

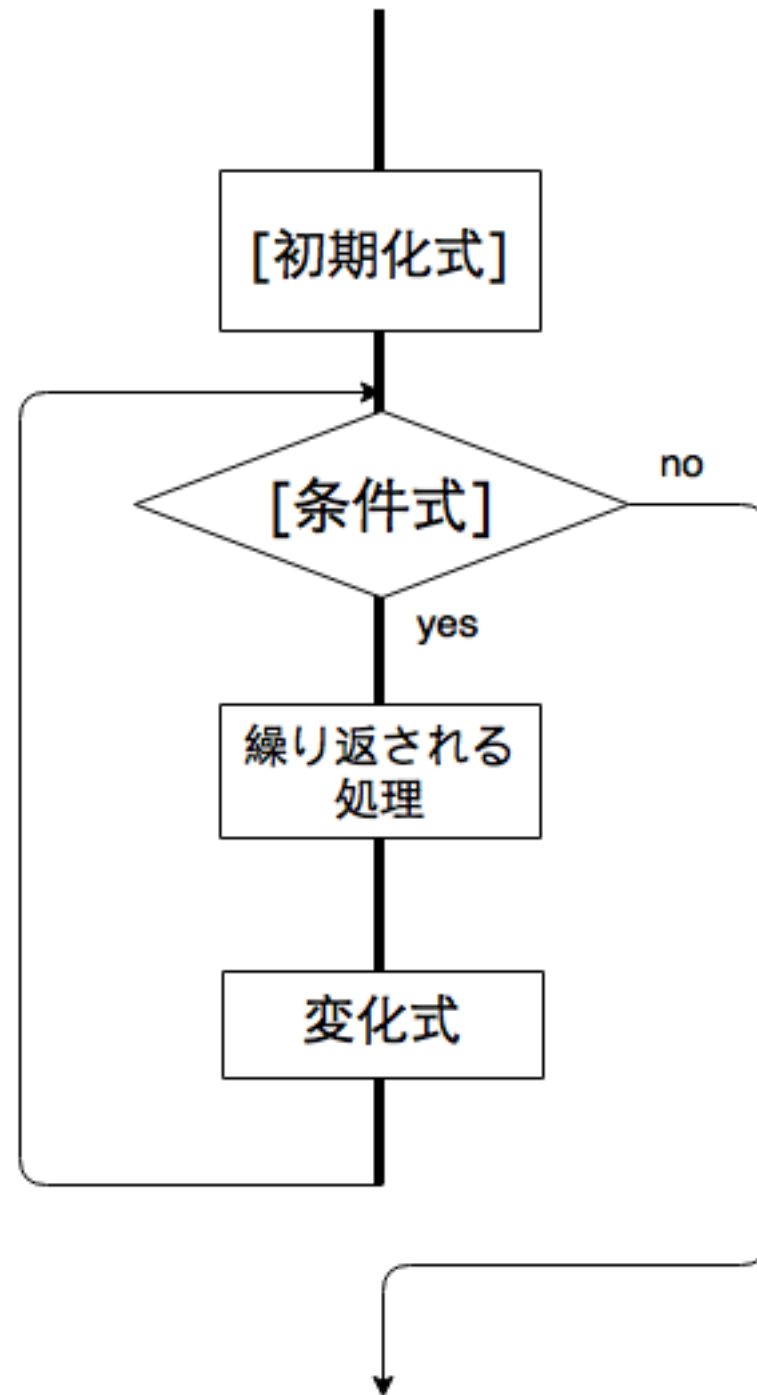
# Java 講習 5

# for文

```
for ( [初期化式]; [条件式]; 変化式){  
    繰り返される処理;  
}
```

## 無限ループ

```
for ( ;; ){  
    繰り返される処理;  
}
```



# for文とwhile文 の違い

- ・ 繰り返す数が決まっているか

決まっているなら

for

決まっていない

初期化処理の必要がない

変化式が必要ない



while

# 繰り返す数が決まっている場合

とある処理を10回行うプログラム

```
for ( int i=0; i<10 ; i++){  
    繰り返される処理;  
}
```

```
int i=0;  
while(i<10){  
    繰り返す処理;  
    i++;  
}
```

