プログラミング・ロジック　実習課題

ORACLE

1 章実習課題

【ステップ A-1】　Java テクノロジー

Java 言語の特徴を 4 つ記述してください。

【ステップ A-2】　Java プログラムの構成

プログラム HelloJava.java をコンパイルし、以下のような実行結果になるようにしてください。

コンパイル例：

```
> javac HelloJava.java
```

実行例：

```
> java HelloJava
Hello Java!
```

完成させるプログラム HelloJava.java

HelloJava.java

```
1.  // HelloJava.java
2.
3.  class HelloJava{
4.    public static void main(String[] args){
5.      System.out.println("Hello Java !");
6.    }
7.  }
```

【ステップ A-3】　Java プログラムの構成

プログラム HelloWorld.java をコンパイルするとコンパイルエラーが発生します。以下のような実行結果になるよう適切に修正、実行してください。

コンパイル例

```
> javac HelloWorld.java
```

実行例：

```
> java HelloWorld
Hello World
```

修正するプログラム HelloWorld.java

HelloWorld.java

```
1.  // HelloWorld.java
2.
3.  Class HelloWorld{
4.    public static void Main(string[] args){
5.      system.out.println("Hello World");
6.    }
7.  }
```

2 章実習課題

【ステップ A-1】　リテラルと変数

以下の実行例のように出力するプログラムを作成してください。
ソースファイル名は、 Variable.java とします。

条件：

- int 型の変数 num を宣言する
- 宣言した num に 10 を代入する
- 変数 num の値を表示する

実行例：

```
> java Variable
num : 10
```

【ステップ A-2】　配列

以下の条件にしたがって、配列の宣言、代入および値を表示するプログラムを作成してください。
ソースファイル名は、ArraySqrt.java とします。

条件：

- double 型の値を 3 つ格納するための配列を宣言する
- 宣言した配列に、以下の値を代入する

  0 番目： 1.414

  1 番目： 1.732

  2 番目： 2.236
- 配列に代入した各要素をそれぞれ実行例のように出力する

ヒント：

配列の宣言、代入、値の表示を行うコードの例は以下のとおりです。

```
double[] array = new double[3];
array[0] = 1.414;
System.out.println("The Square of 2 : " + array[0] );
```

実行例：

```
> java ArraySqrt
The Square of  2 : 1.414
The Square of  3 : 1.732
The Square of  5 : 2.236
```

【ステップ A-3】　文字列

以下の実行例のように出力するプログラムを作成してください。
ソースファイル名は、DispString.java とします。

条件：

- String 型の変数 myString を宣言し、文字列 "Hello Java World" を代入する
- 変数 myString の値を表示する

実行例：

```
> java DispString
myString : Hello Java World
```

【ステップ A-4】　コマンドライン引数

以下の実行例のように出力するプログラムを作成してください。
ソースファイル名は、CommandLineData.java とします。

条件：

- コマンドラインで文字列を 1 つ指定する
- コマンドライン引数として渡された値を出力する

実行例：

```
> java CommandLineData abc
args[0] : abc
```

【ステップ B-1】　リテラルと変数

以下の表をもとに網かけ部分に変数を宣言してプログラムを完成させてください。
変数を宣言したら、プログラムをコンパイルおよび実行し、結果を確認してください。

ソースファイル名は、Expression.java とします。

宣言する変数と代入する値：

変数	値
32 ビットの整数型の変数 abc	-100
64 ビットの整数型の変数 smallNum	-200L
32 ビットの浮動小数点型の変数 pi	3.1415f
64 ビットの浮動小数点型の変数 largeNum	6.022e23
真偽値型の変数 display	false
文字型の変数 alphabet	B
文字列型の変数 str	Welcome to Sun Microsystems
32 ビットの整数型の変数を 3 つ格納する配列 i	1 番目には 100、2 番目には 200、3 番目には 300

実行例：

```
> java Expression
abc        : -100
smallNum   : -200
pi         : 3.1415
largeNum   : 6.022E23
display    : false
alphabet   : B
str        : Welcome to Sun Microsystems
i[0]       : 100
i[1]       : 200
i[2]       : 300
```

完成させるプログラム Expression.java

Expression.java

```
 1.  //  Expression.java
 2.
 3.  class Expression {
 4.    public static void main(String[] args) {
 5.
 6.      // 変数の宣言と初期値の代入
 7.
 8.
 9.
10.
11.
12.
13.
14.
15.
16.
17.      // 変数の値の表示
18.      System.out.println("abc      : " + abc);
19.      System.out.println("smallNum : " + smallNum);
20.      System.out.println("pi       : " + pi);
21.      System.out.println("largeNum : " + largeNum);
22.      System.out.println("display  : " + display);
23.      System.out.println("alphabet : " + alphabet);
24.      System.out.println("str      : " + str);
25.      System.out.println("i[0]     : " + i[0]);
26.      System.out.println("i[1]     : " + i[1]);
27.      System.out.println("i[2]     : " + i[2]);
28.    }
29.  }
30.
```

【ステップ B-2】　コマンドライン引数

以下の実行例のように出力するプログラムを作成してください。
ソースファイル名は、ConnectedString.java とします。

条件：

- コマンドラインから 2 つの値を渡す
- 渡されたデータを - （ハイフン）で連結した形式で表示する

実行例：

```
> java ConnectedString 0120 123456
Connected String : 0120-123456

> java ConnectedString 150 0001
Connected String : 150-0001
```

【ステップ C-1】　リテラルと変数

以下の表をもとに変数を宣言し、実行例のように出力するプログラムを作成してください。
また、作成したプログラムをコンパイルおよび実行し、結果を確認してください。

ソースファイル名は、 DataType.java とします。

宣言する変数と代入する値：

変数	値
32 ビットの整数型の変数 number	104
64 ビットの浮動小数点型の変数 decimal	4.2
真偽値型の変数 truth	false
文字型の変数 letter	V
文字列型の変数 message	Good Morning

実行例：

```
> java DataType
number  : 104
decimal : 4.2
truth   : false
letter  : V
message : Good Morning
```

【ステップ C-2】　配列

以下の実行例のように出力するプログラムを作成してください。
ソースファイル名は、PriceData.java とします。

条件：

- ❑ 配列を使用する

実行例：

```
> java PriceData
Price No1 : 100 yen
Price No2 : 200 yen
Price No3 : 300 yen
Price No4 : 400 yen
Price No5 : 500 yen
```

