.pow(1 + r, n);`
- 실행 예
 - 원금(a) : 10,000,000원, 만기(n) : 5년, 연 복리(r) : 3%
 - 만기 금액 : 11,592,741원

시험성적 확인 (TestScore)

- 프로그래밍 수업의 학점은 아래와 같은 기준으로 결정됩니다. 주어진 성적 변수에 대한 학점을 반환하는 메소드 `grade()` 를 완성하여, 출력 예와 같은 결과를 얻으세요
 - A+: 95점 이상, A0: 90점 이상, B+: 85점 이상, B0: 80점 이상
 - C+: 75점 이상, C0: 70점 이상, D+: 65점 이상, D0: 60점 이상, F : 그 외
- 실행 예
 - 96점 -> A+
 - 85점 -> B+
 - 76점 -> C+

삼항 연산자 (TernaryOperator)

- 국영수 세 과목의 점수를 입력 받은 뒤 평균을 구해서 70점 이상일 때 통과가 된다. 임의의 점수를 입력 받아서 통과여부를 판별하는 메소드를 작성하고 결과를 아래 예와 같이 출력하세요. 이때, 한 과목이라도 60점 미만이면 과락이 되어 통과하지 못한다.
- 실행 예
 - 입력 : 95,65,80, 총계 : 240, 평균 : 80 ➔ 통과
 - 입력 : 95,65,55, 총계 : 215, 평균 : 71.67 ➔ 과락

switch or if-else (ControlStatements)

- 임의의 년도를 입력했을 때, 입력 연도의 12간지 동물 띠를 출력하는 메소드를 완성하고, 아래와 같이 출력되도록 하세요.
- switch문으로 구현하고, if-else 문으로도 구현하세요.
- 실행 예
 - 1986년 => 호랑이띠
 - 1990년 => 말띠
 - 2000년 => 용띠

구구단 (Gugudan) (1)

- 구구단을 출력하는 메소드를 구현하세요.

```
2단
2 * 1 = 2
2 * 2 = 4
2 * 3 = 6
2 * 4 = 8
2 * 5 = 10
2 * 6 = 12
2 * 7 = 14
2 * 8 = 16
2 * 9 = 18
-----
3단
3 * 1 = 3
3 * 2 = 6
3 * 3 = 9
:
```

```
public class Gugudan {
    // 단 하나를 출력하는 메소드
    public void print(int num) {
        // 코드 작성
    }
    // 구구단을 순서대로 출력하는 메소드
    public void printVertical() {
        // print 메소드를 사용하도록 코드 작성
    }
}
```

```
public class GugudanTest {
    public static void main (String[] args) {
        Gugudan ggd = new Gugudan();
        ggd.printVertical();
    }
}
```

구구단 (Gugudan) (2)

- 구구단을 출력하는 메소드를 구현하세요.

2 * 1 = 2	3 * 1 = 3	4 * 1 = 4 ...
2 * 2 = 4	:	:
2 * 3 = 6	:	:
2 * 4 = 8	:	:
2 * 5 = 10	:	:
2 * 6 = 12	:	:
2 * 7 = 14	:	:
2 * 8 = 16	:	:
2 * 9 = 18	3 * 9 = 27	4 * 9 = 36 ...

```
public class Gugudan {  
    :  
    public void printHorizontal() {  
        // 코드 작성  
    }  
}
```

```
public class GugudanTest {  
    public static void main (String[] args) {  
        Gugudan ggd = new Gugudan();  
        ggd.printHorizontal();  
    }  
}
```


구구단 (Gugudan) (3)

- 구구단을 출력하는 메소드를 구현하세요.

2단	3단	4단
5단	6단	7단
8단	9단	

col = 3

2단	3단	4단	5단
6단	7단	8단	9단

col = 4

```
public class Gugudan {  
    :  
    public void printColumn(int col) {  
        // 코드 작성  
    }  
}
```

```
public class GugudanTest {  
    public static void main (String[] args) {  
        Gugudan ggd = new Gugudan();  
        ggd.printColumn(3);  
    }  
}
```

윤년 검사 (LeafYear)

- 임의의 년도를 입력 받아서 윤년인지를 판단하는 메소드를 구현하세요.
 - 4로 나누어 떨어지는 해는 윤년, 그 밖의 해는 평년
 - 4로 나누어 떨어지더라도 100으로 나누어지고 떨어지면 평년
 - 4로 나누어 떨어지더라도 100으로 나누어지고 떨어지지만 400으로 나누어 떨어지면 윤년
- ➔ 4의 배수이면서 100의 배수가 아니거나 400의 배수일 때 윤년

