

6回目 if 文と if else 文

■ 今日の講義で学ぶ内容 ■


- ・ 関係演算子
- ・ if 文と if~else 文
- ・ if 文の入れ子

関係演算子

関係演算子	<code>==, !=, >, >=, <, <=</code>	2つのオペランド間の関係を評価して真(true)または偽(false)を判断します
		演算結果は boolean 型で true または false です
		例えば、 加算演算子の場合は <code>1 + 2</code> の演算結果は <code>3</code> 関係演算子の場合は <code>1 == 1</code> の演算結果は <code>true</code>

関係演算子とその意味

<code>a == b</code>	b が a に等しいとき true となります それ以外では false となります
<code>a != b</code>	b が a に等しくないとき true となります それ以外では false となります
<code>a > b</code>	b より a が大きいとき true となります それ以外では false となります
<code>a >= b</code>	b より a が大きいか等しいとき true となります それ以外では false となります
<code>a < b</code>	b より a が小さいとき true となります それ以外では false となります
<code>a <= b</code>	b より a が小さいか等しいとき true となります それ以外では false となります

 3つの関係演算子 `!=` `>=` `<=` のイコールはすべて右側に書きます
左右逆に書くとエラーになりますので注意しましょう

ソースコード例

ソースファイル名 : Sample6_1.java

```
// 関係演算子
class Sample6_1
{
    public static void main(String[] args)
    {
        boolean b0, b1, b2, b3;
        int i=1, j=10;
        double di=2.5, dj=5.0;

        // 3は5より小さいので false
        b0 = (5 < 3);

        // jはiより大きいので true
        b1 = (i < j);

        // dj*2とjは等しいので true
        b2 = (j == (dj*2));

        // すべての括弧を外して b2 = j==dj*2; としても演算子の優先順位より
        // 乗算->関係演算->代入の順番で演算が行われるため問題ありませんが
        // このように不明確な場合は括弧で優先順位を明示したほうが無難でしょう

        // djとdiは等しくないので true、その true と false は等しくないので false
        b3 = ((di != dj) == false);

        System.out.println("b0=" + b0 + ", b1=" + b1 + ", b2=" + b2 + ", b3=" + b3);
    }
}
// ※ 代入演算子 "=" と関係演算子 "==" を間違えないように !!
```

実行画面

```
>java Sample6_1
b0=false, b1=true, b2=true, b3=false
-- Press any key to exit (Input "c" to continue) --
```

条件判断文 1 if 文

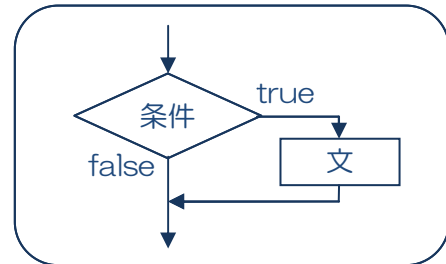
if 文

条件が true の場合、文を処理します

条件は boolean 型で、関係演算子で表現される式などを記述します

例えば、 $a < b$ 、 $a != 5$ など

```
if( 条件 ) 文
```



⚠ 文が記述できる場所には複数の文のまとまりとなる `{ }` (ブロック) を記述できます
※文は `;` (セミコロン) で終わる個々の処理のことです (2回目の講義を参照)

if 文は次のように記述することもできます

```
if( 条件 ) { 文1 文2 ... }
```

⚠ コード上の改行やスペース、タブ、改頁は処理に影響を与えません
ただし、キーワードや識別子、文字列リテラルなどまとまりある単語の途中を除きます

⚠ 改行やスペース、タブ、改頁のことをホワイトスペース (*whitespace*) といいます

⚠ ホワイトスペースを上手に用いて、
私たちが見やすく読みやすくなるようにコードを工夫しましょう
例えば、if 文は次のように書くと分かりやすいでしょう

```
if( 条件 )
{
    文1
    文2
    :
}
```

ソースコード例

ソースファイル名：Sample6_2.java

```
// if 文
import java.io.*;

class Sample6_2
{
    public static void main(String[] args) throws IOException
    {
        // キーボード入力の準備
        BufferedReader br;
        br = new BufferedReader(new InputStreamReader(System.in));

        int i;
        System.out.println("整数を入力してください。");
        i=Integer.parseInt(br.readLine());

        // 入力された値を if 文で判断し、1 であればブロック内を処理する
        if(i==1)
        {
            System.out.println("1 が入力されました。");
            System.out.println("1 が選択されました。");
        }

        System.out.println("処理を終了します。");
    }
}
```

