

7 回目 switch 文と論理演算子

● 条件判断文 3 switch 文

switch 文

式が case の値 (リテラル) と一致する場所から処理を開始し、直後の break;までを処理する。

どれにも一致しない場合、default;から処理を開始し、直後の break;までを処理する。

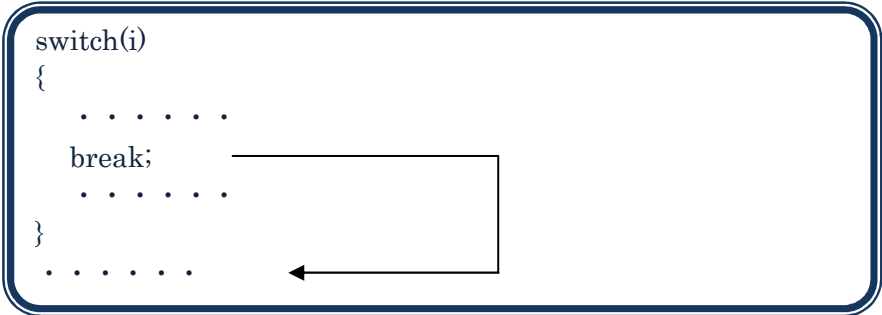
式やリテラル 1、リテラル 2 は

整数(byte 型, short 型, int 型)、または文字(char 型)である。

```
switch( 式 ) // ← if 文同様、この行はセミコロン無し！！
{
    case リテラル 1: // ← コロン “:” です。セミコロン “;” と間違えないように！！
        文 1;
        :
        break;
    case リテラル 2:
        文 2;
        :
        break;
        :
        :
        :
    default: // ← 一般に default: 行は最後を書く。また default 部分は省略できる。
        文 D;
        :
        break;
}
```

○ break 文

break 文 switch ブロック内の実行中の処理を強制的に終了し、ブロックから抜ける。
※繰り返し文 for 文や while 文、do-while 文から抜けるときも使用できる。



```
switch(i)
{
    . . . . .
    break;
    . . . . .
}
```

ソースコード例

ソースファイル名 : Sample7_1.java

```
// 入力値の判定
import java.io.*;

class Sample7_1
{
    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));

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

        switch(i) // 変数 i により処理を分岐
        {
            case 1: // i が 1 のとき、
                System.out.println("1 が入力されました。");
                break;
            case 2: // i が 2 のとき、
                System.out.println("2 が入力されました。");
                break;
            default: // i が 1 でも 2 でもないとき、
                System.out.println("1 か 2 を入力してください。");
                break;
        }
    }
}
```

実行画面

```
>java Sample7_1  
整数を入力してください。  
1  
1 が入力されました。  
-- Press any key to exit (Input "c" to continue) --
```

```
>java Sample7_1  
整数を入力してください。  
2  
2 が入力されました。  
-- Press any key to exit (Input "c" to continue) --
```

```
>java Sample7_1  
整数を入力してください。  
3  
1 か 2 を入力してください。  
-- Press any key to exit (Input "c" to continue) --
```

ソースコード例

ソースファイル名 : Sample7_2.java

```
// 入力文字の判定
import java.io.*;

class Sample7_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));

        // キーボード入力
        System.out.println("a か b を入力してください。");
        String str;
        str = br.readLine();
        //※ br.readLine():キーボード br から一行分の文字列を取り出す。

        // 文字列から一文字を取り出す
        char c;
        c = str.charAt(0);
        //※ str.charAt(x) : 文字列型変数 str に代入されている文字列の
        //   x 文字目の文字を取り出す。0 が文字列の先頭文字位置を表す。

        switch(c) // 変数 c により処理を分岐
        {
            case 'a': // c が'a'のとき、
                System.out.println("a が入力されました。");
                break;
            case 'b': // c が'b'のとき、
                System.out.println("b が入力されました。");
                break;
            default: // c が'a'でも'b'でもないとき、
                System.out.println("a か b を入力してください。");
                break;
        }
    }
}
```