두 직선의 교차점 (IntersectOfLines)

- 두 개의 직선 좌표가 주어졌을 때 두 직선의 교차점을 구해서 출력하세요.
- 실행 예
 - line1 : 1,1,3,3 ➔ x1, y1, x2, y2
 - line2 : 3,1,1,3 ➔ x3, y3, x4, y4
 - intersect : 2,2 ➔ result

```
public class IntersectLines {  
    private double x1,y1,x2,y2,x3,y3,x4,y4;  
    public static void printIntersect() {  
        // 교차점을 구해서 출력하는 코드 작성  
    }  
  
    public static void main (String[] args) {  
        // Scanner를 이용해서 좌표 입력  
        // 코드 작성  
        printIntersect();  
    }  
}
```

경우의 수 (NumberOfCases)

- 2g, 3g, 5g의 추가 각각 10개씩 있을 때, 10~100사이의 임의의 값을 입력 받고, 추의 합이 입력 받은 값이 되는 경우를 찾아서 출력하세요. 단, 각각의 추는 1개 이상은 사용되어야 합니다.
- 실행 예
 - 입력 값 : 31
 - 출력 값
 - (1,3,4)(1,8,1)(2,4,3)(3,5,2)(4,1,4)(4,6,1)(5,2,3)(6,3,2)(7,4,1)(9,1,2)(10,2,1) : 11

행렬 (MatrixCalc)

- M*N 행렬의 값을 랜덤으로 입력하고, 가로 세로의 합을 출력하세요

- 실행 예

- 입력

$$\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}$$

- 출력

- $1 + 2 + 3 = 6$

- $4 + 5 + 6 = 15$

- $7 + 8 + 9 = 24$

- $1 + 4 + 7 = 12$

- $2 + 5 + 8 = 15$

- $3 + 6 + 9 = 18$

04 배열

지역별 접속자 수 (ContactCounter)

- 이번 한주간 어떤 사이트의 접속자 수는 지역별로 아래와 같다. 초기값을 가진 배열로 생성하여 출력 하고, 랜덤으로 임의의 값을 생성해서 기존 배열 값에 추가한 뒤 다시 출력하세요.
 - 서울: 599 명, 부산: 51 명, 인천: 46 명, 대전: 43 명, 대구: 27 명
- 출력 예
 - 서울: 599 명
 - 부산: 51 명
 - 인천: 46 명
 - 대전: 43 명
 - 대구: 27 명

정렬 기초 (BubbleSort)

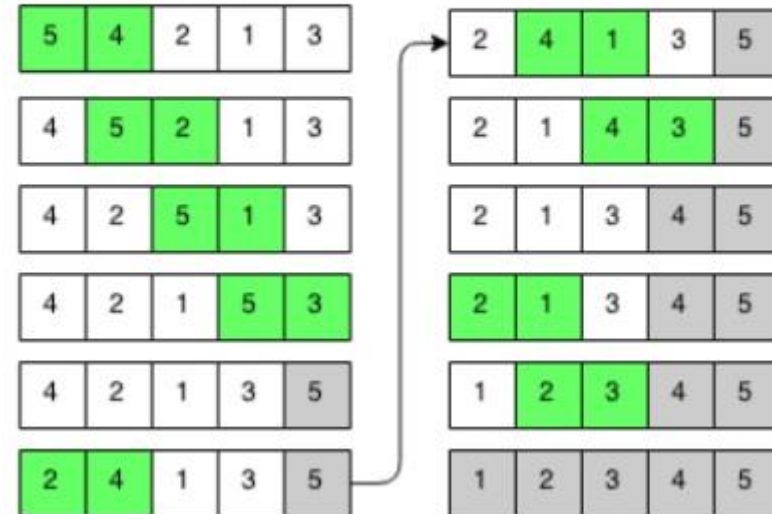
- 45 이하의 랜덤으로 생성한 정수 7개를 가지는 배열을 만들고 정렬하고자 한다.