【ステップ C-3】　文字列

以下の実行例のように出力するプログラムを作成してください。
ソースファイル名は、DispUser.java とします。

条件：

- 名前と年齢を扱う変数をそれぞれ適切なデータ型で宣言する
- 各変数で扱うデータ（名前、年齢）は、プログラムの中で任意の値を用意し、各変数に代入する

実行例：

```
> java DispUser
Your Name : Duke
Your Age  : 20
```

【ステップ C-4】　文字列コマンドライン引数

以下の実行例のように出力するプログラムを作成してください。
ソースファイル名は、UserData.java とします。

条件：

- ユーザ ID、名前、住所、電話番号の順に入力すると、その値を表示する

実行例：

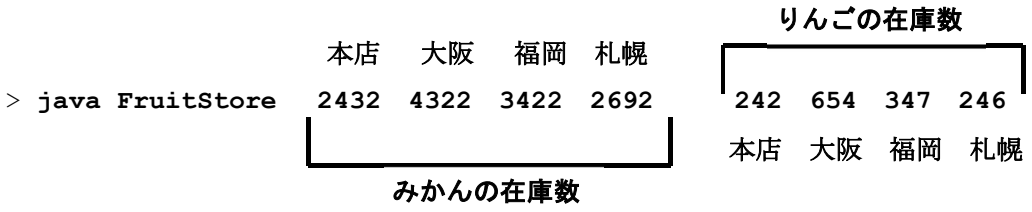
```
> java UserData 0001 Scott Tokyo 03-4150-1300
UserId      : 0001
UserName    : Scott
UserAddress : Tokyo
UserPhone   : 03-4150-1300
```

【ステップ C-5】　コマンドライン引数

以下の実行例のように出力するプログラムを作成してください。
ソースプログラム名は、FruitStore.java とします。

条件：

- 太陽青果は東京に本店があり、大阪、福岡、札幌に支店がある
- コマンドラインからすべての店のみかんとりんごの在庫数を入力すると、それぞれの店の在庫数とすべての店の合計を表示する
- コマンドライン引数はまず、みかんの在庫数を東京、大阪、福岡、札幌の順に入力し、その後続けてりんごの在庫数を東京、大阪、福岡、札幌の順に入力するものとする



実行例：

```
> java FruitStore 2432 4322 3422 2692 242 654 347 246
みかん
東京 ： 2432
大阪 ： 4322
福岡 ： 3422
札幌 ： 2692
合計 ： 12868

りんご
東京 ： 242
大阪 ： 654
福岡 ： 347
札幌 ： 246
合計 ： 1489
```

【ステップ C-6】　コマンドライン引数

以下の実行例のように出力するプログラムを作成してください。
ソースファイル名は、DataInput.java とします。

条件：

- コマンドラインから以下の値を与え、その型と値を表示する

  〈コマンドラインから与える定数〉

  ```
  24  2.33  true
  ```

- コマンドラインから与えた値は、プログラムの中で文字列型からそれぞれのデータ型に変換する

ヒント：

変換するためのメソッド	説明
Integer.parseInt(文字列)	文字列を int 型の数値に変換する。 文字列は数値の形式であること。
Double.parseDouble(文字列)	文字列を double 型の数値に変換する。 文字列は数値の形式であること。
Boolean.valueOf(文字列).booleanValue()	文字列を boolean 型の真偽値に変換する。 文字列は真偽値の形式であること。

実行例：

```
> java DataInput 24 2.33 true
int     : 24
double  : 2.33
boolean : true
```

3 章実習課題

【ステップ A-1】　算術演算子、代入演算子

以下の実行例のように出力するプログラムを作成してください。
ソースファイル名は、Divide.java とします。

条件：

- コマンドライン引数として 2 つの数値を入力し、その値の除算の結果と剰余算の結果を表示する
- 以下の例を参考に、コマンドライン引数を数値に変換する

```
int num1 = Integer.parseInt(args[0]);
```

実行例：

```
> java Divide 7 3
7 / 3 = 2
remainder : 1

> java Divide 8 2
8 / 2 = 4
remainder : 0
```

【ステップ A-2】　インクリメント演算子

以下のような宣言と式があります。それぞれの式が評価された後、変数の値はいくつになりますか。

```
int x, y;
x = 5;

y = x++ :    // ①
y = ++x :    // ②
```

①の処理実行後：　y は＿＿＿＿＿＿、x は＿＿＿＿＿＿。

②の処理実行後：　y は＿＿＿＿＿＿、x は＿＿＿＿＿＿。

【ステップ A-3】　インクリメント演算子

ステップ A-2 のコードをもとに実際にプログラムを作成し、結果を確認してください。

ソースファイル名は、IncreCheck.java とします。

【ステップ A-4】　関係演算子

以下のような宣言があります。各式の演算結果はどうなりますか。

```
int m_int = 10000, t_int = 20000;
double m_double = 10000.0;
float m_float = 10000.5f;
char m_char = 'a', t_char = 'A';

m_int >= t_int              // ①

m_int == m_double           // ②

m_int == m_float            // ③

m_char == t_char            // ④
```

①の結果：m_int >= t_int は_____。
②の結果：m_int == m_double は_____。
③の結果：m_int == m_float は_____。
④の結果：m_char == t_char は_____。

【ステップ A-5】　関係演算子

ステップ A-4 のコードをもとに実際にプログラムを作成し、結果を確認してください。
ソースファイル名は、RelationCheck.java とします。

【ステップ A-6】　基本データ型の型変換

以下のような宣言があります。各式の演算結果はどうなりますか。

```
int n = 10, result_one, result_two;
float x;

x = (float)n / 3;                    // ①
result_one = (int)(1.6 + 1.8);       // ②
result_two = (int)1.6 + (int)1.8;    // ③
```

①の結果：x は＿＿＿＿＿＿。

②の結果：result_one は ＿＿＿＿＿＿。

③の結果：result_two は＿＿＿＿＿＿。

【ステップ A-7】　基本データ型の型変換

ステップ A-6 のコードをもとに実際にプログラムを作成し、結果を確認してください。

ソースファイル名は、CastCheck.java とします。

【ステップ B-1】　算術演算子

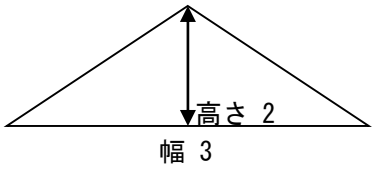
以下の実行例のように出力するプログラムを作成してください。
ソースファイル名は、Area.java とします。

