Ｃ言語入門講座（第１回）

2011/5/9

①コンピューターの構成要素

〇入出力装置

外部から必要な情報や命令をコンピューターに入力したり、
処理した結果などを外部に出力したりする装置のこと。
・入力装置の例　→　キーボード、マウス
・出力装置の例　→　ディスプレイ（画面）、プリンタ

〇記憶装置

プログラムや情報を一時的あるいは半永久的に保存する装置。
CPU が直接読み書きできる主記憶装置(メモリ)と
データの保管などに使われる補助記憶装置(ストレージ)がある。
HDD や CD、DVD などは補助記憶装置である。

〇演算装置

CPU の一部で実際に計算を行う装置。

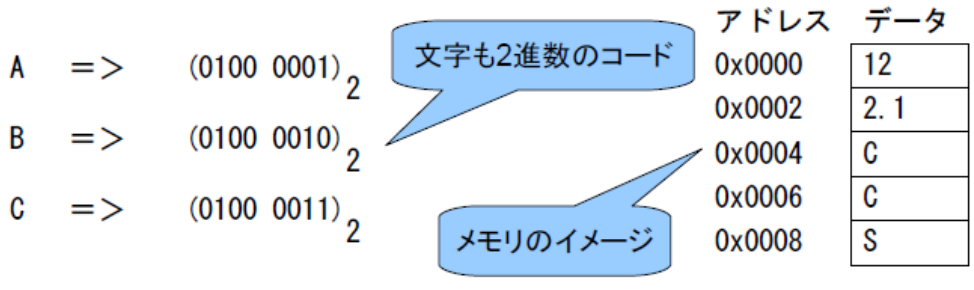
〇制御装置

CPU の一部で演算装置や入出力装置、記憶装置の読み書きなどを制御する装置。

②コンピューターの内部データ

内部でのデータ保持には“電気が流れている”　または“電気が流れていない”の２通りしかできない。つまり、コンピューター内部のデータは全て**２進数**で扱われる。
２進数の１桁を**ビット**(bit)、８桁をまとめて**バイト**(byte)という。
データはバイト単位で取り扱われる。

〇メモリについて



メモリは内部に多くの箱を持ち、それぞれ番号が振ってある（アドレス）。
アドレスを指定するとその箱の中にあるデータを読み書きできる。
プログラムを書く際には通常は意識しなくてもいい。
上図のようにメモリをビルのようにイメージすると、アドレスがビルの階、データがその階にある物といった感じになる。

③プログラミング言語とは

プロセッサ（CPU など）が理解できるのは２進数で書かれた「機械語」の命令のみで、それぞれの命令は極めて単純なものだけである。
これでは人間は操作できないので、機械語に１：１で単語を割り当てたものを**アセンブリ言語**という。
アセンブリ言語で書かれたソースコードを機械語に翻訳するためのプログラムを**アセンブラ**と呼ぶ。（ソースコードやアセンブリ言語それ自体もアセンブラと呼ぶこともある）
表示、条件分岐、演算など人間が扱いやすい単位まで機械語の処理をまとめ、扱いやすくしたものを**高級言語**という。（逆にアセンブリ言語や機械語を低級言語とも言う）

〇高級言語の実行方法
高級言語で書いたソースコードでも最終的には機械語に翻訳しなければコンピューターは理解できない。
この翻訳と実行の方法には言語によって違いがある。
・コンパイラ型言語
　　ソースコードを一度に機械語に変換（コンパイル）する。
　　Ｃ言語、Ｃ＋＋などはこれ。
・インタプリタ型言語
　　ソースコードをプログラムの実行時に１行ずつ解釈しながら実行する。
　　そのため、実行速度はコンパイラ型言語に劣る。
　　BASIC などはこっち。
※JAVA や C#は中間コードと呼ばれるものにソースコードを変換し、それぞれの仮想マシン上でプログラムを動作させる。

