

第3章 「プログラムの流れの分岐」のまとめ

3.1 用語のまとめ

if文 (教 p.42)

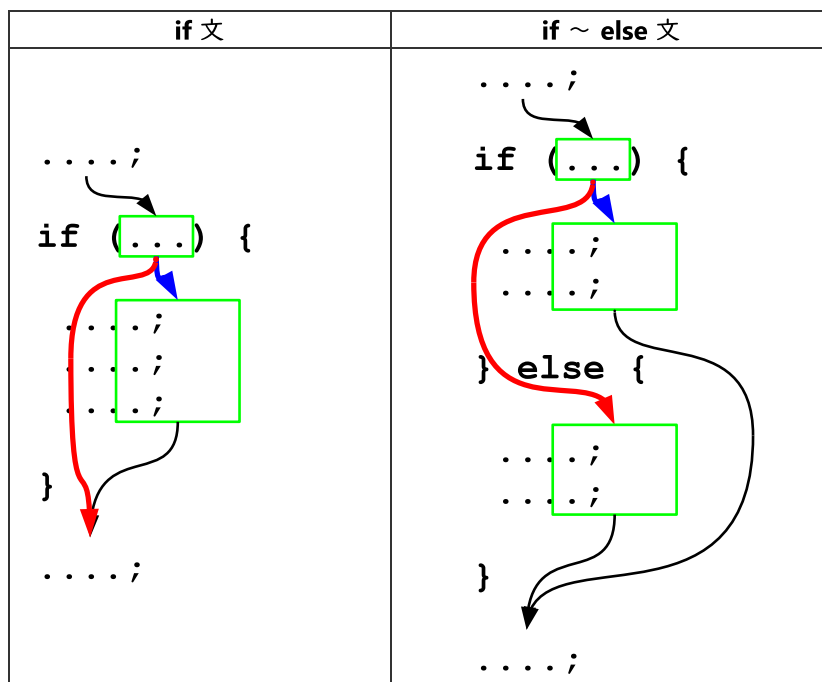
```
if ( 式1 ) 文1
```

という形のこと、式₁ を評価して、その値が 真 (すなわち真) であれば、文₁ を実行する。式の値が 偽 (すなわち偽) であるときは、何もしない

if ~ else文 (教 p.48)

```
if ( 式1 ) 文1 else 文2
```

という形のこと、式₁ を評価して、その値が 真 (すなわち真) であれば、文₁ を実行する。式の値が 偽 (すなわち偽) であるときは、文₂ を実行する。



Q 3.1.1 次のプログラムの断片の出力は? (出力なしのときは「無」と書く。)

- ① if (0) printf("X"); 答: 無
- ② if (0) printf("Y"); else printf("Z"); 答: Z

等価演算子 (教 p.48)

「==」は両辺の値が等しければ 真 (つまり真) を、等しくなければ 偽 (つまり偽) を返す演算子である。「==」と逆に等しくないかどうかを判定する演算子は「!=」である。

関係演算子 (教 p.50)

以下の4つがある。

<	左辺が右辺よりも小さいとき真
>	左辺が右辺よりも大きいとき真
<=	左辺が右辺よりも小さいか等しいとき真
>=	左辺が右辺よりも大きいか等しいとき真

「=<」とか「=>」という演算子はないので注意する。

入れ子になったif文 (教 p.51)

```

1  if (no == 0)
2      puts("その数は 0 です。");
3  else if (no > 0)
4      puts("その数は正です。");
5  else
6      puts("その数は負です。");
7

```

これは、単に `else` の次の文が、また `if` 文になっているだけのことである。

Q 3.1.2 もし、次のようになっていれば、`no` が 0 のときの出力はどうなるか？

```

1  if (no == 0)
2      puts("その数は 0 です。");
3  if (no > 0)
4      puts("その数は正です。");
5  else
6      puts("その数は負です。");
7

```

答: _____

評価 (教 p.53)

式の値を調べる (ために実行する) ことを _____ する (evaluate) という。

条件演算子 (三項演算子) (教 p.56)

```

式1 ? 式2 : 式3

```

まず 式₁ を評価し、その値が、

非 0 (真) であれば、_____ を評価して、その値を返す。_____ は評価しない。

0 (偽) であれば、_____ を評価して、その値を返す。_____ は評価しない。

複合文 (ブロック) (教 p.58)

文の並びを波括弧 (ブレース — 「`{`」 と 「`}`」 —) で囲んだものを複合文またはブロックという。(文のまえにいくつかの宣言があってもよい。) 複合文は構文上単一の文と見なされる。 複合文中の文は上 (左) から順に一つずつ実行される。

通常、if文の制御する文(後述のwhile文、for文などでも同様)は、たとえ一つの文でも(間違いを避けるため)波括弧で囲んでブロックにする。

△ 望ましくないスタイル	◎ 望ましいスタイル
<pre>if (n1 > n2) max = n1; else max = n2;</pre>	<pre>if (n1 > n2) { max = n1; } else { max = n2; }</pre>

教科書の例題は望ましいスタイルでないものが多いので、特に注意する。この授業の課題の解答は「望ましいスタイル」で提出すること。(教科書 p.59 下のほうの▷)(教 p.59)

(発展)ぶら下がりの else (dangling else)

```
1  if (h < 12) if (h < 6) printf("A"); else printf("B");
2
```

は、どのように文法的に解釈されるか?