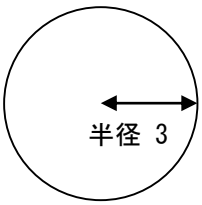
条件：

- コマンドラインから幅と高さを入力し、三角形、四角形、円の面積を求める
- 円の面積を求める場合は、入力した幅を半径、円周率を 3.14 として計算する

実行例：

```
> java Area 3 2
Input Width    : 3.0
Input Height   : 2.0
Triangle area  : 3.0
Square area    : 6.0
Circle area    : 28.26
```





【ステップ B-2】　基本データ型の型変換

以下のプログラムは、コンパイルするとコンパイルエラーが発生します。適切に修正し、実行できるようにしてください。
ソースファイル名は、Casts.java とします。

実行例：

```
> java Casts
d : 5.0
i : 7
b : 7
```

修正するプログラム Casts.java

Casts.java

```
 1.  // Casts.java
 2.
 3.  class Casts{
 4.    public static void main(String args[]) {
 5.      byte b;
 6.      int i = 5;
 7.      double d;
 8.
 9.      d = i;
10.
11.      System.out.println("d : " + d);
12.
13.      d = 7.85;
14.      i = d;
15.
16.      System.out.println("i : " + i);
17.
18.      b = i;
19.
20.      System.out.println("b : " + b);
21.    }
22.  }
```

【ステップ B-3】　算術演算子

以下の実行例のように出力するプログラムを作成してください。
ソースファイル名は Weight.java とします。

条件：

- コマンドラインから入力された身長に対する標準体重を求める
- 身長はセンチメートル単位で入力する
- 標準体重を求める式は、以下を参考にする

＜式＞

標準体重 = 22 × 身長(m) × 身長(m)

実行例：

```
> java Weight 165
あなたの標準体重 : 59.89499999999999 kg
```

【ステップ C-1】　算術演算子、代入演算子

以下の実行例のように出力するプログラムを作成してください。
ソースファイル名は、TimeRequired.java とします。

条件：

- コマンドラインから出発時間、到着時間の順に入力し、それぞれの値を表示する
- 出発時間と到着時間から所要時間を求めて表示する
- 日付はまたがない前提で作成する

実行例：

```
> java TimeRequired 09 30 17 00
Time of Departure  : 09:30
Time of Arrival    : 17:00
Time required      : 7h 30min

> java TimeRequired 16 00 21 45
Time of Departure  : 16:00
Time of Arrival    : 21:45
Time required      : 5h 45min
```

【ステップ C-2】　論理演算子

以下の実行例のように出力するプログラムを作成してください。
ソースファイル名は、AuthUser.java とします。

条件：

- コマンドラインから ID ナンバー(4 桁の数値)、部門ナンバー (1 桁の数値) の順に数値を 2 つ入力し、その値を表示する
- 以下の認証方法にしたがって認証を行い、その結果が正しければ true を、正しくなければ false を表示する
- 認証方法：認証は ID ナンバーが 7777 で、部門ナンバーが 3 のときにのみ true とする
- ID ナンバー、部門ナンバーはどちらも数値に変換して認証する

実行例：

```
> java AuthUser 7777 3
ID Number      : 7777
Section Number : 3
Authentication : true

> java AuthUser 1111 3
ID Number      : 1111
Section Number : 3
Authentication : false

> java AuthUser 7777 2
ID Number      : 7777
Section Number : 2
Authentication : false
```

【ステップ C-3】　基本データ型の型変換

以下の実行例のように出力するプログラムを作成してください。
ソースファイル名は、ImposeTax.java とします。

条件：

- コマンドラインから金額を 1 つ入力し、その値を表示する
- 入力した金額の消費税を求め、税込み金額を表示する
  なお、消費税率は 8% とし、小数点以下は切り捨てるものとする

ヒント：

- 切り捨てを行うには、求めた税込み金額を int 型へキャストする

実行例：

```
> java ImposeTax 295
Price : 295
TaxIn : 318

> java ImposeTax 2500
Price : 2500
TaxIn : 2700
```

【ステップ C-4】　基本データ型の型変換

以下の実行例のように出力するプログラムを作成してください。
ソースファイル名は Change.java とします。

条件：

- 一万円札で商品を購入したときのおつりの金額とお金の種類を表示する
- 商品の値段はコマンドラインから入力する
- おつりは金額の多いものから数えることにする

実行結果

```
> java Change 675
675 円の商品の支払いを一万円札で行うと
おつりは、9325 円になります

五千円札　: 1枚
千円札　　: 4枚
五百円玉　: 0枚
百円玉　　: 3枚
五十円玉　: 0枚
十円玉　　: 2枚
五円玉　　: 1枚
一円玉　　: 0枚
```

4 章実習課題

【ステップ A-1】　条件分岐

以下の実行例のように出力するプログラムを作成してください。
ソースファイル名は、EqualOrNot.java とします。

条件：

- コマンドライン引数として 2 つの数値を入力する
- 入力した値を比較し、同じ値であれば “The numbers you input are same values”、異なる値であれば、”The numbers you input are NOT same values”というメッセージを表示する

実行例：

```
> java EqualOrNot 3 3
The numbers you input are same values


> java EqualOrNot 3 2
The numbers you input are NOT same values
```

【ステップ A-2】　条件分岐

switch 文の中の各 case の処理から抜けるために使用する文は何ですか。

_____

【ステップ A-3】　繰り返し

while 文と do-while 文のもっとも大きな違いは何ですか。

_____

【ステップ A-4】　繰り返し

以下の for 文を利用したコードで、5 回 "Java" と表示するには、①、②、③の欄にどのようなコードを入れればよいですか。下欄の A から E のうち最も適切なものを選択してください。

```
for (int i = ① ; i < ② ; ③ ) {
    System.out.println("Java");
}
```

A.　①:1　②:5　③:i--
B.　①:0　②:6　③:i++
C.　①:1　②:5　③:i++
D.　①:5　②:0　③:i--
E.　①:0　②:5　③:i++

【ステップ A-5】　繰り返し

ステップ A-4 のコードをもとに実際にプログラムを作成し、結果を確認してください。
ソースファイル名は、ForCheck.java とします。

【ステップ B-1】　条件分岐

以下の実行例のように出力するプログラムを作成してください。
ソースファイル名は、ScopeOfNum.java とします。

条件：

- コマンドラインから整数値を入力する
- 入力された数値が 0 かそれより大きく、10 より小さいかを判断し、メッセージを表示する

ヒント：

- 論理演算子を使って条件を指定する

実行例：

```
> java ScopeOfNum 6
Input Number : 6
6 is OK

> java ScopeOfNum 18
Input Number : 18
18 is NG
```

