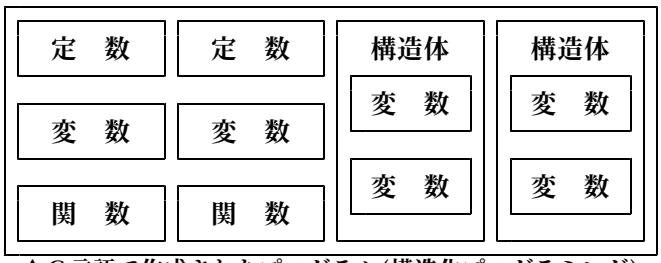
オブジェクト指向と
ゲームプログラミング

● 基礎編 － 第３回　Ｃ言語超入門 ●

Ｃ言語

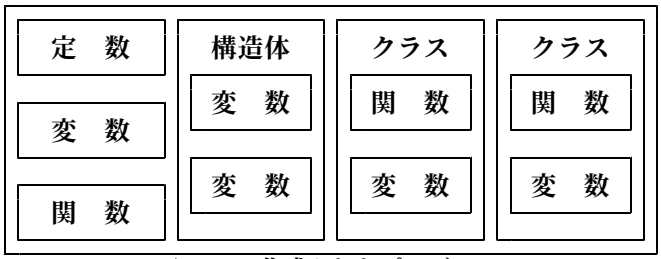
Ｃ言語は、1972年、AT&T社のベル研究所のD.M.RitchieとB.W.Kernighanによって開発された構造化プログラミングを行うための手続き型言語のひとつです。
Ｃ言語は、プログラムの機能を関数として記述し、データを変数や定数として記述します。基本的な制御文と豊富な演算子によって関数(function)を作り、その集まりをプログラムする関数型のプログラミング言語です。入出力も特別な命令があるわけではなく、標準で提供される関数(標準ライブラリ関数)に任されています。また、関連のあるデータをまとめることのできる構造体や、メモリアドレスを指し示すポインタなどもＣ言語の特徴の一つです。
一般的に、Ｃ言語はコンパイラ方式であり、入力されたプログラムは、コンピュータが実行できる形式に変換した後に実行されます。



▲Ｃ言語で作成されたプログラム(構造化プログラミング)

Ｃ++

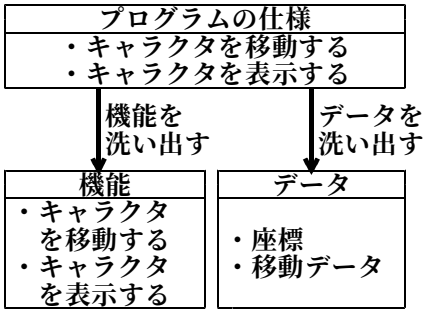
Ｃ++は、手続き型のＣ言語にオブジェクト指向の概念が取り入れられて改良されたオブジェクト指向言語です。AT&T社のベル研究所のBjarne Stroustrupによって、1980年頃から開発された「クラス付きＣ」とよばれた言語が元になっており、1983年にＣ++の名前になりました。
Ｃ++は、Ｃ言語にクラスを定義する機能を追加し、オブジェクト指向プログラミングを実現できるようにしたものです。したがって、Ｃ言語の機能だけで記述されたソースコードをＣ++のコンパイラでコンパイルすることもできます。構造体やポインタもそのまま使われています。Ｃ++は、Ｃ言語に慣れ親しんだプログラマが、構造化プログラミングからオブジェクト指向プログラミングに徐々に移行するためのものだと言えます。



▲Ｃ++で作成されたプログラム
(構造化プログラミング＋オブジェクト指向プログラミング)

構造化プログラミング

構造化プログラミングとは、プログラムの構造を機能(部品)単位に細分化するというものです。構造化プログラミングのための設計のことを構造化設計と呼びます。構造化設計では、プログラムの仕様の中に、どのような機能があるかを洗い出します。つまり、プログラムを機能の集まりと考えるのです。
たとえば、「キャラクタ移動プログラム」を作成するとしましょう。構造化設計では、プログラムの仕様の中から「キャラクタを移動する」「キャラクタを表示する」という機能を洗い出します。
構造化設計では、機能の洗い出しが終わったら、引き続きデータの洗い出しを行います。キャラクタ移動プログラムでは、「キャラクタの座標」および「移動データ」というデータが洗い出されます。