解釈 1:

```
1  if (h < 12) { if (h < 6) printf("A"); else printf("B"); }
2
```

解釈 2:

```
1  if (h < 12) { if (h < 6) printf("A"); } else printf("B");
2
```

Q 3.1.3 上の解釈 1, 解釈 2 は h の値が以下のとき、どのように出力するか?

	解釈 1	解釈 2
h = 3		
h = 9		
h = 15		

ぶら下がりの else は、解釈 と同等である。つまり、近い方の if と対応する。

論理演算子 (教 p.60)

は以下のような演算子である。左右非対称である — つまり左オペランドを評価して値が決まれば、 は評価しない — ことに注意する (短絡評価)。

演算子	呼び方	説明
&&	論理 AND 演算子 かつ	左オペランドを評価して、0 (偽) であれば、0 (偽) を返す。非 0 (真) であれば、右オペランドを評価してその値を返す。
	論理 OR 演算子	左オペランドを評価して、非 0 (真) であればその値を返す。0 (偽) であれば、右オペランドを評価

	または	してその値を返す。
--	-----	-----------

Q 3.1.4 次のプログラム（の一部）の誤り（の可能性が高いところ）を指摘せよ。（教 p.63）

1.

```
if (a == 0);
    printf("hello"); /* _____ */
```

2.

```
if (a = 0)
    printf("hello"); /* _____ */
```

3.

```
if (a == b == c)
    printf("hello"); /* _____ */
```

4.

```
if (-1 <= a <= 1)
    printf("hello"); /* _____ */
```

5.

```
if (-1 <= a & a <= 1)
    printf("hello"); /* _____ */
```

switch 文 (教 p.64)

ある式の値 (整数型) によって、プログラムの流れを複数に分岐するときを使う。

switch (式 ₁) 文 ₁

文₁は、通常、複合文（ブロック）である。switch 文は式₁を評価して、文₁の中の _____ と「:」の間に書かれた定数と一致するところにジャンプする。（どの case にも一致しないときは、_____ にジャンプする。）ただし、break 文に出会うと、一気に switch 文を飛び出る。逆に break 文がなければ、そのまま次の文を実行する。

case ~: や default: のようにプログラムの飛び先を示す目印を _____ (名札) と呼ぶ。

3.2 プログラム例

繰り上がりの計算 (addtime.c)

```

1  #include <stdio.h>
2
3  int main(void) {
4      int hour1, minute1, hour2, minute2, hour3, minute3;
5
6      printf("hour1を入力して下さい:");
7      scanf("%d", &hour1);
8      printf("minute1を入力して下さい:");
9      scanf("%d", &minute1);
10     printf("hour2を入力して下さい:");
11     scanf("%d", &hour2);
12     printf("minute2を入力して下さい:");
13     scanf("%d", &minute2);
14
15     hour3  = hour1 + hour2;
16     minute3 = minute1 + minute2;
17
18     if (minute3 >= 60) {
```

```

19     hour3  = hour3 + 1 ;
20     minute3 = minute3 - 60;
21 }
22 printf("その和は、%d時間%d分です。\\n", hour3, minute3);
23 return 0;
24 }
25

```

2つの数を大きい順に並べる (maxswap.c)

```

1  #include <stdio.h>
2
3  int main(void) {
4      int n1, n2, tmp;
5
6      printf("整数1を入力して下さい:"); scanf("%d", &n1);
7      printf("整数2を入力して下さい:"); scanf("%d", &n2);
8
9      if (n2 > n1) { /* n1とn2を入れ換える */
10         tmp = n1;
11         n1 = n2;
12         n2 = tmp;
13     }
14     printf("大きい方は %dです。小さい方は %dです。\\n", n1, n2);
15     return 0;
16 }
17

```

3.3 文法のとめ

文 (statement)

に以下を追加、

分類	一般形	補足説明
if 文	if (式) 文	(教 p.42)
if ~ else文	if (式) 文 else 文	(教 p.44)
複合文(ブロック)	{ 宣言 ... 文 ... }	(教 p.58)
switch 文	switch (式) 文	(教 p.64)
ラベル付き文	case 整数リテラル : 文 default : 文	(教 p.65)
break 文	break ;	(教 p.65)

式 (expression)

に以下を追加、

分類	一般形	補足説明
三項演算子	式 ? 式 : 式	(教 p.56)