【ステップ B-2】　条件分岐

以下の実行例のように出力するプログラムを作成してください。
ソースファイル名は、Letter.java とします。

条件：

- コマンドラインから 1 文字入力する
- 入力された文字が A ならば "Excellent"、B ならば "Very Good"、C ならば "Good"、D ならば "Fair"、E ならば "Poor" を表示する
- 上記以外の文字が入力された場合はエラーメッセージを表示しプログラムを終了させる

ヒント：

- switch 文を使用し、式を評価する

実行例：

```
> java Letter A
Input a letter[A-E] : A
Excellent

> java Letter Z
Input a letter[A-E] : Z
letter error: Unknown character Z
```

【ステップ B-3】　繰り返し

以下の実行例のように出力するプログラムを作成してください。
ソースファイル名は、Star.java とします。

条件：

- コマンドラインから入力された数字の数だけアスタリスク (*) を表示する

ヒント：

- for 文を使い、* の表示を繰り返す

実行例：

```
> java Star 10
Input Number : 10
**********
```

【ステップ C-1】　条件分岐

以下の実行例のように出力するプログラムを作成してください。
ソースファイル名は、EvenOrOdd.java とします。

条件：

- コマンドライン引数として数値を入力する
- 入力した値が奇数か偶数かを判断し、奇数の場合は “It's an odd number.” を、偶数の場合は “It's an even number.” を表示する

実行例：

```
> java EvenOrOdd 5
Input number : 5
It's an odd number.

> java EvenOrOdd 256
Input number : 256
It's an even number.

> java EvenOrOdd -128
Input number : -128
It's an even number.
```

【ステップ C-2】　条件分岐

以下の実行例のように出力するプログラムを作成してください。
ソースファイル名は、BirthdaySeason.java とします。

条件：

- コマンドラインから誕生月を入力する
- その入力された値から生まれた季節を判断し表示する

  ＜誕生月と季節の関係＞

  3月、 4月、 5月　　は　Spring.
  6月、 7月、 8月　　は　Summer.
  9月、10月、11月　　は　Autumn.
  12月、 1月、 2月　　は　Winter.

- 1 から 12 以外の数値が入力された場合は、入力された数値と、エラーメッセージとして "Please input again." を表示してプログラムを終了させる

実行例：

```
> java BirthdaySeason 7
Input number          : 7
Your birthday season  : Summer.

> java BirthdaySeason 9
Input number          : 9
Your birthday season  : Autumn.

> java BirthdaySeason 24
Input number          : 24
Please input again.

> java BirthdaySeason -3
Input number          : -3
Please input again.
```

【ステップ C-3】　繰り返し

以下の実行例のように出力するプログラムを作成してください。
ソースファイル名は、CountDown.java とします。

条件：

- コマンドラインから整数値を入力する
- 入力した値が 0 になるまで値を 1 つずつ減らしながら、表示する
- 0 またはマイナス値を入力した場合は、その値を表示して終了する

実行例：

```
> java CountDown 3
count : 3
count : 2
count : 1
count : 0
finish!

> java CountDown 0
count : 0
finish!

> java CountDown -2
count : -2
finish!
```

【ステップ C-4】　繰り返し

以下の実行例のように出力するプログラムを作成してください。
ソースファイル名は、LoopSum.java とします。

条件：

- コマンドラインから整数値を 1 つ入力し、1 から入力した値までの合計を求めて表示する
- 1 より小さい数値が入力された場合は、入力された数値と、エラーメッセージとして "Please input again." を表示してプログラムを終了させる

実行例：

```
> java LoopSum 10
Input number : 10
-------------------------
Total of addition : 55

> java LoopSum 0
Input number : 0
-------------------------
Please input again.
```

【ステップ C-5】　繰り返し

ステップ B-3 で作成したプログラムを while 文を使った処理に変更してください。
ソースファイル名は、Star2.java とします。

【ステップ C-6】　繰り返し

ステップ C-5 で作成したプログラムを以下のように変更してください。
ソースファイル名は、Star3.java とします。

- 無限ループにする
- ループから抜け出す処理を追加する
- 終了条件は、入力した数字の数だけ * を表示したら、ループから抜ける

【ステップ C-7】　繰り返し

以下の実行例のように出力するプログラムを作成してください。
ソースファイル名は、Calc.java とします。

条件：

- 10 のべき乗を求めて表示する
- べき乗の計算過程を表示する
- べき乗の結果を long 型の変数に代入する

ヒント：

- n のべき乗とは n を、何回もかけ合わせることです。
  たとえば、3 の 4 乗は、3 を 4 回かけることで、3 × 3 × 3 × 3 = 81 になります。

実行例：

```
> java Calc
*** 10 のべき乗を求めます ***
10 の1乗は 10 です
10 の2乗は 100 です
10 の3乗は 1000 です
10 の4乗は 10000 です
10 の5乗は 100000 です
10 の6乗は 1000000 です
10 の7乗は 10000000 です
10 の8乗は 100000000 です
10 の9乗は 1000000000 です
10 の10乗は 10000000000 です
```

5 章実習課題

【ステップ A-1】　メソッドの定義/呼び出し

以下の実行例のように出力するプログラムを作成してください。
このプログラムはコマンドラインから文字列を入力すると、入力された文字列をメッセージとして表示します。
ソースファイル名は、DispMessage.java とします。

条件：

- **main() メソッド**
  - コマンドラインから入力された文字列が 1 つではない場合は、"Please input again." というメッセージを表示して終了する
  - コマンドラインから入力された文字列が 1 つの場合は、その文字列を disp() メソッドに渡す

  引　数：文字列型配列でコマンドライン引数を格納
  戻り値：なし

- **disp() メソッド**
  - 引数の文字列を出力する

  引　数：String 型のメッセージ
  戻り値：なし

実行例：

```
> java DispMessage Hello
Message : Hello

> java DispMessage Hello Good Bye
Please input again.

> java DispMessage
Please input again.
```

【ステップ A-2】　メソッドの定義/呼び出し

以下の実行例のように出力するプログラムを作成してください。
このプログラムは、コマンドラインから数値を 2 つ入力すると、入力された 2 つの値の大きさを比べて大きい方の値を表示します。
ソースファイル名は、CompareData.java とします。

条件：

- **main() メソッド**
  - コマンドラインから入力された 2 つの値を、 それぞれ int 型 に変換する
  - int 型に変換した 2 つの数値を compData() メソッドに渡して、結果を取得する
  - 取得した情報が 1 の場合は、"First input data is larger." というメッセージを表示する
  - 取得した情報が 2 の場合は、"Second input data is larger." というメッセージを表示する
  - 取得した情報が 0 の場合は、"Two input data is the same number." というメッセージを表示する