④Ｃ言語

１９７２年、アメリカのＡＴ＆Ｔベル研究所で誕生。
コンパイラ言語で、非常に高速に動作する。
現在使われている各種のプログラミング言語に対し、Ｃ言語の作法などが受け継がれている。
移植性の高い仕様のため、それ単体でのグラフィック操作などとは疎遠、Windows などの力を借りて実現する。
Java などの高級言語と比べると比較的低級（コンピューターに近い）記述が可能。
と同時にそれによってバグを起こしやすかったりもする。

⑤実際にコードを書き始める前に（開発環境の導入）

この講座では統合開発環境(Integrated Development Environment：IDE)として、
Microsoft 社の提供する **VisualStudio** を利用します。
現在では Express Edition という無償版があるので、これを使ってＣ言語のプログラムを書いていきます。
（ちなみに Microsoft DreamSpark というものもあります。詳細は割愛）
最新版は 2010 ですが、結構重いそうなのでＰＣの性能に自信がない人は前バージョンの 2008 を使うと良いかと。

以下では VisualStudio をインストール済みであるという前提で話を進めます。

○ソースファイルの作成
実際にプログラムを書く前に VisualStudio では「プロジェクトの作成」→「ソースファイルの追加」を行わなければなりません。

①[ファイル]→[新規作成]→[プロジェクト]でウィザードを開く。
②初級講座で扱うのはコンソールアプリケーションなので、
Win32 → [Win32 コンソールアプリケーション]を選択し、適当に名前をつけて[OK]
③Win32 アプリケーションウィザードが出たら「空のプロジェクト」にチェックを入れて完了。
④プロジェクトができたら[プロジェクト]→[新しい項目]→[コード]→[C++ファイル]でソースファイルを追加する。
⑤準備は完了。ようやくコードが書けます。

⑥実際にプログラムを書いてみる

まずはプログラミング言語の入門の最初によく用いられる有名なプログラムから。

```
#include <stdio.h>
int main(void)
{
        printf( "Hello World!¥n" );
        return 0;
}
```

書き終わったら Ctrl+F5(または[デバッグ]→[デバッグなしで開始])を押してみましょう。

結果：コンソール画面に”Hello World!”と表示される。

〇変数（整数型、浮動小数点型）と四則演算

変数とは値を入れられる箱のようなもので、C言語などの言語では使う前にその箱の大きさと名前をコンピューターに伝えないと使えないことになっています。

ここでは整数を扱う int 型と小数を扱う double 型、また、その演算について見ていきます。

```
#include <stdio.h>
int main(void)
{
        int x , y ;      /*計算用*/
        int wa , sa , seki , i_syo , amari ; /*和、差、積、商、余り*/
        double z;               /*計算用*/
        double d_syo;    /*小数での商*/

        x = 10;
        y = 3;
        z = 3.0;

        wa = x + y ;
        sa = x - y ;
        seki = x * y ;  /*x×y*/
        i_syo = x / y ; /*x÷y*/
        amari = x % y ; /*xをyで割った余り*/
        d_syo = x / z ;

        printf( "%d + %d = %d¥n" , x , y , wa );
        printf( "%d - %d = %d¥n" , x , y , sa );
        printf( "%d * %d = %d¥n" , x , y , seki );
        printf( "%d / %d = %d¥n" , x , y , i_syo );
        printf( "%d %% %d = %d¥n" , x , y , amari );
        printf( "%d / %lf = %lf¥n" , x , z , d_syo );
        return 0;
}
```