どのようなプログラムであっても、その構成要素は機能と機能の対象となるデータに大別されます。これは、構造化プログラミングでもオブジェクト指向プログラミングでも同じです。

関数

構造化プログラミングでは、プログラムを機能の集まりとして捉えますが、C言語では、機能を「関数」で記述します。

C言語は、関数の集まりで成り立っています。つまり、ある関数を実行することで、ある処理をするとも言えます。関数ひとつひとつは大した処理をしませんが、それらの関数を次々に実行することにより、複雑な処理を実現します。

C言語の関数は大きく２種類に分けることができます。C言語の規格であらかじめ用意されている標準ライブラリ関数と、プログラマが作成するユーザ関数です。標準ライブラリ関数には、画面に文字を表示するprintf関数をはじめ、文字列操作、ファイルの読み書き、メモリの読み書き、数学関数などさまざまな関数が提供されています。しかし、標準ライブラリ関数だけでは足りないので、独自の関数を作成できるようになっています。

変数

変数とは、プログラムで計算を行うときに使う数値を記憶するために、メモリに数値を角のするための箱のことをいいます。この箱(変数)を介してさまざまな演算を行うことができます。

```
int   a = 8, b = 9, c = 3;
```

a[8]　　b[9]　　c[3]

変数に付ける名前のことを変数名といいます。プログラムは変数名によって変数の区別を行います。上の例では、a, b, cという変数名がつけられ、それぞれ８，９，３が入っています。

また、変数には型という属性があります。型によって、記憶できる数値の種類と範囲が異なります。C言語では、基本の型として以下の４つの型があります。上の変数a, b, cはint型なので、約±２１億の数値を記憶できます。

型名	説明	Windows上での値の範囲
char	文字を格納	-128～127
int	符号付き整数	-2147483684～2147483647
float	浮動小数点	±10の±38乗(有効桁数7桁)
double	倍精度浮動小数点	±10の±308乗(有効桁数15桁)

コンピュータは無限に大きな数字や、無限に大きな精度の数を記憶することができません。上の表のようにそれぞれの型には一定の範囲内の数しか格納することができないようになっています。

演算子

C言語の演算子は、加算と減算はそれぞれ+と-を用いますが、乗算は×ではなく*を用います。また、キーボードに÷のキーがないので、除算は/(スラッシュ)で代用します。すなわち、a × bはa * bと記述し、a ÷ bはa / bと記述します。括弧があれば、その中が先に計算されます。代表的な演算子を紹介します。

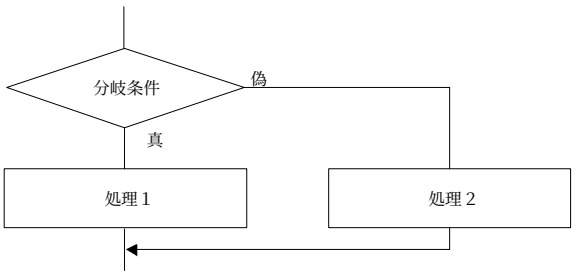
算術演算子	Cでの記号	意味	備考
＋	+	加算	
－	-	減算	
×	*	乗算	
÷	/	除算	
余り	%	余り	整数のみ使用可

制御文

制御文とは、プログラムの流れを制御するための文です。C言語では、上から順番にプログラムが実行されますが、制御文を使うことにより、プログラムを分岐させたり、反復したりすることができます。
分岐させる制御文としてif-else、switch-caseがあり、反復(ループ)させる制御文としてwhile、do-while、forがあります。