  引　数：文字列型配列でコマンドライン引数を格納
  戻り値：なし

- **compData() メソッド**
  - 2 つの引数のうち、1 つめの数値が大きい場合は 1 を返す
  - 2 つの引数のうち、2 つめの数値が大きい場合は 2 を返す
  - 2 つの引数の値が同じ場合は 0 を返す

  引　数：int 型の数値 1、int 型の数値 2
  戻り値：int 型の数値

実行例：

```
> java CompareData 794 710
Input data 1 : 794
Input data 2 : 710
First input data is larger.

> java CompareData 38 50
Input data 1 : 38
Input data 2 : 50
Second input data is larger.
```

【ステップ B-1】　メソッドの定義/呼び出し

以下の実行例のように出力するプログラムを作成してください。
このプログラムは、コマンドラインから支払額を入力すると、入力された支払額の大きさに応じて手数料を加算して表示します。
ソースファイル名は、Commission.java とします。

条件：

- **main() メソッド**
  - コマンドラインから入力された値を、int 型 に変換する
  - コマンドラインから入力された値が 1000 以上の場合は、calcComm() メソッドにその値を渡し、戻り値として手数料込みの金額を取得して表示する
  - 1000 より小さい数値が入力された場合は、エラーメッセージとして ″Please input 1000 or more.″ を表示し、プログラムを終了させる

  引　数：文字列型配列でコマンドライン引数を格納
  戻り値：なし

- **calcComm() メソッド**
  - 引数の数値が 2000 より小さい場合は、手数料として 10% を加算した値を返す
  - 引数の数値が 5000 より小さい場合は、手数料として 5% を加算した値を返す
  - 5000 以上の場合は、手数料として 3% を加算した値を返す

  引　数：int 型の数値
  戻り値：double 型の小数値

実行例：

```
> java Commission 1000
Your Payment        : 1000
Commission included : 1100.0

> java Commission 3000
Your Payment        : 3000
Commission included : 3150.0

> java Commission 7000
Your Payment        : 7000
Commission included : 7210.0

> java Commission 200
Your Payment        : 200
Please input 1000 or more.
```

【ステップ C-1】　メソッドの定義/呼び出し

以下の実行例のように出力するプログラムを作成してください。
このプログラムは、コマンドラインから数字を入力して、入力された数値が条件を満たすかどうかをチェックします。
ソースファイル名は、ArgsCheck.java とします。

条件：

- **main() メソッド**
  - コマンドラインの引数を入力し、check() メソッドで数値のチェックを行う
  - 戻り値が true の場合は、“コマンドライン引数の値 is OK!” というメッセージを表示する
  - 戻り値が false の場合は、“コマンドライン引数の値 is NG!” というメッセージを表示する
  - コマンドライン引数の数が 1 つではない場合は、エラーメッセージとして “Wrong Arguments” を表示してプログラムを終了する

  引　数：文字列型配列でコマンドライン引数を格納
  戻り値：なし

- **check() メソッド**
  - 数値が条件の範囲にあるかを確認し、範囲内にあれば true、範囲外であれば false を返す。
    条件は 100 以上 60000 以下の数値を true とする。

  引　数：int 型の数値
  戻り値：真偽値(boolean 値)

実行例：

```
> java ArgsCheck 80
80 is NG!

> java ArgsCheck 20000
20000 is OK!
```

【ステップ C-2】　メソッドの定義/呼び出し

以下の実行例のように出力するプログラムを作成してください。
このプログラムは、コマンドラインから縦、横の数値を入力して、# で直方形を表示します。
ソースファイル名は、Rectangle.java とします。

条件：

- **main() メソッド**
  - コマンドラインから縦と横を表す 2 つの数を入力し、drawRect() メソッドを呼び出す
  - コマンドラインの引数が 2 つ与えられていない場合は、エラーメッセージとして “Wrong Parameters” を表示してプログラムを終了する

  引　数：文字列型配列でコマンドライン引数を格納
  戻り値：なし

- **drawRect() メソッド**
  - 引数で渡された数値をもとに、縦、横を ’#’ で塗りつぶし、長方形を描く

  引　数：int 型の縦、int 型の横
  戻り値：なし

ヒント：

- drawRect() メソッドで、縦の繰り返し処理の中に、横の繰り返し処理を入れ子にする

実行例：

```
> java Rectangle 8 20
####################
####################
####################
####################
####################
####################
####################
####################
```

【ステップ C-3】　メソッドの定義/呼び出し

以下の実行例のように出力するプログラムを作成してください。
このプログラムは、コマンドラインから元金、利息利率（年率）、利息期間を入力して、利息を計算し、利息を含めた金額を表示します。なお、利息は複利法で算出するものとします。
ソースファイル名は、Saving.java とします。

条件：

- **main() メソッド**
  - コマンドライン引数として、元金、金利率、利息期間の 3 つの数値を入力する
  - コマンドライン引数で渡された各データの型は、元金は long 型、金利率は double 型、利息期間は int 型として扱う
  - これらの値を calculate() メソッドに渡し、戻り値として返された金額を表示する
  - コマンドライン引数が 3 つ与えられていない場合は、エラーメッセージとして、”Please input number : principal interest term” を表示し、プログラムを終了する

  引　数：文字列型配列でコマンドライン引数を格納
  戻り値：なし

- **calculate() メソッド**
  - 元金を指定の金利率、利息期間で複利計算し、利息込みの金額を返す

  引　数：long 型の元金、double 型の金利率、int 型の利息期間
  戻り値：double 型の利息計算後の金額

ヒント：

- calculate() メソッドで、計算時の利率は 1 + 金利率とし、単年の金額は、元金 * 利率で計算する

実行例：

```
> java Saving 100000 0.004 3
Principal      : 100000
Interest rate  : 0.0040
Term           : 3
Now Saving     : 101204
```

【ステップ C-4】　メソッドの定義/呼び出し

以下の実行例のように出力するプログラムを作成してください。
このプログラムは、コマンドラインから入力された数値にランダムな値を乗算した結果を表示します。
ソースファイル名は、Random.java とします。

条件：

- **main() メソッド**
  - コマンドライン引数として、数値を 1 つ入力し、Math.random() メソッドの戻り値を乗算した結果を表示する
  - コマンドラインの引数が 1 つも与えられていない場合は、メッセージとして “Wrong Parameters” を表示してプログラムを終了する

  引　数：文字列型配列でコマンドライン引数を格納
  戻り値：なし

- **Math.random() メソッド**
  - Math クラスのメソッドで、ランダムな数値を求める
  - コマンドラインの引数が 1 つも与えられていない場合は、メッセージとして “Wrong Parameters” を表示してプログラムを終了する