int 型の x と int 型の y との除算、int 型の x と double 型の z との除算の結果がポイントになります。

int 型は”整数しか扱えない”ため、小数点以下は切り捨てられた結果になります。

また、他の型との演算ではより表現できる範囲が大きい方に合わせられます。

この場合 int / double の結果は double（小数）になります。

C言語のコメントは本来 /* ～ */で書きます。

昔のC言語(C89)ではこれしか使えません。

現在(C99)は　//　のコメントも使えます。これはその部分から行末までをコメントとします。

これから先のコードでは　//　のコメントアウトも使っていきます。

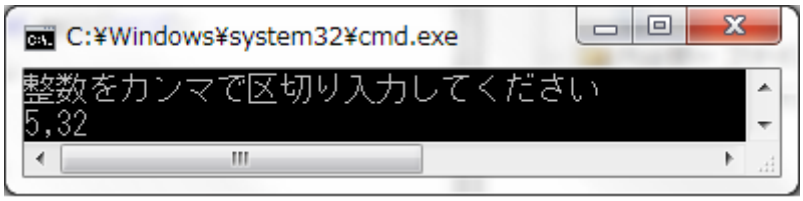
○キーボードからの入力の受け取り

```
#include <stdio.h>
int main(void)
{
        int num ;       //入力を受け取る変数

        printf( "整数を入力してください¥n" );
        scanf( "%d" , &num );    //整数を受け取る
        printf( "入力された値は%dです。¥n",num);
        return 0;
}
```

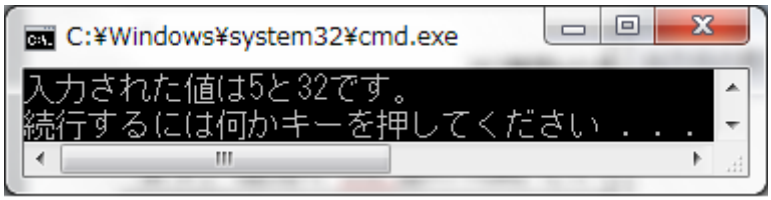
○複数の数値を一度に読み込みたいときの書き方

↑のプログラムを改変します.

```
#include <stdio.h>
int main(void)
{
        int num , num2;  //入力を受け取る変数

        printf( "整数をカンマで区切り入力してください¥n" );
        scanf( "%d,%d" , &num ,&num2 );   //整数を受け取る
        printf( "入力された値は%dと%dです。¥n", num , num2 );
        return 0;
}
```



こんな風に入力します. すると

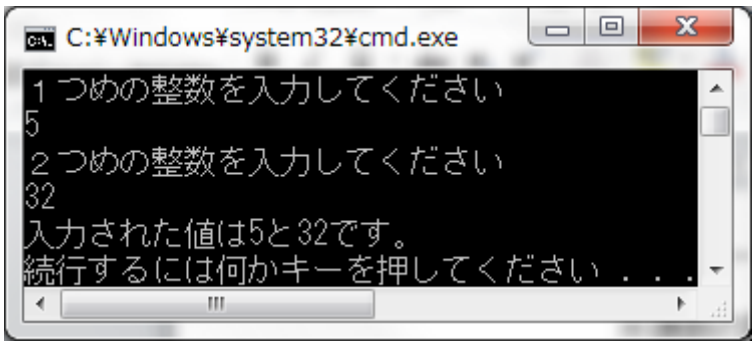


ちゃんとそれぞれに入力されているのがわかります.
しかし入力の度にカンマを打つのは面倒なので,次のプログラムのように scanf を複数書くのがオススメ.

・複数の scanf 関数を用いたプログラムの例

```
#include <stdio.h>
int main(void)
{
        int num , num2;  //入力を受け取る変数
        printf( "１つめの整数を入力してください¥n" );
        scanf( "%d" , &num );
        printf( "２つめの整数を入力してください¥n" );
        scanf( "%d" , &num2 );
        printf( "入力された値は%dと%dです。¥n",num , num2) ;
        return 0;
}
```

実行結果



〇まとめ

・変数
値を入れられる箱のようなもの。
int 型は整数を，double 型は小数を扱える。

・printf("表示したい文字列",%d などがあれば表示する変数);
→画面に「表示したい文字列」を表示する。%d で整数、%lf で小数を表示する。

・scanf("書式指定", &受け取る変数);
→書式指定には受け取る変数の型に対応するものを指定する。%d で整数、%lf で小数を受け取る。
　＆をつけ忘れないように。（この部分に関しては数回後に説明します）

演習

１．２つの整数の入力を受け取り，その数の和・差・積・商・割ったときの余り・それぞれの数の２乗を計算して画面に出力するプログラムを作りなさい．ただし答えのみを表示させるのではなく，どういう計算を行なっているかも表示させなさい．（ex. 5 + 32 = 38）