１．if-else

if-else文は、以下のように処理されます。条件が成り立つ場合は「処理１」が行われ、成り立たない場合は「処理２」が実行されます。

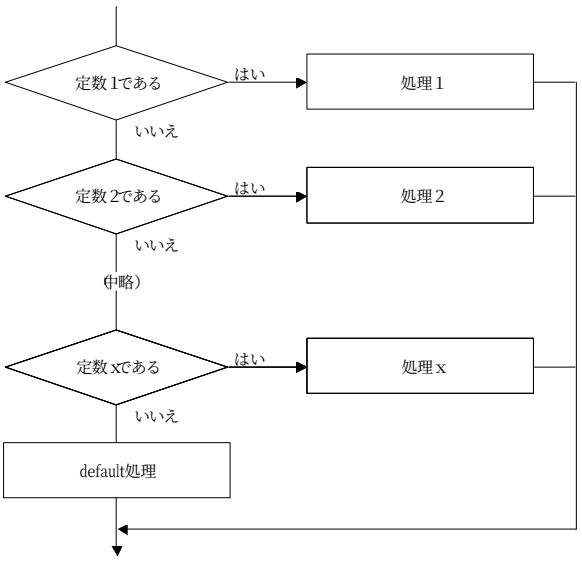


if-else文の書式は以下のようになります。else文以降を省略する場合もあります。

```
if(条件式) {
    条件式を満たしたときに実行する処理
} else {
    条件式を満たさなかったときに実行する処理
}
```

２．switch-case

switch文は、以下のように処理されます。値によって実行する処理を多分岐させることができます。



switch-case文の書式は以下のようになります。

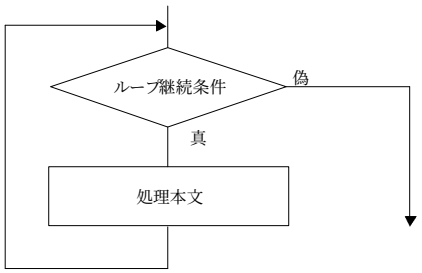
```
switch(変数) {
case 定数式１:
     処理１
     break;

case 定数式２:
     処理２
     break;
  (中略)
case 定数式ｘ:
     処理ｘ
     break;

default:
     すべての定数式を満たさなかったときの処理
     break;
}
```

３．while

while文は、以下のように処理が行われます。ループに入る前に条件判定を行い、条件が成り立つ間処理を反復します。

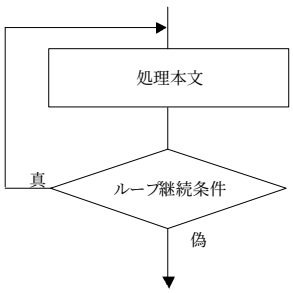


while文の書式は以下のようになります。

```
while(ループ継続条件) {
     処理本文
}
```

４．do-while

do-while文は、ループの最後に条件判定を行い、条件が成り立つ間処理を反復します。

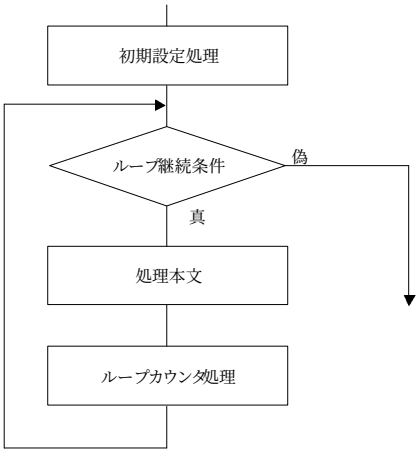


do-while文の書式は以下のようになります。while文の最後に、";"（セミコロン）が必要なことに注意してください。

```
do {
     処理本文
} while(ループ継続条件);
```

５．for

for文は、以下のように処理されます。条件が成り立つ間、処理を反復させることができます。



for文は、指定した回数だけ反復させたいときに使用します。反復回数を管理するために、ループカウンタという変数を使用します。for文の書式は以下のようになります。

```
for(初期設定処理；ループ継続条件；ループカウンタ処理）{
    反復させたい処理
}
```