# 繰り返す数が決まっていない場合

0が入力されない限り繰り返す

```
Scanner sc = new Scanner(System.in);
int i = sc.nextInt();
while(i != 0){
    繰り返される処理;
    i = sc.nextInt();
}
```

```
Scanner sc = new Scanner(System.in);
int i = sc.nextInt();
for( ; i != 0; ){
    繰り返される処理;
    i = sc.nextInt();
}
```

# 配列

変数に似たもので、

変数・・・1つの値しか保持できない

```
int a;
```

```
int b = 2;
```

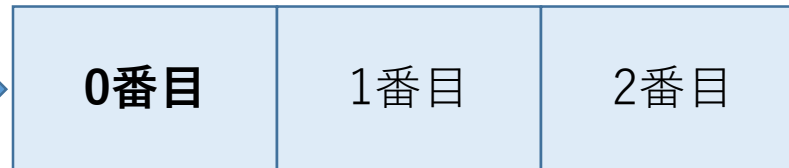
名前:a  
型:int

名前:b  
型:int

名前:c  
型:int

配列・・・複数の値を保持できる

名前:a  
型:int



# 配列の必要性

クラス10人分の国語の点を保持しておく変数を用意  
各変数に値を代入  
全員分の点数を出力

```
int japanese1, japanese2, japanese3, japanese4, japanese5 ...;  
japanese1 = 90;  
japanese2 = 50  
japanese3 = 80;  
    :  
System.out.println(japanese1);  
System.out.println(japanese2);  
    :
```

クラス10人分の国語の点を保持しておく変数を用意  
各変数に値を代入  
全員分の点数を出力



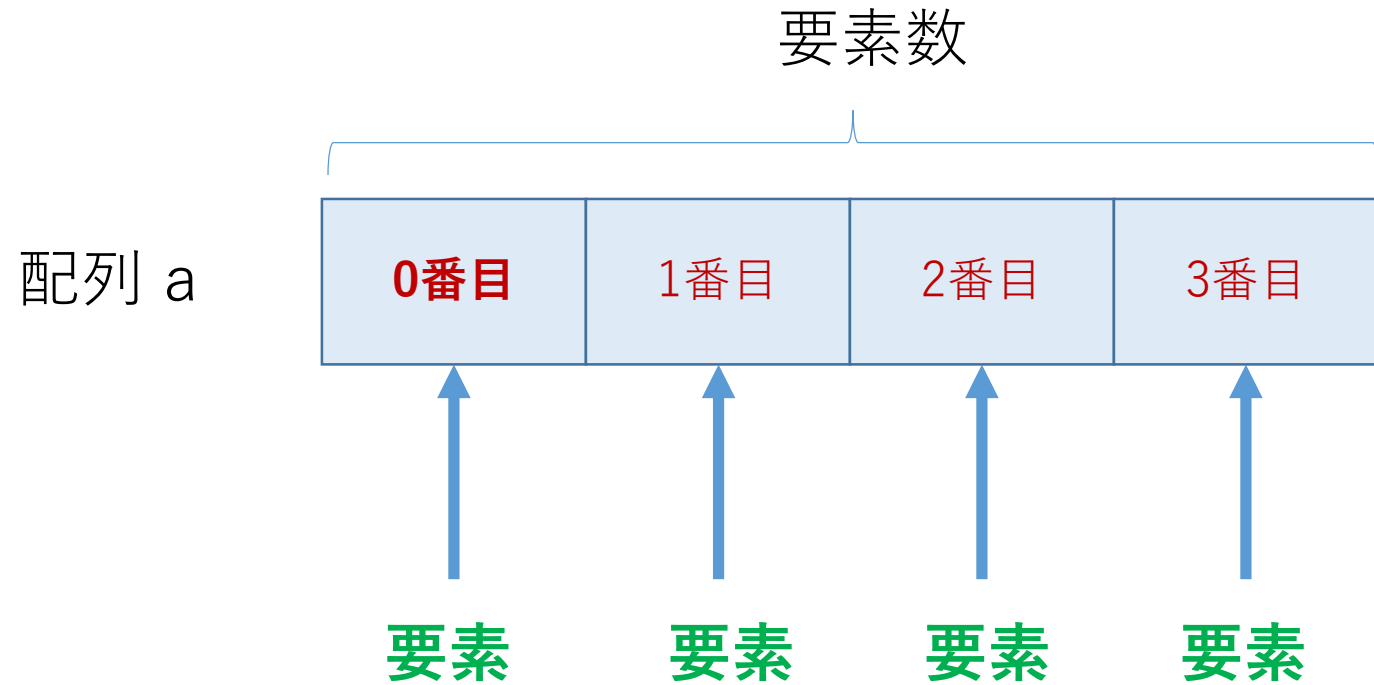
配列

```
int[] japanese = new int[10];  
japanese[0] = 90;  
japanese[1] = 50;  
japanese[2] = 80;  
:  
for(int i=0; i<10; i++){  
    System.out.println(japanese[i]);  
}
```



