第２章
型変換

負の値である符号付き整数 -1 が、正の値である符号無し整数 1U より小さいかどうかの比較を行う

```
-1 < 1U
```

の判定結果は《偽》となります。なんと、負の値である -1 は、正の値である 1U 以上であると判断されるのです。

異なる型の式に対する演算においては、暗黙の内に型変換が行われます。このような型変換は、どのようなタイミングで、どのように行われるのでしょうか。

本章では、型変換について学習します。

2-1　型変換

Aさんから、次のような質問をいただきました。

> 重大なバグにもつながりかねない現象が発生しています。
>
> **if**（-1 < 1**U**）
>
> という判定を行うと《偽》になります。どうやら、
>
> **if**（**int** 型の負値 < **unsigned int** 型の正値 ）
>
> の判定結果は《偽》となるようです。
>
> **if** 文の条件判定は、**int** 型に変換されて行われますから、右辺の値が、**int** 型が表現できる最大値を超えない限り《真》になるのではないでしょうか。
>
> これは、私の勘違いでしょうか。それとも、私の利用しているコンパイラのバグなのでしょうか。

思いがけない **if** 文の挙動に驚いたAさんは、**List 2-1** に示すプログラムを作成して、その動作を確認されたそうです。

List 2-1

```
/*
    符号付き整数と符号無し整数の比較
*/

#include  <stdio.h>

int main(void)
{
    int       sdata = -1;       /* 符号付き整数 */
    unsigned  udata =  1;       /* 符号無し整数 */

    printf("sdata < udata すなわち -1 < 1U は ");
    if (sdata < udata)
        puts("真。");
    else
        puts("偽。");

    printf("sdata < (int)udata すなわち -1 < (int)1U は ");
    if (sdata < (int)udata)
        puts("真。");
    else
        puts("偽。");

    return (0);
}
```

実行結果

```
sdata < udata すなわち -1 < 1U は 偽。
sdata < (int)udata すなわち -1 < (int)1U は 真。
```

▶　型指定子 **signed** あるいは **unsigned** を伴わない **short**，**int**，**long** 型は、符号付き型となりますから、ただの **int** 型は **signed int** 型のことを表します（『入門編』p.152）。

-1 は 1 より大きい？

符号付きの **int** 型である変数 *sdata* の値は -1 で、符号無しの **unsigned int** 型である変数 *udata* の値は 1 です。

最初の **if** 文は、問題となっている条件判定の挙動を確認するためのものです。実行結果から、以下の判定は《偽》であることが分かります。

```
sdata < udata                          /* -1 < 1U は偽 */
```

2 番目の **if** 文は、**unsigned int** 型である変数 *udata* の値をいったん **int** 型へとキャストして比較します。

```
sdata < (int)udata                     /* -1 < (int)1U は真 */
```

この判定の結果は《真》となることが実行結果からも確認できます。-1 や 1 でなく、-5 や 100 としても結果は同様です。

どうやら、**Fig.2-1** に示す関係が成立するようです。

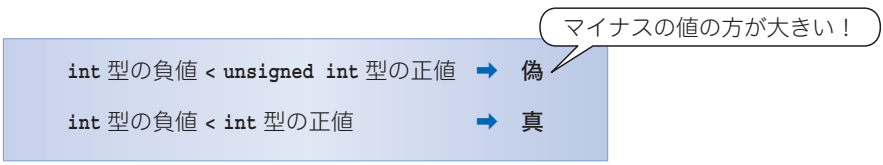

Fig.2-1　符号付き整数と符号無し整数の比較結果

if 文の評価

プログラム中の 2 番目の **if** 文と、

『**if** 文の条件判定は、**int** 型に変換されて行われますから…』

との文面を合わせて考えると、A さんは、次のように考えていらっしゃるようです。

『**if** 文では、両辺の値は **int** 型に変換されてから評価される。』

これは**誤り**です。二つの浮動小数点値の比較を行う判定を考えると分かります。

```
if (3.14 < 3.21)
```

もしも不等号の両側の式が **int** 型に変換した上で評価されるのであれば、

```
if ((int)3.14 < (int)3.21)
```

との比較が行われることになります。すなわち、

```
if (3 < 3)
```

と解釈されて、《偽》と判定されてしまいます。これは、おかしいですね。

＊

問題を解決するために、二つのポイントを学習することにします。それは、〔**if** 文〕に関するものと、〔関係式〕に関するものです。