  引　数：なし
  戻り値：double 型の 0.0 から 1.0 までのランダムな数値

参考：　java.lang.Math クラスは、数学関連の機能を集めたクラスです。このクラスで提供されているメソッドの 1 つとして、ランダム値を求める random() メソッドがあります。

```
static double random()
```

0.0 以上 1.0 未満の間のランダムな値を返す。

使用例：

```
double d = Math.random();
```

実行例：

```
> java Random 100
Input number       : 100.0
Calculated number  : 42.00529225896671
> java Random 100
Input number       : 100.0
Calculated number  : 75.99352056767454
```

【ステップ C-5】　メソッドの定義/呼び出し

以下の実行例のように出力するプログラムを作成してください。
ソースファイル名は、Exchange.java とします。

条件：

- コマンドラインから入力された円を米ドル、ウォン、ユーロ、インドネシア・ルピアに変換する
- コマンドライン引数が 1 つも入力されなかった場合、または 2 つ以上入力された場合にはエラーメッセージを表示する
- 米ドル、ウォン、ユーロ、インドネシア・ルピアに変換するメソッドをそれぞれ作成し、それを呼び出す
- メソッドの引数は浮動小数型のデータとする
- 結果の表示は小数点第 1 位を 四捨五入して表示する
- 為替レートは次のようにする

変数	値
米ドル	123 円
韓国ウォン	0.1 円
ユーロ	121 円
インドネシア・ルピア	0.01円

参考：　四捨五入を行うには、値に 0.5　を加算してから java.lang.Math クラスの floor() メソッドを呼び出します。

```
static double floor(double a)
```

引数 a の小数点第 1 位を切り捨てた値を返します。

使用例：

```
double d = Math.floor(2.35);
```

実行例：

```
> java Exchange 10000
10000.0円を各国通貨に両替すると次のようになります
81.0ドル
100000.0ウォン
83.0ユーロ
1000000.0ルピア
```

総合演習

【ステップ A-1】

以下の実行例のように出力するプログラムを作成してください。
ソースファイル名は、Message.java とします。

条件：

- コマンドラインからいくつかの文字列を入力する
- 入力された文字列を順に表示する

実習課題のポイント：

- コマンドライン引数
- 繰り返し文

実行例：

```
> java Message Hello Good morning
Hello
Good
morning
```

【ステップ A-2】

以下の実行例のように出力するプログラムを作成してください。
このプログラムは、コマンドラインからデータを受け取り、各データの平均値を求めます。
ソースファイル名は、Average.java とします。

条件：

- コマンドラインから 3 つの数値データを受け取るようにする
- 受け取ったデータを int 型に変換し、変数に代入する
- 3 つのデータの平均を求めて、変数に代入する
- それぞれのデータを標準出力する
- コマンドラインから入力されるデータが 3 つでなかった場合、エラーメッセージを表示し、プログラムを終了させる

実習課題のポイント：

- コマンドライン引数
- 平均を求める演算
- それぞれの得点を数学、国語、英語、3 教科の平均点の順番で標準出力

ヒント：

- プログラムを終了させるには、System.exit(int n) メソッドを使用します。通常、引数に 0 以外の数値を指定すると、異常終了であることを示します。

実行例：

```
> java Average 100 90 80
数学・国語・英語の点数を計算します。
数学の点数     : 100
国語の点数     : 90
英語の点数     : 80
３教科の平均点 : 90.0

> java Average
３つの整数を入力してください
```

【ステップ A-3】

以下の実行例のように出力するプログラムを作成してください。
このプログラムは、コマンドラインから時刻として 0 から 23 までの数値を 1 つ入力し、入力された時刻に応じてその時刻に合った挨拶メッセージを標準出力します。
ソースファイル名は、Greeting.java とします。

条件：

- 6 から 11
  表示メッセージ：''おはようございます。''
- 12 から 18
  表示メッセージ：''こんにちは。''
- 0 から 5、19 から 23
  表示メッセージ：''こんばんは。''
- それ以外の数値
  表示メッセージ：''入力された時刻が間違っています。''

実習課題のポイント：

- コマンドライン引数
- if 文による値の比較
- switch 文による条件分岐

実行例：

```
> java Greeting 10
おはようございます。

> java Greeting 23
こんばんは。
```

【ステップ B-1】

網かけ部分にコードを挿入し、以下の実行例のように出力するプログラムを完成させてください。
ソースファイル名は、SunShop.java とします。

条件：

- ❑ if 文を使用しコマンドラインから受け取ったデータを処理する
  商品番号が入力されなかった場合、メッセージを出力する
  ・メッセージ：″ご希望の商品番号を入力して下さい。″
- ❑ switch 文を使用し商品番号 0 から 7 まで定義する
  それ以外の商品番号が入力された場合、メッセージ出力する
  ・メッセージ：″ご希望の商品は ： 0 から 7 までの商品番号指定が必要です。″

商品番号	商品名
0	Duke 時計
1	Java One T シャツ
2	Duke トレーナー
3	Duke ペン
4	Java キャップ
5	Sun ドライバーキット
6	Sun バッグ
7	Duke 人形

実習課題のポイント：

- ❑ コマンドライン引数
- ❑ 多分岐
- ❑ break 文

実行例：

```
> java SunShop 4
ご希望の商品は ： Java キャップです。

> java SunShop 8
ご希望の商品は ： 0 から 7 までの商品番号指定が必要です。

> java SunShop
ご注文希望の商品番号を入力して下さい。
```

完成させるプログラム SunShop.java

SunShop.java

```
 1.  // SunShop.java
 2.
 3.  class SunShop {
 4.    public static void main(String[] args){
 5.
 6.      int index = 0;
 7.      String name;
 8.
 9.      // コマンドラインから入力されたデータを int 型に変換し代入
10.      if(args.length == 1) {
11.        index = Integer.parseInt(args[0]);
12.      } else {
13.        System.out.println("ご注文希望の商品番号を入力して下さい。");
14.        System.exit(-1);
15.      }
16.
17.
18.
19.
20.
21.
22.
23.
24.
25.
26.
27.
28.
29.
30.
31.
32.
33.
34.
35.
36.
37.
38.
39.
40.
41.
42.
43.
44.
45.
46.
47.
48.      // 希望商品の標準出力
49.      System.out.println("ご希望の商品は ： " + name + "です。");
50.    }
51.  }
```