# 用語

**要素** . . . 一つの配列に確保される一つ一つ場所



**添え字** . . . 何番目の要素であるか

# 配列の作成方法 1

```
型名[]△変数名;  
変数名 = new△型名[要素数];
```

```
型名[]△変数名 = new△型名[要素数];
```

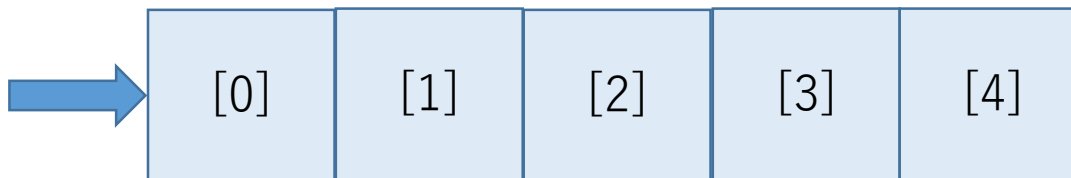
型:int

変数名:array

要素数:5

```
int[]△array = new△int[5];
```

名前:array  
型:int



```
型名[]△変数名    =    new△型名[要素数];
```

※あくまで場所を確保だけで中は値が入っていない

通常のint型の変数で表す・・・     int array;

# 最初から値を入れて作成する方法

通常のint型の変数で表す・・・ `int array = 0;`

**型名[]△変数名 = {初期値0,初期値1,初期値2, ... }**

~~型名[]△変数名;  
変数名 = {初期値0,初期値1,...};~~      型名[]△変数名;  
変数名 = new 型名[]{初期値0,初期値1,...};

型:int

変数名:array

要素数:5

初期値:0

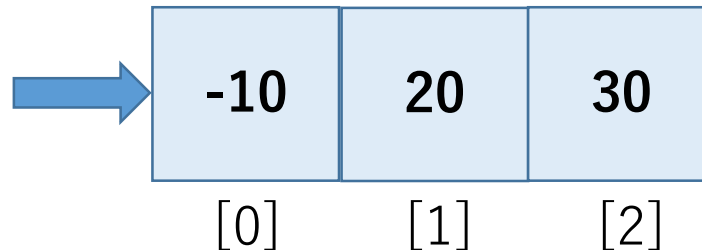
**`int[]△array = {0,0,0,0,0};`**

# 配列の使い方

- ・ 配列の変数名に添え字(何番目か)を[]に入れて書く

```
int ar = new int[3];  
ar[0] = 10;  
ar[1] = 20;  
ar[2] = 30;  
ar[0] = -10;
```

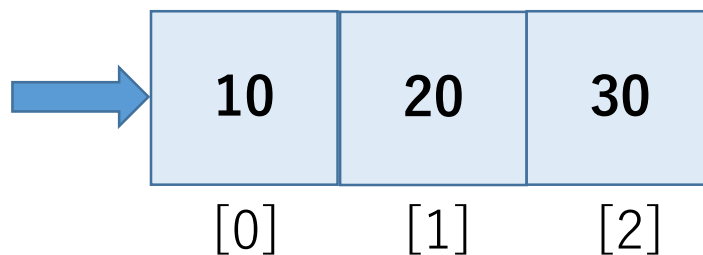
名前:array  
型:int



配列の添え字は変数を使うこともできる

```
int i = 0;  
int ar = new int[3];  
ar[i] = 10;  
i++; //iは1  
ar[i] = 20;  
i++; //iは2  
ar[i] = 30;
```