- 실행 예

- 입력 : { 42, 26, 10, 6, 11, 36, 2 }
- 출력 : { 2, 6, 10, 11, 26, 36, 42 }

Bubble Sort

옆과 비교해서 왼쪽이 더 크면 좌우 교환하는 정렬



배열 합치기 (MergeArray)

- 길이가 $N+1$ 인 두 정수 배열 A와 B가 있을 때 배열 C를 만들고자 한다.
 - $\text{int}[] A = \{ a_0, a_1, \dots, a_N \};$
 - $\text{int}[] B = \{ b_0, b_1, \dots, b_N \};$
 - $\text{int}[] C = \{ a_0, b_0, a_1, b_1, \dots, a_N, b_N \};$
- 메소드 `merge()`를 완성하고, 각 배열 값을 출력하세요.

자모 계산 (CharCounter)

- 영어의 모음이 A, E, I, O, U라 가정했을 때, 문자열의 자음과 모음의 개수를 출력하는 메소드를 작성하세요.
 - [Hint] toCharArray : 문자열을 문자 배열로 String의 메소드
- 출력 예
 - Programming is fun! right? => 자음 15개, 모음 6개

팩토리얼 (Factorial)

- 아래 출력 예와 같은 결과를 얻도록 메소드를 작성하세요.
- 출력 예
 - 입력 : 4 ➔ $4! = 4 * 3 * 2 * 1 = 24$

피보나치 수열 (Fibonacci)

- 피보나치 수열은 아래와 같은 규칙성을 갖는 특별한 수열입니다. 피보나치 수열을 배열로 만드는 메소드를 구현하세요.
 - $F_0=0$
 - $F_1=1$
 - $F_n=F_{n-2}+F_{n-1}(n \geq 2)$
- 출력 예
 - 입력 : 11
 - 출력 : 0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55

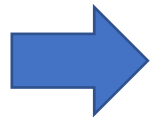
전치행렬 (MatrixTransposed)

- 3*3, 3*4, 4*3 행렬 3개를 각각 선언하고 각 행렬의 전치행렬을 구해서 출력하세요.

- 전치행렬 : 행과 열을 바꾼 행렬

- 실행 예

1	2	3
4	5	6
7	8	9



1	4	7
2	5	8
3	6	9

행렬 곱 (MatrixMultiplication)

- 3*4 배열과 4*3 배열을 선언하고 0~10사이의 값을 랜덤으로 입력
- 두 행렬의 곱을 계산해서 출력하세요.

$$AB = \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix} \begin{pmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{pmatrix} = \begin{pmatrix} a_{11}b_{11}+a_{12}b_{21} & a_{11}b_{12}+a_{12}b_{22} \\ a_{21}b_{11}+a_{22}b_{21} & a_{21}b_{12}+a_{22}b_{22} \end{pmatrix}$$

$$AB = \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix} \begin{pmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{pmatrix} = \begin{pmatrix} a_{11}b_{11}+a_{12}b_{21} & a_{11}b_{12}+a_{12}b_{22} \\ a_{21}b_{11}+a_{22}b_{21} & a_{21}b_{12}+a_{22}b_{22} \end{pmatrix}$$

$$AB = \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix} \begin{pmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{pmatrix} = \begin{pmatrix} a_{11}b_{11}+a_{12}b_{21} & a_{11}b_{12}+a_{12}b_{22} \\ a_{21}b_{11}+a_{22}b_{21} & a_{21}b_{12}+a_{22}b_{22} \end{pmatrix}$$

$$AB = \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix} \begin{pmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{pmatrix} = \begin{pmatrix} a_{11}b_{11}+a_{12}b_{21} & a_{11}b_{12}+a_{12}b_{22} \\ a_{21}b_{11}+a_{22}b_{21} & a_{21}b_{12}+a_{22}b_{22} \end{pmatrix}$$

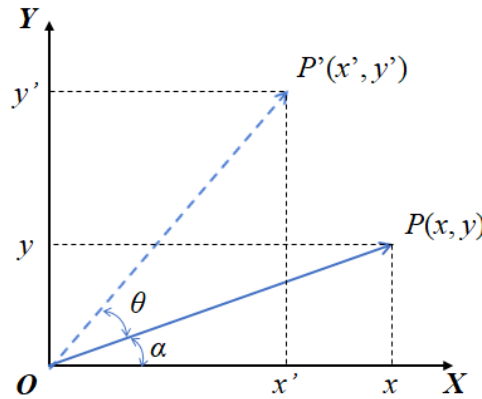
```
public class CalcMatrix {  
    int[][] mA = new int m[3][4];  
    int[][] mB = new int m[4][3];  
    public void multiply () {  
        // 코딩  
    }  
    public void print(int[][] m) {  
        // 코딩  
    }  
}
```

2차원 회전 변환 행렬 (MatrixTransform)

- 임의로 입력한 한 점을 원점을 기준으로 임의의 각도로 반시계방향으로 회전했을 때의 좌표를 출력하세요.