【ステップ B-2】

以下の実行例のように出力するプログラムを作成してください。
ソースファイル名は、SalesTax.java とします。

条件：

- 金額、年、月をコマンドラインから入力し、金額に対する税額を求める
- 税率は、2014 年 3 月までは 5%、2014 年 4 月以降は 8% として計算する

ヒント：

- 「2014 年 4 月以降」は、2014 年 4 月～ 12 月および 2014 年より先と考える

実習課題のポイント：

- コマンドライン引数
- if 文

実行例：

```
> java SalesTax 1000 2014 4
Input Price   : 1000
Input Year    : 2014
Input Month   : 4

Sales Tax : 80

> java SalesTax 1000 2014 3
Input Price   : 1000
Input Year    : 2014
Input Month   : 3

Sales Tax     : 50
```

【ステップ B-3】

以下の実行例のように出力するプログラムを作成してください。
ソースファイル名は、SearchData.java とします。

条件：

- コマンドラインから整数値を 1 つ入力し、その数値がコード内で用意した配列のデータと一致するかどうかを調べてメッセージを表示する
- 用意した配列のデータは以下の 5 つとします
  1111 , 3333 , 5555 , 7777 , 9999

ヒント：

- for 文では、配列の全てのデータと入力された数値が等しいかどうかをチェックする
- 等しいデータがあった場合は、事前に用意した変数 flag に 1 を代入して for 文を抜ける

実習課題のポイント：

- コマンドライン引数
- 繰り返し文
- break 文

実行例：

```
> java SearchData 1111
Input data : 1111
Data was found ! index: 0

> java SearchData 7777
Input data : 7777
Data was found ! index: 3

> java SearchData 8888
Input data : 8888
Data was not found.
```

【ステップ B-4】

以下の実行例のように出力するプログラムを作成してください。
ソースファイル名は、Triangle.java とします。

条件：

- コマンドラインから整数を入力すると、その行数分だけ # を表示する
- # で三角形を描く。各行で表示する # の個数は行が増えるごとに 1 ずつ増加
- 0 より小さい数値が入力された場合は、エラーメッセージを表示してプログラムを終了させる

実習課題のポイント：

- コマンドライン引数
- 入れ子のループ処理

実行例：

```
> java Triangle 3
Input number : 3
#
##
###

> java Triangle 5
Input number : 5
#
##
###
####
#####

> java  Triangle -1
Input number : -1
Please input again.
```

【ステップ C-1】

以下の実行例のように出力するプログラムを作成してください。
このプログラムは、コマンドラインからデータを受け取り、そのデータの階乗をメソッド処理によって求めて、表示します。
ソースファイル名は、Factorial.java とします。

条件：

- ❑ コマンド・ラインからデータを受け取るようにする
- ❑ コマンドラインから受け取ったデータが 0 以下ならばメッセージを表示する
  - ・ メッセージ：″ 0 より大きい値を入力して下さい。″
- ❑ 受け取ったデータを int 型に変換し、変数に代入する
- ❑ 階乗を求めるメソッドを定義する
  - ・ for 文を使い階乗を求める
  - ・ 戻り値は int 型
- ❑ 求めた階乗を表示する

ヒント：

- ❑ n の階乗とは 1 から n までのすべての整数の積です。10 の階乗は 10! と表現され、1 × 2 × 3 × 4 × 5 × 6 × 7 × 8 × 9 × 10 の値になります。

実習課題のポイント：

- ❑ コマンドライン引数
- ❑ メソッド定義

実行例：

```
>java Factorial 10
10 の階乗を表示します。： 3628800
```

【ステップ C-2】

以下の実行例のように出力するプログラムを作成してください。
このプログラムは、コマンドラインからデータを受け取り、最大公約数を求めます。
ソースファイル名は、Euclid.java とします。

条件：

- コマンドラインから入力されるデータが 2 つでなかった場合、メッセージを表示し、プログラムを終了する
  メッセージ：″ 2 つの整数を入力してください。″
- 入力されたデータを String 型 から int 型に変換する
- do-while 文を使用し 2 つの整数の最大公約数を求める

ヒント：

- 最大公約数の求め方
  1. a >= b の整数について、a を b で割った余りを c とする
  2. [a が b で割り切れる場合]
     b が a, b の最大公約数
  3. [a が b で割り切れない場合]
     b の値を新しい a とし、c の値を新しい b とする
  4. 1. に戻る

実習課題のポイント：

- 配列
- 繰り返し文

実行例：

```
> java Euclid 24 18
2つの整数の最大公約数 = 6
```

【ステップ C-3】

以下の実行例のように出力するプログラムを作成してください。
このプログラムは、コマンドラインからデータを受け取り、素数を求めます。
ソースプログラム名は、PrimeNumber.java とします。

条件：

- if 文を使用しコマンドラインから入力されるデータをチェックする
  - ・入力されたデータが 1 つであるかをチェックする
  - ・2 以下の整数が入力された場合メッセージを表示する
    メッセージ：〝2 より大きい数を入力してください。〟
- for 文を使用し繰り返し処理によりデータの素数判定をする
- for 文を使用し繰り返し処理により素数を表示する

ヒント：

- 素数とは、1 と自分自身以外には約数を持たない数のことです

実習課題のポイント：

- 分岐文
- 繰り返し文
- ネスティング
- 配列

実行例：

```
> java PrimeNumber 26
26 以下の素数を求めます。
素数 : 2
素数 : 3
素数 : 5
素数 : 7
素数 : 11
素数 : 13
素数 : 17
素数 : 19
素数 : 23
```

【ステップ C-4】

網かけ部分にコードを挿入し、以下の実行例のように出力するプログラムを完成させてください。
このプログラムは、各商品をグループ分けし表示します。
ソースファイル名は、SunJapanShop.java とします。

条件：

- 配列の配列を定義し値を商品名で初期化する
  - ・配列の配列を定義し値を初期化するには、以下のように定義する

    データ型 配列名[][] =

    {{グループ 1 の値 1, 値 2, . . . }, {グループ 2 の値 1, 値 2, . . . }, {グループ 3 の値 1, 値 2, . . . }};

グループ	商品名
1	Duke 時計、Duke トレーナー、Duke ペン、 Duke 人形
2	Java One T シャツ、Java キャップ
3	Sun ドライバーキット、Sun バッグ

- 繰り返し処理により、商品をグループ毎に表示する

実習課題のポイント：

- 配列
- 繰り返し文

実行例：

```
> java SunJapanShop
----- 今月のお薦め商品は -----
Duke 時計
Duke トレーナー
Duke ペン
Duke 人形
****** グループ別商品出力 *****
Java One T シャツ
Java キャップ
****** グループ別商品出力 *****
Sun ドライバーキット
Sun バッグ
****** グループ別商品出力 *****
```

