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　==, !=, >, >=, <, <=

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 型です

boolean 型の変数には論理値リテラルの true と false を代入できます

たとえば、

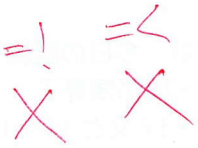
- 加算演算子では 1+2 の演算結果は 3 で数値ですが、
- 関係演算子では 1==1 の演算結果は true で論理値です

関係演算子とその意味

a == b	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
a != b	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
a > b	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
a >= b	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
a < b	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
a <= b	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

"==" は**関係演算子**で**等価**を表し、"=" は**代入演算子**で**代入**を表します

3 つの関係演算子 != >= <= の**イコール**はすべて**右側**に書きます
左右を逆にしたコードをコンパイルすると、
「式の開始が不正です」や「型の開始が不正です」
などの**コンパイルエラー**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 i1=1, i2=10;
        double d1=2.5, d2=5.0;

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

        // i2 は i1 より大きいので true
        b1 = (i1 < i2);

        // d2*2 と i2 は等しいので true
        b2 = (i2==(d2*2));

        // すべての括弧を外して b2 = i2==d2*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

        // d1 と d2 は等しくないので true、それと false は等しくないので false
        b3 = ((d1!=d2)==false);

        System.out.println("b0="+b0+", b1="+b1+", b2="+b2+", b3="+b3);
    }
}
```

関係演算子と型変換

関係演算子を用いた**式**を評価する場合も算術演算子と同様にオペランドの型の**拡大変換**が行われます
ただし、
演算結果は **boolean 型**です

関係演算子と代入演算子

代入演算子"="はイコールが**1 つ**で、**関係演算子**"=="はイコールが **2 つ**です
間違えないようにしましょう

実行画面

```
b0=false, b1=true, b2=true, b3=false
```

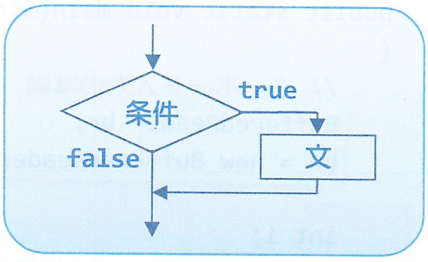
条件判断文 1 if 文

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
```

true / false

コード例
```
if(a<3)System.out.println(a);
```



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**個々の単一の処理や命令**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、字句を区切る役割は持ちますので、
- 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に入れたり、
  たとえば、○ double → × doub le
- キーワードや識別子を空白なしで続けて書くと**コンパイルエラー**になります
  たとえば、○ int a; → × inta;

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、見やすくなるようにコードを工夫しましょう
たとえば、**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 1

```
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。
```

実行画面 2

```
整数を入力してください。
2
処理を終了します。
```

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==2);
        {
            System.out.println("2 が入力されました。");
            System.out.println("2 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

次の 1 文が if 文の条件が真のときに実行する文と解釈されます

単独のセミコロン

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**空の文**です

条件が真のときに実行する文が**空の文**と解釈されます
次に続くブロックは if 文とは関係のない通常の文です

if(—)000;

if(—){000;}

実行画面

```
1 が選択されました。
処理を終了します。

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。
```

文法エラー

このようなミスは、Java の文法的には間違いではないためコンパイラエラーとして発見されることはありません。このようなエラーを**論理エラー**といいます。

論理エラーを防ぐために、**読みやすく分かり易いコードを書くこと**を心がけましょう!!

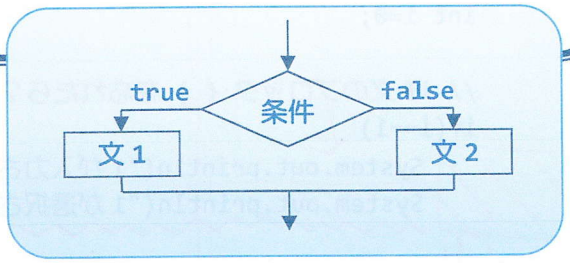
条件判断文 2 if~else 文

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

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 1 else 文 2
```

コード例 | `if(a<=0)a=0; else a=1;`



else は「条件が満たされなかった場合」という意味ですので、else の直後に
■ `if(a<=0) a=1; else(a>0) a=0;`
とすると、「文ではありません」という**コンパイルエラー**になりますので注意しましょう

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 1A 文 1B … } else {文 2A 文 2B … }
```

または

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

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

```
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。
```

実行画面 2

```
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。
```

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 文や **if~else 文**は **1 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**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
```

このような場合は、最初から**ブロックを付けて明示**すると誤解もなくなり良いでしょう

```
if(条件) { if 文 1 } else { if 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 1

```
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。
```

実行画面 2

```
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。
```

実行画面 3

```
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。
```

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

```
if ( 条件 A )
{
    文 1 ;
}
else
{
    if ( 条件 B )
    {
        文 2 ;
    }
    else
    {
        文 3 ;
    }
}
```

＝

```
if ( 条件 A )
{
    文 1 ;
}
else if ( 条件 B )
{
    文 2 ;
}
else
{
    文 3 ;
}
```

Java の解釈による if と else の対応

左側と同じ処理をしているのがわかりますね

今日の講義のまとめ

・関係演算子は、オペランドのどちらが大きいか小さいか、またそれらが等しいかどうかを演算します。演算結果は、真の場合には `true`、偽の場合には `false` です。

・`if` 文は条件判断を行います。条件が真の場合には、指定された文が実行されます。

・ホワイトスペースとは、改行やスペース、タブ、改項のことです。これらは処理に影響を与えませんので、上手に用いて見やすいコードを心がけましょう。

・`if~else` 文は、条件が真の場合に実行される文と、条件が偽の場合に実行される文の両方を指定できます。

・`if` 文や `if~else` 文を他の `if` 文などにいれて、入れ子にすることができます。