

第3章 「プログラムの流れの分岐」のまとめ

3.1 「if 文」のまとめ

if 文 (教 p.44)

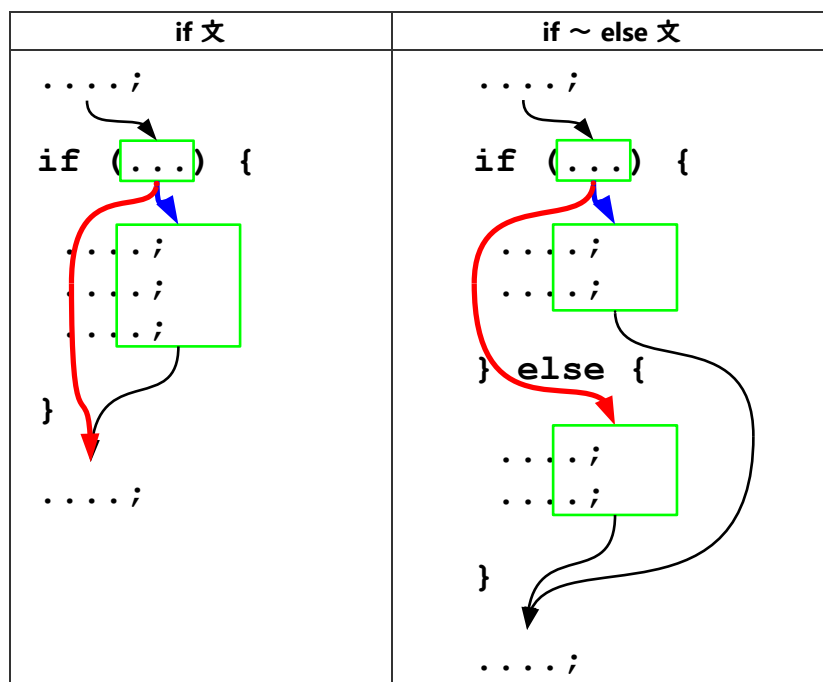
```
if ( 式1 ) 文1
```

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₁ の値が 偽 (すなわち偽) であるときは、何もしない。

else 付きの if 文 (教 p.46)

```
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₁ 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.50)

「==」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.52)

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等しいとき真

「 $\leq$ 」とか「 $\geq$ 」はもちろん、「 $=<$ 」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.52)

```
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。");
```

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？
改行は `\n` で表せ。

```
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。");
```

答: _____

評価 (教 p.55)

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.58)

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.60)

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。(文のほか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 do 文、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) printf("hello"); else printf("hi");</pre>	<pre>if (n1 > n2) { printf("hello"); } else { printf("hi"); }</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.61 下のほうの▷) (教 p.61)

繰り上がりの計算 (addtime.c)

```
1 #include <stdio.h>
2
3 int main(void) {
4     int hour1, minutel, hour2, minute2,
5         hour3, minute3;
6
7     printf("hour1 を入力して下さい:");
8     scanf("%d", &hour1);
9     printf("minutel を入力して下さい:");
10    scanf("%d", &minutel);
11    printf("hour2 を入力して下さい:");
12    scanf("%d", &hour2);
13    printf("minute2 を入力して下さい:");
14    scanf("%d", &minute2);
15
16    hour3 = hour1 + hour2;
17    minute3 = minutel + minute2;
18
19    if (minute3 >= 60) {
20        hour3 = hour3 + 1 ;
21        minute3 = minute3 - 60;
22    }
23    printf("その和は、%d 時間 %d 分です.\n",
24           hour3, minute3);
25    return 0;
26 }
```

Q 3.1.3 次のように入力したとき、出力はどうなるか

```
hour1 を入力して下さい: 1
minutel を入力して下さい: 40
hour2 を入力して下さい: 2
minute2 を入力して下さい: 30
```

その和は、__ 時間 __ 分です。

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",
7     &n1);
8     printf("整数 2 を入力して下さい:"); scanf("%d",
9     &n2);
10
11     if (n2 > n1) { /* n1 と n2 を入れ換える */
12         tmp = n1;
13         n1 = n2;
14         n2 = tmp;
15     }
16     printf("大きい方は %d です。小さい方は %d です.\n",
17           n1, n2);
18     return 0;
19 }
```

Q 3.1.4 10 ~ 12 行めが、もし

```
n2 = n1;
n1 = n2;
```

だったら、整数 1 が 3、整数 2 が 4 のとき、14 行目でどう出力されるか？
大きい方は __ です。小さい方は __ です。

(発展) ぶら下がりの else (dangling else)

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

は、どのように文法的に解釈されるか？

解釈 1:

```
1 if (h < 12) {
2     if (h < 6) {
3         printf("A");
4     } else {
5         printf("B");
6     }
7 }
```

解釈 2:

```
1 if (h < 12) {
2     if (h < 6) {
3         printf("A");
4     }
5 } else {
```

```
6 | printf("B");
7 | }
```

Q 3.1.5 上の解釈 1, 解釈 2 は h が以下の値のとき、どのように出力するか？
 （出力なしのときは「無」と書く。）

	解釈 1	解釈 2
h = 3		
h = 9		
h = 15		

さらに次の文の空欄を埋めよ。

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

論理演算子 (教 p.62)

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、1 を返す。非 0 (真) であれば、右オペランドを評価して、0 であれば 0 を、非 0 であれば 1 を返す。
	論理 OR 演算子 または	左オペランドを評価して、非 0 (真) であれば 1 を返す。0 (偽) であれば、右オペランドを評価して、0 であれば 0 を、非 0 であれば 1 を返す。

Q 3.1.6 次のプログラム（の一部）は誤っている（作成者の意図に反している）可能性が高い。このままだと、どう出力するか、`hello` と出力する場合は①、出力がない場合は②と書け。（教 p.65）

1.

```
int a = 1;
if (a == 0);
printf("hello");
```

2.

```
int a = 0;
if (a = 0)
printf("hello");
```

3.

```
int a = 2, b = 2, c = 2;
if (a == b == c)
printf("hello");
```

```
4. int a = 10;
   if (-1 <= a <= 1)
       printf("hello");
```

```
5. int a = 2;
   if (a & -1 <= a)
       printf("hello");
```

3.2 「switch 文」のまとめ

switch 文と break 文 (教 p.66)

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。

default の綴りを間違ってもコンパイルエラーにならないので注意すること。

Q 3.2.1 次のプログラムの断片の出力は?

```
1 int no = 2;
2
3 switch (no) {
4     case 1: printf("A");
5     case 2: printf("B");
6     case 3: printf("C");
7     default: printf("D");
8 }
```

文法のまとめ

文 (statement)

に以下を追加、

分類	一般形	補足説明
----	-----	------

if 文	if (式) 文	(教 p.44)
else 付き if 文	if (式) 文 else 文	(教 p.46)
複合文(ブロック)	{ 宣言 ... 文 ... }	(教 p.60)
switch 文	switch (式) 文	(教 p.66)
ラベル付き文	case 整数リテラル : 文 default : 文	(教 p.67)
break 文	break ;	(教 p.67)

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.58)
