

[出発点のアイデア]

#include とか main とか、そういうのは初心者にとっては「よくわからないおまじない」でしかありません。いや、それが何を意味するのか、なぜ必要なかを説明してわからせることもできるでしょうが、結局は毎回同じことを書くだけであって、手間を増やしているだけにすぎません（初心者にとっては）。・・・そうであるならば、いっそ「main の中身だけ書けばよい」というモードがあったらどうでしょうか。#include に関しては標準関数は全部インクルードしておくことにします。もしこれではかえって不便ということになったら、それはもう初心者卒業だと思うので、このモードを脱して普通に C 言語のソースコードを書けばいいと思います。・・・そう思って作り始めたのですが、いつの間にか初心者以外にも便利な言語環境になってしまいました。

[easy-C でできること #1]

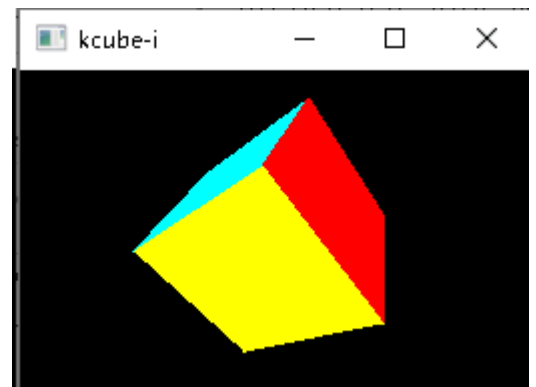
起動するとコンソール画面にプロンプト HL9> が出ます（これは初期の easy-C が HL-9 の改造版として作られたことに由来します）。それでこんなことができます。

```
HL9>printf("hello, world¥n");    ← C 言語でワンライナーできます。  
hello, world
```

```
HL9>j = 0; for (i = 1; i <= 10; i++) { j = j + i; } printf("sum=%d¥n", j);  
sum=55
```

ワンライナーで書けないような 2 行以上のプログラムは RUN コマンドで実行します。
（もちろん 1 行のプログラムも RUN コマンドで実行できます。）

```
HL9>run kcubei.c    ← 3D キューブが回ります
```



小学校低学年の C 言語入門のために、いくつかの命令だけは太文字でも書けます。かっこも略せます。
（アルファベットがわからないので、キーの刻印にない小文字は入力できません。かっこは面倒です。）

```
HL9>PLAY "CDEFEDCREFGAGFERCRCRCRCRC8C8D8D8E8E8F8F8EDCR"    ← かえるの歌が演奏されます
```

テキストファイルに C 言語で適当に関数を書いて保存して、そのファイル位置を設定ファイルに書いておくと、

```
char *myfunc(int a, int b) {                                (紙面の都合で無理やり 2 行で書いてすみません)  
    char *s = malloc(100); sprintf(s, "hello: a+b=%d¥n", a + b); return s; }
```

コマンドラインから簡単に利用できます。バッチファイル（スクリプトファイル）を書いている感覚です。

```
HL9>!puts(myfunc(2, 3));  
hello: a+b=5
```

[easy-C の仕組み]

HL-9 は小規模な C 言語インタプリタです。ワンライナー実行とテキストファイルを読み込んでの実行のどちらもできます。しかし未完成なので、浮動小数点演算や構造体や関数定義などがまだサポートされていません（将来のバージョンで改善予定）。これらの機能が必要な時や、厳密に gcc の仕様で実行したい時は、裏で gcc を呼び出して実行ファイルを生成し、それを実行する機能もあります。この方法は gcc との互換性はばっちりですが、実行開始までに少し時間がかかることと、激しくディスクアクセスすることが弱点です（乱用すると HDD や SSD の寿命に影響するかもしれません）。

HL9>!run kcube1.c ← このように!をつけると gcc でコンパイルして実行

[easy-C でできること #2]