- 실행 예

- 입력 좌표 : 10, 20
- 입력 각도 : 90
- 출력 좌표 : -20, 10



$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

➔ `Math.toRadians()`, `Math.sin()`, `Math.cos()`

윤년 검사 (LeapYear)








- 금년 이후 윤년 10개를 찾아서 배열에 저장하고 이를 출력하는 메소드를 구현해 보세요.
 - 4로 나누어 떨어지는 해는 윤년, 그 밖의 해는 평년
 - 4로 나누어 떨어지더라도 100으로 나누어지고 떨어지면 평년
 - 4로 나누어 떨어지더라도 100으로 나누어지고 떨어지지만 400으로 나누어 떨어지면 윤년
- ➔ 4의 배수이면서 100의 배수가 아니거나 400의 배수일 때 윤년

장기 말이 이동할 수 있는 경로

- 장기의 말(馬)은 가로로 2칸 이동하면 세로로 1칸, 세로로 2칸 이동하면 가로로 1칸 이동할 수 있다. 8 * 8 칸의 장기 판이 있을 때 임의의 지점을 입력했을 때 말이 이동할 수 있는 지점과 모든 경우의 수를 출력하세요.

• 예)

- 말의 위치 : b3
- 정답 : a1,a5,c1,c5,d2,d4 → 6

	a	b	c	d	e	f	g	h
1								
2								
3								
4								
5								
6								
7								
8								

05, 06 객체지향

사각기둥 클래스 (SquarePillar)

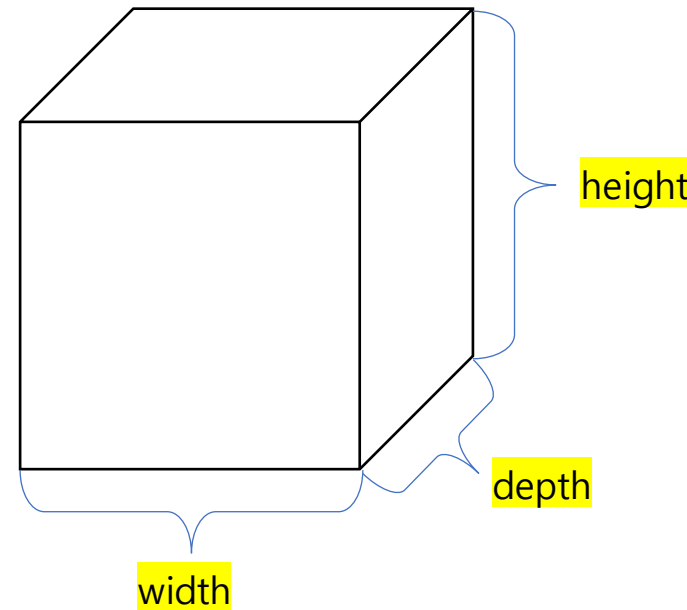
- 아래의 필드와 메소드를 갖는 사각기둥 클래스를 작성하세요.

- 필드:

- width : 너비
- depth : 깊이
- height : 높이

- 메소드:

- getVolume() : 부피 반환
- getArea() : 겉넓이 반환



- 출력 예

- 사각기둥의 부피: xx.y $\text{width} * \text{depth} * \text{height}$
- 사각기둥의 겉넓이: xx.y $(\text{width} * \text{depth}) * 2 + (\text{width} * \text{height}) * 2 + (\text{depth} * \text{height}) * 2$

원기둥 클래스 (Cylinder)

