● JKad16D「配列！」

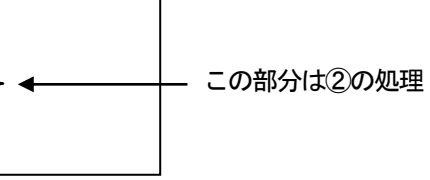
int 型配列（初期値：1、2、3）を宣言し、①②の処理を作成せよ。配列宣言は以下の通り。

```
int[] a = {1, 2, 3};          // 配列（初期値：1、2、3）の宣言
```

① 配列の各要素の値を表示する処理を作成せよ。
② 最初と最後の要素の値を入れ替えて表示する処理を追加せよ。

課題完成時の画面

```
a[0]：1
a[1]：2
a[2]：3
a[0]とa[2]を入れ替えます！
a[0]：3
a[1]：2
a[2]：1
```



● JKad16C「カードを表示しよう！」

カードを表す int 型配列 cards を宣言し、カードの枚数と各要素の値を表示する処理を作成せよ。

① 配列 cards の初期値を 1、2、3、4、5、6 としてカードの枚数と各要素の値を表示する処理を作成せよ。
② 初期値にさらに 7、8、9 を追加してカードの枚数と各要素の値を表示するように①の処理を変更せよ。

①まで完成したときの画面

```
カードが6枚あります！
カードを順番に表示します！
cards[0]：1
cards[1]：2
cards[2]：3
cards[3]：4
cards[4]：5
cards[5]：6
```

②まで完成したときの画面

```
カードが9枚あります！
カードを順番に表示します！
cards[0]：1
cards[1]：2
cards[2]：3
cards[3]：4
cards[4]：5
cards[5]：6
cards[6]：7
cards[7]：8
cards[8]：9
```

● JKad16B「ECC 長屋の住人」

ECC 長屋（部屋数 5）に以下の住人が住んでいる。

① 部屋番号を入力すると、その部屋の住人の名前を表示する処理を作成せよ。なお、入力と表示の処理は無限ループを使って繰り返すものとする。また、存在しない部屋番号（-1 とか 5 とか）を入力すると例外が発生するが、ここではそのままとする。
② 存在しない部屋番号を入力したとき、部屋番号を再入力させるように処理を追加せよ。プログラムの実行を終了させるにはウインドウ右上の[×]ボタンを押すこと。

ECC 長屋の住人

部屋番号	名前
0	ピタゴラス
1	アルキメデス
2	ユークリッド
3	エラトステネス
4	フィボナッチ

①まで完成したときの画面

```
何号室を見ますか？0
ピタゴラスが住んでいます！
何号室を見ますか？1
アルキメデスが住んでいます！
何号室を見ますか？2
ユークリッドが住んでいます！
何号室を見ますか？3
エラトステネスが住んでいます！
何号室を見ますか？4
フィボナッチが住んでいます！
何号室を見ますか？5
Exception in thread "main" java.lang.ArrayIndexOutOfBoundsException: Index 5 out of bounds for length 5
        at JKad16B.main(JKad16B.java:28)
```

不正な値（-1 や 5 など）を入力すると、例外（要はエラーのこと）が発生して、プログラムの実行が止まる。

②まで完成したときの画面

```
何号室を見ますか？-1
そんな部屋番号はありません！
何号室を見ますか？0
ピタゴラスが住んでいます！
何号室を見ますか？4
フィボナッチが住んでいます！
何号室を見ますか？5
そんな部屋番号はありません！
何号室を見ますか？
```

不正な値が入力されたときには再入力させる

プログラムの実行を終了させるにはウインドウ右上の[×]ボタンを押すこと

● JKad16A「最高得点を探せ！」

得点データを格納する int 型配列 scores を宣言し、最高得点を表示する処理を作成せよ。

```
int[] scores = {30, 50, 100, 70, 95};      // 得点データ（のび太、ジャイアン、出木杉、スネ夫、しずか）
```

課題完成時の画面

```
最高得点を探します！
最高得点は 100 点です！
```

● JKad16S1「一番背の高い人は最後尾へ！」

身長データを格納する int 型配列 heights を宣言し、一番大きい値が最後尾になるように配列データの入れ替えを行え。なお、入れ替えの過程で配列データの並び替えが発生しても構わないが、失われるデータがないようにすること。

```
int[] heights = {160, 155, 170, 150, 175, 180, 165};      // 身長データ
```

課題完成時の画面

```
一番背の高い人を一番後ろにします！
heights[0]：155
heights[1]：160
heights[2]：150
heights[3]：170
heights[4]：175
heights[5]：165
heights[6]：180
```

一番大きな値が 180 なので 180 が最後尾（heights[6]）になる。
データ入れ替えの過程でデータの並びが変わるのは構わないが（たとえば 160 と 155 は入れ替わっている）、データが失われないように注意すること。

● JKad16S2「カードシャッフル！」

カードを表す int 型配列 cards（初期値：1、2、3、4、5、6、7、8、9）を宣言し、カードをランダムに並べ替える処理を作成せよ。有効に並べ替えるには何回シャッフルすればいいのかは各自で考えること。

課題完成時の画面（乱数を使うのでこれとは異なる並べ替えになることもある）

```
カードを並べ替えます！
cards[0]：5
cards[1]：3
cards[2]：8
cards[3]：2
cards[4]：7
cards[5]：6
cards[6]：9
cards[7]：4
cards[8]：1
```

もともとは 1 から 9 まで順に並んでいたのが
シャッフルされてランダムに並ぶ。

● JKad16X「背の順に並べ！」

JKad16S1 を参考に、背の低い順(値の小さい順)に身長データを並び替える処理を作成せよ。身長データは JKad16S1 と同じものを使ってよい。

ヒント：
JKad16A では一番大きな値を heights[6]へ持って行ったので、残ったデータで一番大きな値を heights[5]へ持って行けば、heights[5]には 2 番目に大きな値が入る。残ったデータで一番大きな値を heights[4]へ持って行けば、heights[4]には 3 番目に大きな値が入る。これを繰り返せば・・・

課題完成時の画面

```
背の順に並べ替えます！
heights[0]：150
heights[1]：155
heights[2]：160
heights[3]：165
heights[4]：170
heights[5]：175
heights[6]：180
```