C++ 003 標準入出力機能

【std::cout】

C++の標準出力機能で、コマンドプロンプトに文字列を出力します。

「std::cout」は std::ostream のインスタンスで、iostream ヘッダーファイルにグローバル変数で宣言されています。

```
std::cout << 文字列・変数 << std::endl
```

「<<演算子」を使用して、文字列や変数の値を出力します。

```
int var = 10;
std::cout << "値は" << var << std::endl;    //「値は 10」と出力
```

<<演算子を繋げて、文字列を連結できます。

std::endl

std::endl は改行（¥n）して出力用バッファを消去するマニピュレータ（入出力を調節する機能）です。

バッファの消去だけで改行したくない場合は「std::flush」を使用します。

【数値に関するマニピュレータ】

以下のマニピュレータは std::cout の数値に関する出力モードを指定します。

```
std::cout << std::dec << 10;                        //10 進数モード。「10」と出力
std::cout << std::oct << 10;                        //8 進数モード。「12」と出力
std::cout << std::hex << 10;                        //16 進数モード。「a」と出力
std::cout << std::showpoint << 10.0;                //小数を付ける。「10.000」と出力
std::cout << std::showpos << 10;                    //正数に+を付ける。「+10」と出力
std::cout << std::hex << std::showbase << 10;       //N 進数の接頭辞を付ける。「0xa」と出力
```

なお、テキストでは都合上、std::endl による改行を入れていません。

1 度指定したモードは切り替えられるまで続きます。

std::showpoint を無効にするには「std::noshowpoint」、std::showpos を無効にするには「std::noshowpos」、std::showbase を無効にするには「std::noshowbase」を使用します。

```
std::cout << std::hex;            //16 進数モードにする
std::cout << std::showbase;       //接頭辞を付ける
std::cout << 10;                  //16 進数モードなので「0xa」と出力
std::cout << 12;                  //16 進数モードなので「0xc」と出力
std::cout << std::dec;            //10 進数モードに切り替える
std::cout << std:: noshowbase;    //showbase を無効にする
```

【書式に関するマニピュレータ】

以下のマニピュレータは std::cout の書式に関する出力モードを指定します。

以下のマニピュレータを使用するには iomanip ヘッダーファイルをインクルードします。

```
//数値の桁数を 3 桁にする。「12.3」と出力
std::cout << std::setprecision(3) << 12.34;

//数値の桁数を 2 桁にして、それを小数点のみに適用する。「10.12」と出力
std::cout << std::setprecision(2) << std::setiosflags(std::ios::fixed) << 10.123;
```

端数は「最近接偶数への丸め」になります。

```
//出力幅を 5 桁に指定する。5 文字未満の場合はスペースが入って「  abc」と出力
std::cout << std::setw(5) << "abc";

//出力幅を 5 桁に指定する。5 文字未満の場合は 0 が入って「00123」と出力
std::cout << std::setw(5) << std::setfill('0') << 123;

//出力幅を 5 桁に指定して左詰めにする。「12300」と出力
std::cout << std::setiosflags(std::ios::left) << std::setw(5) <<   （続く）
std::setfill('0') << 123;
```

【std::cin】

C++の標準入力機能で、コマンドプロンプトからの入力を受けます。

「std::cin」は std::istream のインスタンスで、iostream ヘッダーファイルにグローバル変数で宣言されています。

```
std::cin >> 変数
```

「>>演算子」を使用して、入力された値を変数に代入します。

```
int var1, var2;
std::cin >> var1 >> var2;
```

>>演算子を繋げて、入力数を増やせます。

入力時にスペース区切りで「10 20」と入力すると変数 var1 に 10、変数 var2 に 20 が代入されます。

```
char str[100];
std::cin >> str;
```

>>演算子で文字配列にも代入できます。

std::cin は「空白を読みません」「空白までしか読みません」

上記の例文で「 aaa」（先頭に空白を入れている）と入力しても文字配列 str の値は「aaa」

「aaa bbb」（文字の間に空白を入れている）と入力すると文字配列 str の値は「aaa」になります。

```
char str[100];
std::cin.getline(str, 100);    //第 2 引数は取得文字数
```

空白も取得したい場合は「getline」を使用します。

```
int var;
std::cin >> var;                    //ここで数値以外を入力する
if (std::cin.fail() == true) {    //ここで真になる
    std::cout << "数値で入力してください" << std::endl;
    std::cin.clear();               //エラーフラグを戻す
}
```

入力された値が、代入される変数の型に合っているかどうかの判定には「fail」を使用します。

ただし上記の例文では実数値を入力した場合は整数値に変換されます。

「fail」のエラーフラグを「false」に戻す場合は「clear」を使用します。

＜バッファフラッシュに注意＞

```
char str[100];

//入力値の最初の 4 バイトを取得（¥0 含めて 5 バイト）
std::cin >> std::setw(5) >> str;
```

std::setw を使用すれば、取得文字数を指定できます。

```
char str1[100];
std::cin >> std::setw(5) >> str1;    //ここで 5 バイト以上の文字列を入力すると
char str2[100];
std::cin >> std::setw(5) >> str2;    //ここがスキップされる
```

最初の std::cin で 5 バイト以上の文字を入力すると入力用バッファに値が残ってしまい、バッファフラッシュが発生します。
これを抑えるためには「ignore」を使用します。

```
char str1[100];
std::cin >> std::setw(5) >> str1;     //ここで 5 バイト以上入力しても
std::cin.ignore(100, '¥n');           //バッファ内の 100 バイト分、改行までの値を消去
char str2[100];
std::cin >> std::setw(5) >> str2;     //スキップされない
```

「INT_MAX」は int 型で扱える最大値を示すマクロ。
std::cin.ignore(INT_MAX, '¥n');
と記述することもできます。