（点数は使ってみて便利だと思った度合い）

[開発環境としてみたときの easy-C の特徴]	小計+3.9 点
#00 ワンライナーができる。	+1.0 点
#01 別ファイルに関数を書いておくと、それを組み込み関数のように使える。	+0.9 点
#02 プロンプトで\$や\$\$を使えばシェルコマンドが実行できる。	+1.0 点
#03 インタプリタ実行モードの他に gcc 実行モードがある。	+1.0 点
[言語拡張としてみたときの easy-C の特徴]	小計+2.2 点
#04 pr などの便利命令が使える。	+1.0 点
#05 簡易ループ演算子が使える。	+0.9 点
#06 AXFnc を使うと、関数の返値を複数受け取れる。	+0.3 点
[プログラミング入門としてみたときの easy-C の特徴]	小計+4.5 点
#07 入門に最適な PLAY 命令がある。	+1.0 点
#08 入門者が使いそうな命令については、大文字でも入力できる。PR や PLAY や IF など。	+1.0 点
#09 「NEW ファイル名」「\$\$ファイル名」によって、ファイルエクスプローラ無しで操作できる。	+1.0 点
#10 I~N (i~n) については、変数が int で宣言済みなので、宣言せずに使える。	+0.5 点
#11 グラフィック描画ライブラリが使える（簡単なゲームが作れるように）。	+1.0 点

現状は言語拡張系が弱いですが、以下の改良を予定しています。

[言語拡張としてみたときの easy-C の特徴]（追加予定）	小計+2.1 点
#12 インタプリタモードの時だけ、演算子の優先順位を変更できる。	+0.3 点
#13 gcc モードでの #longdef, #endlongdef, #longuse	+0.8 点
#14 Uniform Function Call Syntax に対応。	+1.0 点

[ダウンロード・より詳しい説明]

https://essen.osask.jp/?a23_ec001

[開発プロジェクトの位置付け]

（ここに書いた以外にも多数のサブプロジェクトがあります）

OSASK 計画[1997 年～]（超要約：OS から作り直せば、面白いことができるんじゃないか?!・・・ちょっと中二病的?）

→ Essen[2016 年～]: プログラミング言語開発サブプロジェクト（言語処理系で OSASK の理想を追求できるのでは?）

→ easy-C[2023 年～]: 当プロジェクト（OSASK にまず必要なのは新言語ではなく C 言語の改良版なのでは?）

[小学校低学年にプログラミング入門]

C 言語でプログラミング入門といえば、まずは hello, world です。しかしたった一行の表示のために、ファイルを作って何行も書いてコンパイルしなければいけないのだとしたら、たとえエラーなく動いたとしても、それは「えー、そんなの全然うれしくないじゃん」って思うのが普通でしょう。easy-C ならそれくらいはワンライナーでできます。

```
HL9>printf("hello, world\n");  
hello, world
```

いやいや、これでもまだうれしさはありません。hello と書きたいなら hello を 5 文字だけ入力すればいいのです。printf とか書き足すだけ面倒というものです。

じゃあ fillRect や drawLine などをお絵かきをするというのはどうでしょうか？・・・もちろん悪くないのですが、学校やお友達からペイントの使い方を教えられてしまったら、もうやってくれなくなります(泣)。そこで！ play 文なのです。play 文は MML を演奏するだけの機能で、昔の BASIC では普通にできたことです。

```
HL9>play "cdefgabo5c"; ← ドレミファソラシド
```

これだけで音が鳴ります。これは小学生にウケます！画面にテキストを表示するのとはわけが違います。あーでも待ってください。キーボードは大文字が刻印されています。小学 1 年生とかですと、アルファベットの小文字なんて想像もできません。それじゃあ入力できないですね。だから CapsLock してあげてください。

```
HL9>PLAY "CDEFGABO5C";
```

これでちゃんと同じように鳴ります。easy-C では入門向けの命令は、みんな大文字でも書けるようにしてあるのです。かっこも略せます。

printf で%d を使う頻度が多いので、pr という関数を用意しました。もちろん大文字でも OK です。

```
HL9>pr 1+2*3  
7
```

もともとは入門用に作った命令でしたが、かっこ・変数・数学関数などが自由に使えるうえに、エラーになってもコマンドヒストリから修正して再実行できるので、もはや電卓アプリよりも便利で、大人のほうが多用します(笑)。%f で表示する prf という関数もあります。