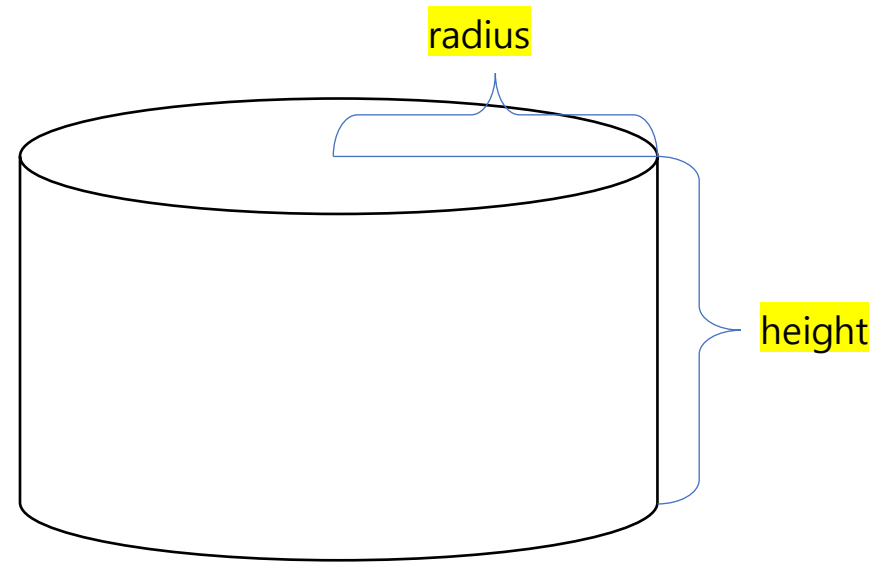
- 아래의 필드와 메소드를 갖는 원기둥 클래스를 작성하세요.

- 필드:

- radius : 반지름
- height : 높이

- 메소드:

- getVolume() : 부피 반환
- getArea() : 겉넓이 반환



- 출력 예

- 원기둥의 부피: 251.33
- 원기둥의 겉넓이: 226.19

Math.**PI**

$\text{PI} * \text{radius} * \text{radius} * \text{height}$

$\text{PI} * \text{radius} * \text{radius} * 2 + (2 * \text{PI} * \text{radius}) * \text{height}$

Food 클래스 (Food)

- 아래의 필드와 메소드를 갖는 Food 클래스를 작성하고, 10개의 객체를 생성해서 배열에 담은 다음, 저장된 객체들을 반복문을 이용해서 출력하세요.
 - 필드:
 - name : 이름
 - price : 가격
 - 메소드:
 - toString () : 문자열 출력
 - **Food { name: 치킨, price: 18000원 }**

Member 클래스 (Member)

- 아래의 필드와 메소드를 갖는 Member 클래스를 작성하고, 10개의 객체를 생성해서 배열에 저장한 다음, 저장된 객체들을 반복문을 이용해서 출력하세요.
 - 필드:
 - username : 이름
 - password : 암호
 - role : 권한(사용자 or 관리자)
 - enabled : 사용가능 (True or False)
 - 메소드:
 - toString () : 문자열 출력
 - **Member { username: 홍길동, password: abcd, role: 사용자, enabled: True }**

통합

Collection

- 100개의 정수를 랜덤하게 생성
 - ArrayList 에 입력해서 출력하세요.
 - LinkedList 에 입력해서 출력하세요.
 - Vector 에 입력해서 출력하세요.
- HashSet 에 입력해서 출력하세요.
- TreeSet 에 입력해서 출력하세요
- HashMap 에 입력해서 출력하세요.

Map – HashMap

- 아래 데이터를 HashMap을 사용하여 저장하고, 출력 예와 같은 결과를 얻으세요.

- 이름 이메일
- 홍길동 hongkd@korea.kr
- 이순신 leess@korea.kr
- 강감찬 kangkc@korea.kr

- 출력 예

- emails.size() -> 3
- == key set ==
 - 홍길동
 - 강감찬
 - 이순신
- == values ==
 - hongkd@korea.kr
 - kangkc@korea.kr
 - leess@korea.kr

큰 정수 더하기 연산 (CalcBigNumber)

- 20자리 이상의 큰 정수는 long형 데이터를 넘어서기 때문에 바로 연산을 할 수 없다. 한 가지 방법으로 모든 자리수의 데이터를 byte형 배열로 분리해서 더하는 방법을 사용할 수 있다. 아래 예와 같은 방법으로 임의의 20자리 두 수가 저장된 텍스트 파일(input.txt)을 읽어서 결과를 텍스트 파일(output.txt)에 출력하는 메서드를 작성하세요.
- 연산 예
 - $12345 + 6789 \rightarrow \{ 5, 4, 3, 2, 1 \} + \{ 9, 8, 7, 6 \}$
 - $\rightarrow \{ 14, 12, 10, 8, 1 \} \rightarrow 4, 3, 1, 9, 1 \rightarrow 19134$

정렬 (SelectionSort)