実行画面

```
>java Sample7_2
a か b を入力してください。
a
a が入力されました。
-- Press any key to exit (Input "c" to continue) --

>java Sample7_2
a か b を入力してください。
b
b が入力されました。
-- Press any key to exit (Input "c" to continue) --

>java Sample7_2
a か b を入力してください。
c
a か b を入力してください。
-- Press any key to exit (Input "c" to continue) --
```

○ **switch** 文で **break**; を省略したらどうなる？

Sample7_1.java の **break**; を取り除いた場合の実行画面

```
>java Sample7_1
整数を入力してください。
1
1 が入力されました。
2 が入力されました。
1 か 2 を入力してください。
-- Press any key to exit (Input "c" to continue) --

>java Sample7_1
整数を入力してください。
2
2 が入力されました。
1 か 2 を入力してください。
-- Press any key to exit (Input "c" to continue) --

>java Sample7_1
整数を入力してください。
3
1 か 2 を入力してください。
-- Press any key to exit (Input "c" to continue) --
```

● 論理演算子

論理演算子

!, &&, ||

真(true)と偽(false)の間で論理演算を行う。
“～ではない”、“かつ”、“または”を表現する。

!	論理否定 「～ではない」	<table><tr><th>a</th><th>!a</th></tr><tr><td>true</td><td>false</td></tr><tr><td>false</td><td>true</td></tr></table>	a	!a	true	false	false	true	例えば、!(3<5)は false									
a	!a																	
true	false																	
false	true																	
&&	論理積 「かつ」	<table><tr><th>a</th><th>b</th><th>a&&b</th></tr><tr><td>true</td><td>true</td><td>true</td></tr><tr><td>true</td><td>false</td><td>false</td></tr><tr><td>false</td><td>true</td><td>false</td></tr><tr><td>false</td><td>false</td><td>false</td></tr></table>	a	b	a&&b	true	true	true	true	false	false	false	true	false	false	false	false	例えば、(1==0)&&(1<2)は false
a	b	a&&b																
true	true	true																
true	false	false																
false	true	false																
false	false	false																
	論理和 「または」	<table><tr><th>a</th><th>b</th><th>a b</th></tr><tr><td>true</td><td>true</td><td>true</td></tr><tr><td>true</td><td>false</td><td>true</td></tr><tr><td>false</td><td>true</td><td>true</td></tr><tr><td>false</td><td>false</td><td>false</td></tr></table>	a	b	a b	true	true	true	true	false	true	false	true	true	false	false	false	例えば、(1==0) (1<2)は true
a	b	a b																
true	true	true																
true	false	true																
false	true	true																
false	false	false																

但し、a と b は boolean 型の変数

ソースコード例

ソースファイル名 : Sample7_3.java

```
// 論理演算子の真理値表
class Sample7_3
{
    public static void main(String[] args)
    {
        System.out.println("!true = "+ (!true));
        System.out.println("!false = " + (!false));
        System.out.println("true && true = "+ (true && true));
        System.out.println("true && false = "+ (true && false));
        System.out.println("false && true = "+ (false && true));
        System.out.println("false && false = "+ (false && false));
        System.out.println("true || true = "+ (true || true));
        System.out.println("true || false = "+ (true || false));
        System.out.println("false || true = "+ (false || true));
        System.out.println("false || false = "+ (false || false));
    }
}
```

実行画面

```
>java Sample7_3
!true = false
!false = true
true && true = true
true && false = false
false && true = false
false && false = false
true || true = true
true || false = true
false || true = true
false || false = false
-- Press any key to exit (Input "c" to continue) --
```