名前:array  
型:int



# 具体例

int型で要素数5の配列に好きな値を代入  
→配列に入っている値をすべて出力

```
public class Example{  
    public static void main(String[] args){  
        int[] ar = {50,43,85,90,65};  
        for(int i=0; i<5; i++){  
            System.out.println(ar[i]);  
            //System.out.printf("%d¥n",ar[i]); としても動く  
        }  
    }  
}
```

# 演習

具体例は配列の値をすべて出力していた  
→配列の中身をチェックして、偶数なら出力

## 1,具体例を改造し、 配列に入っている値が偶数なら出力するプログラム

ヒント:あまりは 割られる数%割る数 で求められる  
例:iが偶数か) `if(i%2 == 0)`

## 2, int型で要素数5の配列を用意し、プログラム実行後

キーボードから整数を入力し、配列に格納していく

→この配列の中で最大値を出力するプログラム

ヒント:常に最大値を保持する変数を用意しておく



# 解答例1

```
public△class△Example{  
    public△static△void△main(String[]△args){  
        int[]△ar = {50,43,85,90,65};  
        for(int i=0; i<5; i++){  
            if(ar[i]%2 == 0) //2で割り切れたら  
                System.out.println(ar[i]);  
        }  
    }  
}
```

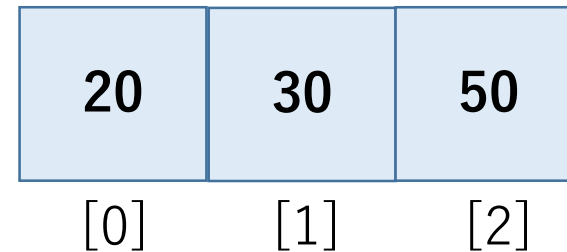
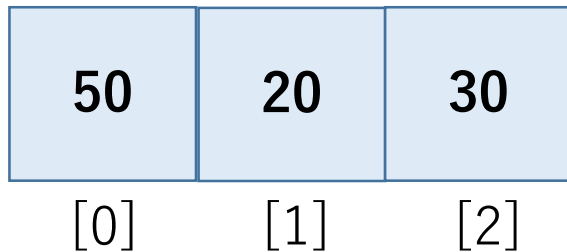
## 解答例2

```
import java.util.Scanner;
public class Example{
    public static void main(String[] args){
        Scanner sc = new Scanner(System.in);
        int[] ar = new int[5];
        int max; //変数maxに最大値を保持
        ar[0] = sc.nextInt(); //1回目ar[0]に値を入力
        max = ar[0]; //その値を現在の最大値とする
        for(int i=1; i<5; i++){ //3回繰り返す
            ar[i] = sc.nextInt();
            if(max < ar[i]) //もし変数maxの値より入力値が大きければ
                max = ar[i];
        }
        System.out.println(max);
    }
}
```