実行画面

```
>java Sample6_2
整数を入力してください。
1
1 が入力されました。
1 が選択されました。
処理を終了します。
-- Press any key to exit (Input "c" to continue) --

>java Sample6_2
整数を入力してください。
2
処理を終了します。
-- Press any key to exit (Input "c" to continue) .
```

? 次にように if 文を記述するとどうなるでしょうか？

// if 文のよくあるミス

```
class Sample6_2_1
```

```
{
```

```
    public static void main(String[] args)
```

```
    {
```

```
        int i=0;
```

```
        // if 文のブロック {} を忘れたら？
```

```
        if(i==1)
```

```
            System.out.println("1 が入力されました。");
```

```
            System.out.println("1 が選択されました。");
```

```
        System.out.println("処理を終了します。¥n");
```

```
        // if 文ブロック前に ; (セミコロン) を入れてしまったら？
```

```
        if(i==1);
```

```
        {
```

```
            System.out.println("1 が入力されました。");
```

```
            System.out.println("1 が選択されました。");
```

```
        }
```

```
        System.out.println("処理を終了します。");
```

```
    }
```

```
}
```

if 文のブロック {} がない場合は、次の 1 行が if 文の条件が真のときに実行する文となります



文はセミコロンでおわる処理です。単独のセミコロンは処理のない文です

条件が真のときの実行する文が空の if 文となります。次に続くブロックは if 文の真／偽に関わらず常に実行される通常の文です

実行画面

```
>java Sample6_2_1
```

```
1 が選択されました。
```

```
処理を終了します。
```

```
1 が入力されました。
```

```
1 が選択されました。
```

```
処理を終了します。
```

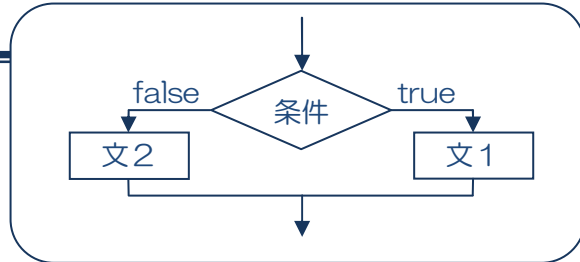
```
-- Press any key to exit (Input "c" to continue) --
```

条件判断文2 if ～ else 文

if ～ else 文

条件が true の場合、文 1 を処理し、
条件が false の場合、文 2 を処理します

```
if( 条件 ) 文1 else 文2
```



if～else 文はブロックを用いて次のように記述することもできます

```
if( 条件 ) { 文1 A 文1 B … } else { 文2 A 文2 B … }
```

または

```
if( 条件 )
{
    文1 A
    文1 B
    :
}
else
{
    文2 A
    文2 B
    :
}
```

ソースコード例

ソースファイル名：Sample6_3.java

```
// if ~ else 文
import java.io.*;

class Sample6_3
{
    public static void main(String[] args) throws IOException
    {
        // キーボード入力の準備
        BufferedReader br;
        br = new BufferedReader(new InputStreamReader(System.in));

        int i;
        System.out.println("整数を入力してください。");
        i=Integer.parseInt(br.readLine());

        // 入力された値を if ~ else 文で判断し、
        if(i==1) // 入力値が 1 であればこのブロックを処理する
        {
            System.out.println("1 が入力されました。");
            System.out.println("1 が選択されました。");
        }
        else // 入力値が 1 以外であればこのブロックを処理する
        {
            System.out.println("1 以外が入力されました。");
            System.out.println("1 を入力してください。");
        }

        System.out.println("処理を終了します。");
    }
}
```

実行画面 1

```
>java Sample6_3
整数を入力してください。
1
1 が入力されました。
1 が選択されました。
処理を終了します。
-- Press any key to exit (Input "c" to continue) --
```