[Q&A]

[Q]音楽演奏がしたいなら、C 言語ではなく専用ツールのほうがいいのでは？

[A]音楽演奏だけをやりたいなら完全にその通りです。でも私が小学生にやらせたいのはプログラミングなのです。その入り口として PLAY 文がいいと思っていますのです。ですからここからループを教えて 5 回リピートしたり、演奏しながら絵をかいたり、そういうことにつなげていきたいです。

[Q]プログラミングを教えたいのなら、もはや標準になりつつある Scratch のほうがいいのでは？

[A]それはそうなのですが・・・私はどうもビジュアルなプログラミング言語が好きになれなくて、やっぱり子供にもテキストベースの言語を教えたいです。いろいろと試した結果、変数の概念さえ強要しなければ、低学年でも結構ついてこられることがわかりました。・・・小学校低学年が、テキストエディタを操作してプログラミングをやり、エラーが出たら自力で直す様子は、結構感動的ですよ！・・・Scratch については、きっと学校で習うだろうと思うので、そこで存分にやってもらえればいいかなと思います。

[簡易ループ演算子]

適当な配列があったとして、全要素を 0 に初期化したいことがあります。この時は `memset` が使えます。しかし全要素を 1 にしたい時は、`char` 型でない限り `memset` は使えません。仕方なく、`for` 文でやることになります。もしここで、

```
a[0~99] = 1;
```

ってできたら、きっと楽だろうなあと前から思っていました。これを実現するのが `easy-C` の簡易ループ演算子です。以下のように書けば、勝手に `for` を補って実行します。

```
a[0:<=99] = 1;      → for (i = 0; i <= 99; i++) { a[i] = 1; } に読み替えられる
```

`a[0:<100]=1;` という書き方も可能で、どちらも簡易ループ演算子です。・・・これを使うとこんなこともできます。

```
HL9>i = 0; i += 1:<=10; pr i
55
HL9>pr 0:<10
0123456789
```

どちらにせよ、今までできなかった何かができるようになるようなものではなく、単によくある処理をちょっと短く書けるようになっただけです。でも重宝しています。・・・簡易ループ演算子には別の記法もあります。

```
HL9>[i = 1:<=9] { pr i * i; }    ← pr i は整数とスペースを print する関数
1 4 9 16 25 36 49 64 81
```

これはループカウンタと繰り返し範囲を明示できる記法です。もうほぼ `for` 文と同じなのですが、ちょっとだけ短くなるので、これもまあ使っています。構文を角カッコにしたのはなんとなくです。あまり考えていません。

さらにこんな書き方もできます。

```
HL9>[5] { prs "hello "; }      ← prs は文字列を print する関数
hello hello hello hello hello
```

これはもはや演算子がないので簡易ループ演算子とは言えないのですが、ループカウンタが不要で、とにかく指定した回数だけ繰り返しさえいい場合は、このように書けるのです。

簡易ループ演算子は `for` の完全な代用ではなく、`for` にしかできない処理はたくさんあります。それは `for` を使って書きます。

[AXFnc]

C 言語で複数の計算結果を返したい時は、ポインタを使うのが一般的です。たとえば、2023 年 1 月 100 日を渡すと、それを 4 月 10 日に直してくれる `fixDate` という関数があったとします。

```
HL9>!fixDate(2023, 1, 100, &i, &j, &k); printf( "%d/%d/%d\n", i, j, k);      ← gcc モード
2023/4/10
```

私はこれが不満でした。返値が 1 つならポインタなんかいらなのに、なぜ 2 個以上になったらポインタにしなければいけないのか。・・・ということで、こんなマクロを作りました。

```
HL9>!AXFnc_i3(i, j, k, =, fixDate(2023, 1, 100)); printf( "%d/%d/%d\n", i, j, k);
2023/4/10
```

まだ十分に見やすい形になったとは言い難いのですが、心意気だけでもご理解いただけたら幸いです・・・。

なお、これは下記の記法で上記に読み替えられるというわけではなく、`fixDate` の中身も `AXFnc` 用書き換える必要があります。