# 演習 バブルソート

3, int型で要素数10の配列を用意し、プログラム実行後

キーボードから整数を入力し、配列に格納していく  
その値を小さいほうから順になるように配列の値を入れ替える  
プログラム





小さいものを添え字0に近づけるには、大きい値を添え字の最大に近づけていけばいい


つまり


下の要素と比較し、上のほうが大きければ互いに交換する

1回目

0	<b>35</b>	 交換 なし
1	<b>50</b>	
2	45	
3	60	
4	30	

0	35	 交換
1	<b>50</b>	
2	<b>45</b>	
3	60	
4	30	

0	35	 交換 なし
1	45	
2	<b>50</b>	
3	<b>60</b>	
4	30	

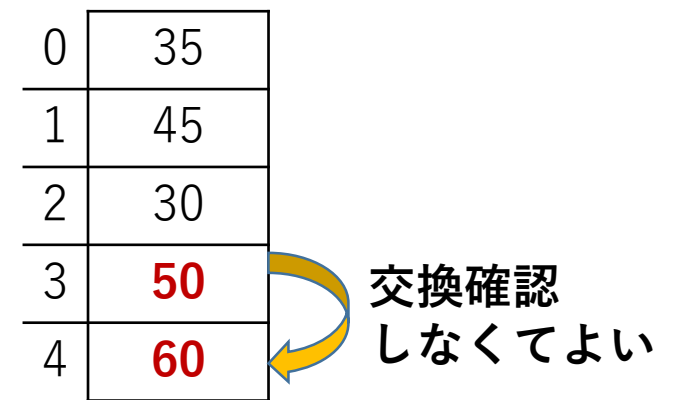
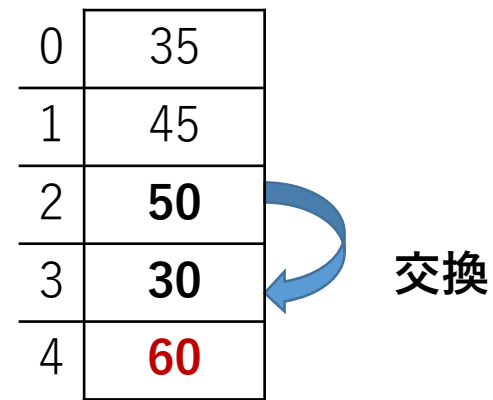
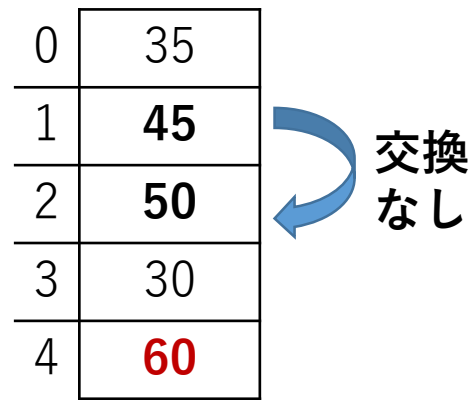
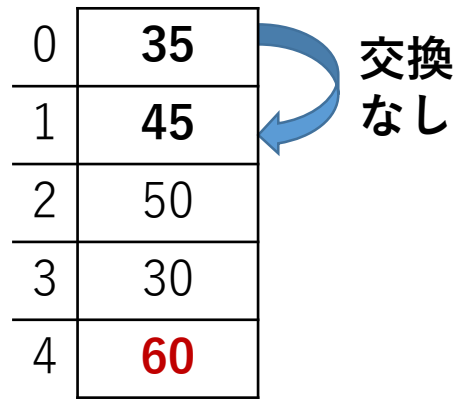
0	35	 交換
1	45	
2	50	
3	<b>60</b>	
4	<b>30</b>	

0	35
1	45
2	50
3	30
4	<b>60</b>

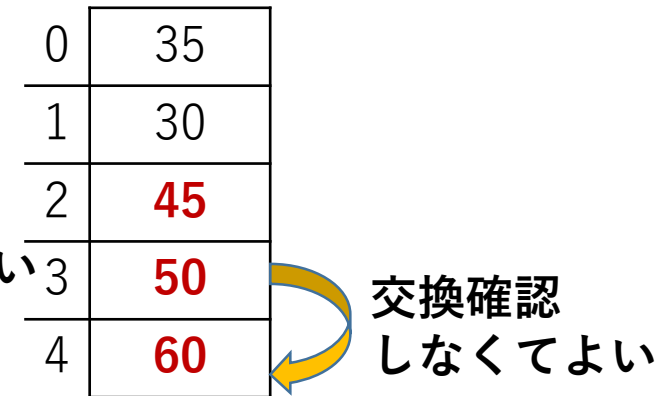
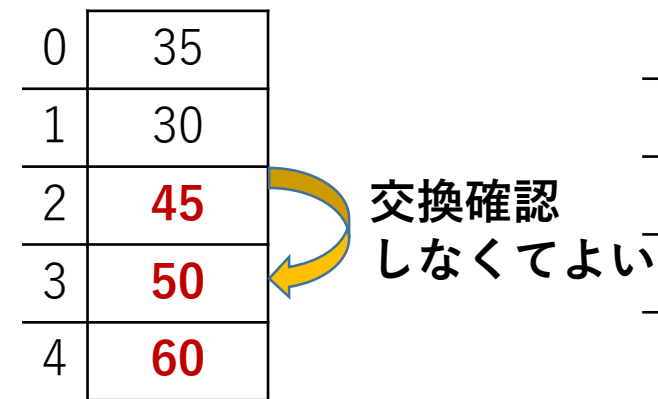
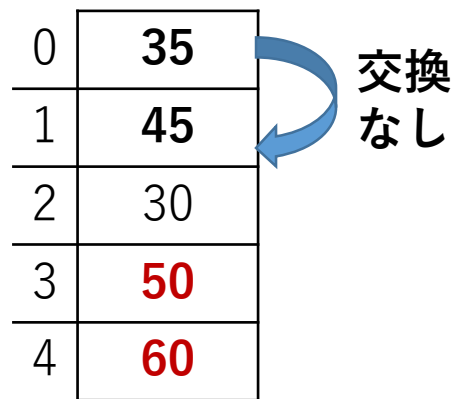
**必ず最大値が一番下に来る**