- sortinput.txt에는 10개의 중복되지 않는 정수가 저장되어 있습니다. 이 수들을 모두 읽어서 배열에 저장한 뒤 정렬하고 결과를 출력하는 클래스를 작성하세요.
- 정렬 방법 (선택 정렬)
 1. 전체 배열에서 제일 큰 값을 검색
 2. 찾은 제일 큰 값과 배열의 마지막 값을 swap
 3. 제일 마지막을 제외한 배열에서 제일 큰 값을 검색
 4. 찾은 제일 큰 값과 배열의 마지막 앞의 값을 swap

:

행렬 곱 (MatrixMultiplication) 2

- 3*4 배열과 4*3 배열을 파일로부터 입력 (Text File)
- 두 행렬의 곱을 계산해서 출력하세요.

$$AB = \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix} \begin{pmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{pmatrix} = \begin{pmatrix} a_{11}b_{11} + a_{12}b_{21} & a_{11}b_{12} + a_{12}b_{22} \\ a_{21}b_{11} + a_{22}b_{21} & a_{21}b_{12} + a_{22}b_{22} \end{pmatrix}$$

$$AB = \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix} \begin{pmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{pmatrix} = \begin{pmatrix} a_{11}b_{11} + a_{12}b_{21} & a_{11}b_{12} + a_{12}b_{22} \\ a_{21}b_{11} + a_{22}b_{21} & a_{21}b_{12} + a_{22}b_{22} \end{pmatrix}$$

$$AB = \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix} \begin{pmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{pmatrix} = \begin{pmatrix} a_{11}b_{11} + a_{12}b_{21} & a_{11}b_{12} + a_{12}b_{22} \\ a_{21}b_{11} + a_{22}b_{21} & a_{21}b_{12} + a_{22}b_{22} \end{pmatrix}$$

$$AB = \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix} \begin{pmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{pmatrix} = \begin{pmatrix} a_{11}b_{11} + a_{12}b_{21} & a_{11}b_{12} + a_{12}b_{22} \\ a_{21}b_{11} + a_{22}b_{21} & a_{21}b_{12} + a_{22}b_{22} \end{pmatrix}$$

```
public class CalcMatrix {  
  
    :  
  
    public void loadMatrix(String fname)  
    {  
        // 코딩  
    }  
}
```

행렬 (MatixLinear)

- 24개의 원소를 가지는 정수형 1차원 배열을 선언해서 랜덤으로 값을 입력

1. 입력된 배열 출력 & 오름차순으로 정렬한 뒤 출력

```
int[] array = new int[24];
```

2. 배열을 3by4 A₁행렬과 3by4 B₁행렬로 만들어서 행렬 합을 구한 뒤 출력

3. 배열을 3by4 A₂행렬과 4by3 B₂행렬로 만들어서 행렬 곱을 구한 뒤 출력

4. 배열을 4by3 A₃행렬과 3by4 B₃행렬로 만들어서 행렬 곱을 구한 뒤 출력

s : start index
r : row (행)
c : column (열)

// 행렬합

```
Matrix mA = new Matrix(array, 0, 3, 4);  
Matrix mB = new Matrix(array, 12, 3, 4);  
mA.print();  
mB.print();  
Matrix mC = Matrix.sum(mA, mB);  
mC.print();
```

s, r, c

// 행렬곱

```
Matrix mA = new Matrix(array, 0, 3, 4);  
Matrix mB = new Matrix(array, 12, 4, 3);  
mA.print();  
mB.print();  
Matrix mC = Matrix.multiple(mA, mB);  
mC.print();
```

s, r, c

GeometryTest

- 오른쪽 코드를 완성하세요.
- 상속 (인터페이스)
- 다형성
- 제네릭
- 람다식
- 함수형 인터페이스 API

```
interface Geometry {  
    double getArea();  
}  
  
class Circle implements Geometry {  
    int radius;  
}  
  
class Triangle implements Geometry {  
    int width;  
    int height;  
}  
  
class Rectangle implements Geometry {  
    int width;  
    int height;  
}  
  
public class GeometryTest {  
    public static void main(String[] args) {  
        Function<Geometry, Double> func = (g) -> {  
            // coding  
        };  
        // 반지름이 5인 원의 면적 출력  
        // 너비가 10, 높이가 5인 삼각형의 면적 출력  
        // 너비가 10, 높이가 5인 사각형의 면적 출력  
    }  
}
```