完成させるプログラム SunJapanShop.java

SunJapanShop.java

```
 1.  // SunJapanShop.java
 2.
 3.  class SunJapanShop {
 4.    public static void main(String[] args){
 5.
 6.      //配列の定義
 7.
 8.
 9.
10.
11.      System.out.println("----- 今月のお薦め商品は -----");
12.
13.      // 配列データを繰り返し処理
14.
15.
16.
17.
18.
19.
20.    }
21.  }
```

【ステップ C-5】

以下の実行例のように出力するプログラムを作成してください。
このプログラムは、ハノイの塔を実現するプログラムです。
ソースプログラム名は、Hanoi.java とします。

条件：

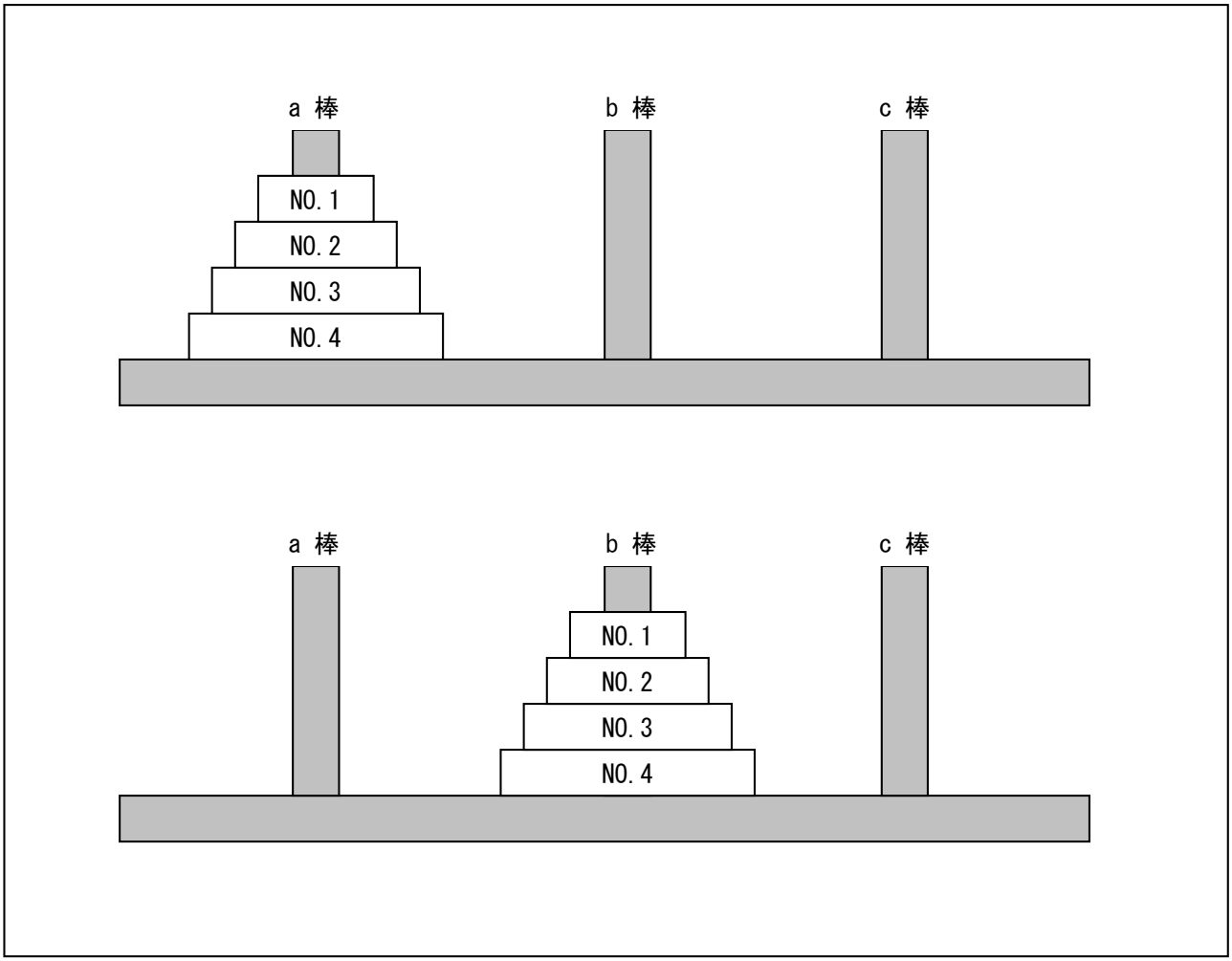
- 変数の初期化をする
- ルール
  3 本の棒（ここでは便宜的にa, b, c と呼ぶ）があり、棒 a には 4 枚の円板がかけられています。この円板を棒 b に 1 枚ずつ移動させます。しかし、移動途中で小さい円板には大きい円板は乗せることはできません
- 以上のルールを守り、3 本の棒を使用して 4 枚の円板を移動させてください
  今回のプログラムでは、何番目の円板をどこからどこの棒に移動しているのか表示してください

ヒント：

- ハノイの塔とは、フランスのパズル研究家Ｅ．リュカが考えたゲームです。プログラミングでは、再帰を実現する問題として取りあげらています。再帰とは、自分自身（n 次）を定義するのに、自分自身より 1 次低い集合（n-1 次）を用い、さらにその部分集合は、より低次の集合部分を用いて定義するということを繰り返す構造のことです。

実習課題のポイント：

- メソッドの定義
- メソッド定義内に再帰処理を定義



実行例：

```
> java Hanoi
円盤の枚数 4 枚
1 番目の板を a から c に移動
2 番目の板を a から b に移動
1 番目の板を c から b に移動
3 番目の板を a から c に移動
1 番目の板を b から a に移動
2 番目の板を b から c に移動
1 番目の板を a から c に移動
4 番目の板を a から b に移動
1 番目の板を c から b に移動
2 番目の板を c から a に移動
1 番目の板を b から a に移動
3 番目の板を c から b に移動
1 番目の板を a から c に移動
2 番目の板を a から b に移動
1 番目の板を c から b に移動
```

プログラミング・ロジック　実習課題　　実習テキスト

2016 年 8 月　RevisionB.1　第 1 刷
2017 年 3 月　RevisionB.2　誤植改訂

著者　　日本オラクル株式会社
　　オラクルユニバーシティ

発行所　　日本オラクル株式会社
　　オラクルユニバーシティ

Printed in Japan