つまり次は最後の要素を除いて並び替えばよい  
for文は2重になる

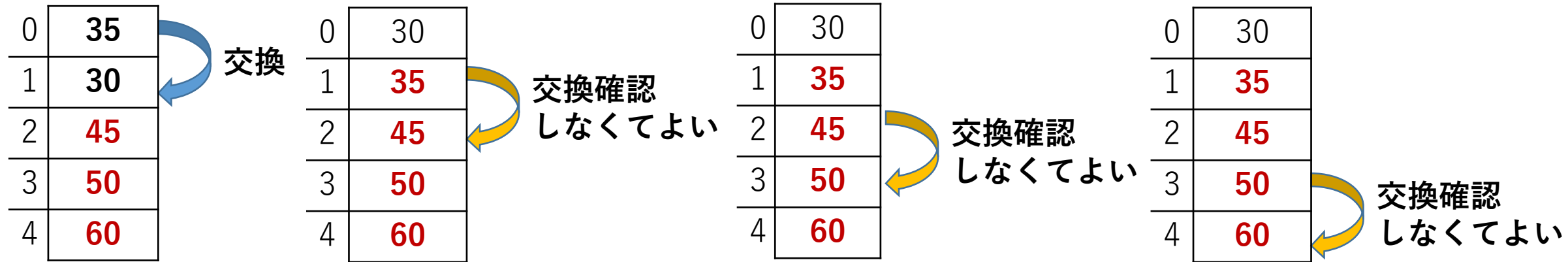
2回目



3回目



## 4回目



つまり、要素数が5であれば4回ほど [0]と[1],[1]と[2]のように  
「下の要素と比較し、上のほうが大きければ互いに交換する」という作業をする

要素数nの場合はn-1回なので,要素数10の場合を一部javaで書くと

```
for(int i=0; i< 10-1; i++){  
    //下の要素と比較し、上のほうが大きければ互いに交換する処理を繰り返す  
}  
となる
```

下の要素と比較し、上のほうが大きければ互いに交換する方法

i	35
i+1	50

下の要素と上の要素を比較する場合

上の要素をi番目とすると下の添え字はi+1となる

よって配列名arの場合

if(ar[i] > ar[i+1])となる　そして、この場合入れ替えるので

tmp = ar[i];

ar[i] = ar[i+1];

ar[i+1] = tmp;

という処理をすればよい

また、この処理を繰り返す回数は 要素数5の配列の場合  
5-1回である。

よって要素数nの場合n-1回である

さらに繰り返されるごとに1つつ調べなくてよい数が増加する  
つまり要素数10の配列の場合java言語では

```
for(int i=0;i<10-1; i++){  
    for(int j=0; j<10-1-i; j++){  
        //下の要素と比較し、上のほうが大きければ互いに交換する  
    }  
}
```

となる



# 解答例3

```
import java.util.Scanner;
public class Main{
    public static void main(String[] args){
        Scanner△sc = new△Scanner(System.in);
        int[]△ar = new△int[10];
        int△tmp;
        for(int△i=0; i<10; i++){
            ar[i] = sc.nextInt();
        }
        for(int△i=0; i<10-1; i++){
            for(int△j=0; j<10-1-i; j++){
                if(ar[j] > ar[j+1]){
                    tmp = ar[j];
                    ar[j] = ar[j+1];
                    ar[j+1] = tmp;
                }
            }
        }
        System.out.println();
        for(int△i=0; i<10; i++){
            System.out.println(ar[i]);
        }
    }
}
```

//int型の配列,10の要素数で作成  
//交換時に使用する一時的な保管場所  
  
//キ-ボードから入力  
  
//バブルソート  
  
//要素の比較  
//上側の要素が大きいなら交換  
  
  
  
  
  
  
  
//見やすくするために改行  
//入れ替え後の配列を出力