実行画面2

```
>java Sample6_3
整数を入力してください。
2
1 以外が入力されました。
1 を入力してください。
処理を終了します。
-- Press any key to exit (Input "c" to continue) --
```

if ~ else 文の入れ子

if 文や if~else 文は、1 つの文です

if 文や if ~ else 文を他の if 文や if ~ else 文に入れることができます

```
if( 条件 ) if 文1 else if 文2
```



if 文は一つの文ですので、ブロックを書かなくてもエラーにはなりません
しかし、次のように if と else の対応が一意的ではなくなる場合があります

```
if ( 条件A ) if ( 条件B ) 文1 else 文2
```

このような場合、

Java は逐次的に else を else とまだ対応のない直前の if と対応付けます

```
if ( 条件A ) if ( 条件B ) 文1 else 文2
```

ユーザの解釈
外側に if~else 文
内側に if 文

Java の解釈
外側に if 文
内側に if~else 文

誤解を招きますので次のようにブロックを入れて明示しましょう

```
if ( 条件A ){ if ( 条件B ) 文1 } else 文2
```


ソースコード例

ソースファイル名：Sample6_4.java

```
// if ~ else 文の入れ子
import java.io.*;

class Sample6_4
{
    public static void main(String[] args) throws IOException
    {
        // キーボード入力の準備
        BufferedReader br;
        br = new BufferedReader(new InputStreamReader(System.in));

        int i;
        System.out.println("整数を入力してください。");
        i=Integer.parseInt(br.readLine());

        // 入力された値を if ~ else 文の入れ子で判断し、
        if(i==1) // 入力値が 1 であればこのブロックを処理する
        {
            System.out.println("1 が入力されました。");
            System.out.println("1 が選択されました。");
        }
        else // 入力値 1 以外で、
        {
            if(i==2) // 入力値が 2 であればこのブロックを処理する
            {
                System.out.println("2 が入力されました。");
                System.out.println("2 が選択されました。");
            }
            else // 2 でなければこのブロックを処理する
            {
                System.out.println("1 または 2 を入力してください。");
            }
        }

        System.out.println("処理を終了します。");
    }
}
```

実行画面

```
>java Sample6_4
整数を入力してください。
1
1 が入力されました。
1 が選択されました。
処理を終了します。
-- Press any key to exit (Input "c" to continue) --

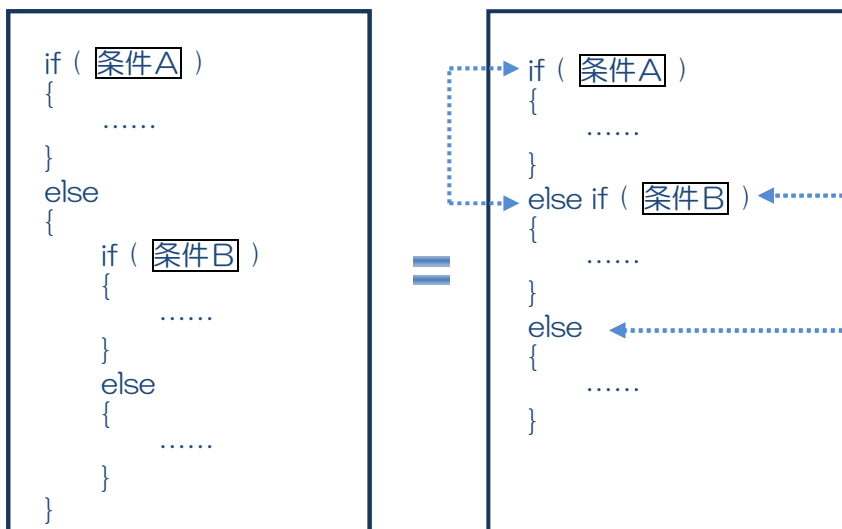
>java Sample6_4
整数を入力してください。
2
2 が入力されました。
2 が選択されました。
処理を終了します。
-- Press any key to exit (Input "c" to continue) --

>java Sample6_4
整数を入力してください。
3
1 または 2 を入力してください。
処理を終了します。
-- Press any key to exit (Input "c" to continue) --
```



慣れてきたら、Sample6_4.java で出てきた左側のような if～else 文の入れ子を右側のように書くとより読みやすくなり、良いでしょう

どちらも同じ処理ですので分かりやすい方で書いてください



Java の解釈によるifとelseの対応です

左側と同じ処理をしているのがわかりますね