ソースコード例

ソースファイル名 : Sample7_4.java

```
// 大文字・小文字の処理
import java.io.*;

class Sample7_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));

        // キーボード入力
        System.out.println("あなたは男性ですか？");
        System.out.println("Y か N を入力してください。");
        String str;
        str = br.readLine();

        // 文字列から一文字を取り出す
        char c;
        c = str.charAt(0); // 先頭の文字を取り出す
        //※ str.charAt(x) : 文字列型変数 str に代入されている文字列の
        //   x 文字目の文字を取り出す。0 が文字列の先頭文字位置を表す。

        if(c == 'Y' || c == 'y') // Y または y のとき、
        {
            System.out.println("あなたは男性ですね。");
        }
        else
        {
            if(c == 'N' || c == 'n') // N または n のとき、
            {
                System.out.println("あなたは女性ですね。");
            }
            else
            {
                System.out.println("Y か N を入力してください。");
            }
        }
    }
}
```


実行画面

```
>java Sample7_4  
あなたは男性ですか？  
Y か N を入力してください。  
Y  
あなたは男性ですね。  
-- Press any key to exit (Input "c" to continue) --
```

```
>java Sample7_4  
あなたは男性ですか？  
Y か N を入力してください。  
y  
あなたは男性ですね。  
-- Press any key to exit (Input "c" to continue) --
```

```
>java Sample7_4  
あなたは男性ですか？  
Y か N を入力してください。  
m  
Y か N を入力してください。  
-- Press any key to exit (Input "c" to continue) --
```

○ **switch** 文での **break**;の省略をうまく利用して **Sample7_4.java** は次のように表現できる。
ソースファイル名 : Ext7_1.java

```
// 大文字・小文字の処理 2
import java.io.*;

class Ext7_1
{
    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));

        // キーボード入力
        System.out.println("あなたは男性ですか?");
        System.out.println("Y か N を入力してください。");
        String str;
        str = br.readLine();

        // 文字列から一文字を取り出す
        char c;
        c = str.charAt(0); // 先頭の文字を取り出す
        //※ str.charAt(x) : 文字列型変数 str に代入されている文字列の
        //   x 文字目の文字を取り出す。0 が文字列の先頭文字位置を表す。

        switch(c)
        {
            case 'Y':
            case 'y': // Y または y のとき、
                System.out.println("あなたは男性ですね。");
                break;
            case 'N':
            case 'n': // N または n のとき、
                System.out.println("あなたは女性ですね。");
                break;
            default:
                System.out.println("Y か N を入力してください。");
        }
    }
}
```

● 条件演算子

条件演算子 `?:` 条件が真 (true) のとき式 1 が、
偽 (false) のとき式 2 が条件演算子の値となる。

条件は boolean 型であり、関係演算子で表現される式などを記述
例えば、`a<b`、`a!=5` など

`条件 ? true のときの式 1 : false のときの式 2`

○if 文との大きな違いは？

if 文	制御構造の一つ	→ 演算結果をもたない
条件演算子	演算子の一つ	→ 演算結果をもつため、式の一部に利用できる

例えば、

`ans = 条件 ? true のときの式 1 : false のときの式 2 ;`

のように代入文などの他の演算子と組み合わせて使用できる。

このコードを if~else 文を用いて書き下すと

```
if( 条件 )
{
    ans = true のときの式 1 ;
}
else
{
    ans = false のときの式 2 ;
}
```

となる。

ソースコード例

ソースファイル名 : Sample7_5.java

```
// 偶数・奇数の判定
import java.io.*;

class Sample7_5
{
    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));

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

        // 偶数・奇数の判断
        char c;
        c = ((i%2==0) ? 'E' : 'O');

        System.out.println("入力された整数は "+c+" です。(E)ven:偶数、(O)dd:奇数");
    }
}
```

実行画面

```
>java Sample7_5
整数を入力してください。
4
入力された整数は E です。(E)ven:偶数、(O)dd:奇数
-- Press any key to exit (Input "c" to continue) --

>java Sample7_5
整数を入力してください。
5
入力された整数は O です。(E)ven:偶数、(O)dd:奇数
-- Press any key to exit (Input "c" to continue) --
```