

無料 eBook サンプルペーパー

つべこべ言わず組んでみよう

～体で覚えるC/C++プログラミング～

プログラムを組める環境があるなら、
とにかくプログラムを組むことこそ重要である！
……ってことで、このペーパーを片手に
C/C++言語のプログラムに挑戦してみよう！

このペーパーについて

このペーパーはコピーフリーです。あなたの知り合いに C/C++言語を学びたいというプログラマー志望の方などがおりましたら、このペーパーをコピーして渡していただいても構いません。

更に、ホームページやブログをお持ちの方は、あなたのサイト等での二次配布等も全面的に許可致します。むしろ二次配布して頂けるなら是非してくださいと嬉しいです。

事前、事後の報告も必要はありませんが、報告くださったらとっても喜びます(笑)

ただし、著作権等は放棄しておりませんので、著作情報等の内容を書き換えての配布や自分が書いた文章と称して販売したりするのは止めてください。

そして、このペーパーにより、いかなる損害が発生したとしても、僕はいかなる責任も負わない事とします。あくまでも自己責任にてご利用下さい。

ちなみに、このペーパーの内容は以前僕が発行したメールマガジンの内容をまとめたものです。僕のメールマガジンを読んだことのある人なら、一度見たことのある内容かもしれませんのでご了承くださいませ。

目次

はじめに	4
C/C++のプログラムにて必要なもの	5
誰でも出来るプロジェクト構成 (DOS 編)	6
1、プロジェクトの新規作成	7
2、プロジェクトの選択	8
3、ディレクトリの選択	9
4、プロジェクト名の入力	10
5、空のプロジェクト作成	11
6、新規プロジェクト情報の確認	12
7、ワークスペース	13
8、ソースファイルの新規作成	14
9、ソースファイル名の入力	15
10、プログラムの準備完了	16
初めてのプログラミング (DOS 編)	17
0、よくあるコンパイルエラー (笑)	18
1、最も短いプログラム (笑)	19
2、コメントは目に優しい (笑)	20
3、はろ～、わ～るど (笑)	21
4、見えねえよ！ ってお方に (笑)	22
5、なんか変な世界 (笑)	23
6、改行コード	24
7、変数	25
8、繰り返しの for 文	29
9、インデント	32
10、条件分岐の if 文	33
終わりに	35

はじめに

このペーパーを読む気になってくださり、ありがとうございます。

C/C++言語の参考書は世の中に腐るほど出回っていることでしょうか、僕が思うに参考書というものは書いてあることが固く苦しくて、ずぶの初心者達にはとっつき難いものに見えてなりません。

確かにプログラムをある程度組める僕とかが読めば、自分の知識と照らし合わせて更に新しい知識を得ることが出来ますが、前知識無し状態で参考書と睨めっこしても、なかなか知識を読み取るのは困難なことです。

僕はこの前知識と言えるものを、専門学校の授業として教わりました。

プログラミングの講師より「こうしてあーしてこうすると、プログラムが組めるよ」と聞き、最初の段階である「Hello world」を組みました。

プログラミング知識が0から1になった瞬間です。

その後は参考書の知識もすんなり頭に入ってくるようになり、色々なプログラムを黙々と組んでC/C++言語を習得しました。つまりはかけ算と似ていて0にどんな数値をかけ算しても0ですが、1にかけ算すればかけた数値になります。更に知識を増やして2や3の知識で参考書を読めば、自分の知識は跳ね上がります。

プログラム習得で最初の方でつまづく人は、その0から1になれていないことが多いと僕は思います。何故なら難解な参考書を開いただけであまりの情報量に頭が痛くなり、サンプルプログラムを組んでみようという気が無くなるからです。

しかも、無駄とも思える内容のサンプルプログラムを延々と組ませ、必要知識を得るためには、何百ページもの情報を読んでいくことになります。気が遠くなる話です。

でも、プログラムを組める僕から言わせてもらえば、そんなことをせずとも良いです。

分厚いプログラミング言語の参考書を何百ページも暗記する必要なく、C/C++言語のプログラミングの基礎を学ぶ方法を、このペーパーにて提供致します。

C/C++のプログラムにて必要なもの

正直に言うと、この「必要なものを揃える」のが C/C++言語の習得にて一番大変な部分だと思います。

でも自分の為に投資する覚悟があるなら、この問題は解決できたも同然です。

最寄の電気屋さん等で「Microsoft Visual C++」等のソフトをお買い上げ下さい。

おそらく高額なので、買うのは困難かもしれませんが、開発環境が無ければプログラムは組めませんし、買う以外で開発環境を整えるのにはそれなりの知識が必要になるかもしれませんので覚悟してください。

ちなみに、僕が C/C++言語でのプログラムを組むときに使用している開発環境（ソフト）は「Microsoft Visual C++ 6.0」です。かなり古いものですが、問題なく愛用しています。

マイクロソフトのホームページにて、「Microsoft Visual C++」の無料バージョンのものをダウンロードするという方法もあります。

Visual Studio 2008 Express Edition
<http://www.microsoft.com/japan/msdn/vstudio/express/default.aspx>

こちらはいつまで無料で提供されているのか等は不明なので、リンク切れしていたり、無料公開の「Microsoft Visual C++」がダウンロード出来なかった際の文句は、一切受け付けません（……ってか、そういった質問やクレームは勘弁して下さい）

あと、プログラムを組んでいる画面のスクリーンショット図面を所々に組み込んで解説していきますが、僕が使用している「Microsoft Visual C++ 6.0」は、かなり古いものですので、最新の「Microsoft Visual C++」とはデザインが違っていたり、プロジェクト構成等の設定方法が若干違っているかもしれませんが、脳内補完で補ってください。

誰でも出来るプロジェクト構成 (DOS 編)

僕は常日頃から思っていることは、「パソコン初心者にとってプログラムのプロジェクト構成はかなり難しい」ってことです。

僕は「Microsoft Visual C++ 6.0」のコンパイラ（統合開発環境）を持っており、それで大抵のプログラムを組んでおりますが、コレがパソコン初心者にとっては余計な機能が多くて非常にとっつき難いものになっています（苦笑）

一応、パソコンに関してもプログラムに関しても上級者に位置する僕からすれば、いらぬ機能は無視して、必要な機能を色々応用して使いこなせていますが、ずぶの初心者にしてみたら「Hello world」を組む前に、「どうやったらプログラムが組めるのですか？」っていうふうに根本的な部分でつまづいてしまう事でしょう。

そこでそんな「ずぶの初心者」達のために、最も簡単な C/C++言語のプログラムが組める DOS（コンソール）プログラムのプロジェクトの作り方を親切に図面付きで解説します。

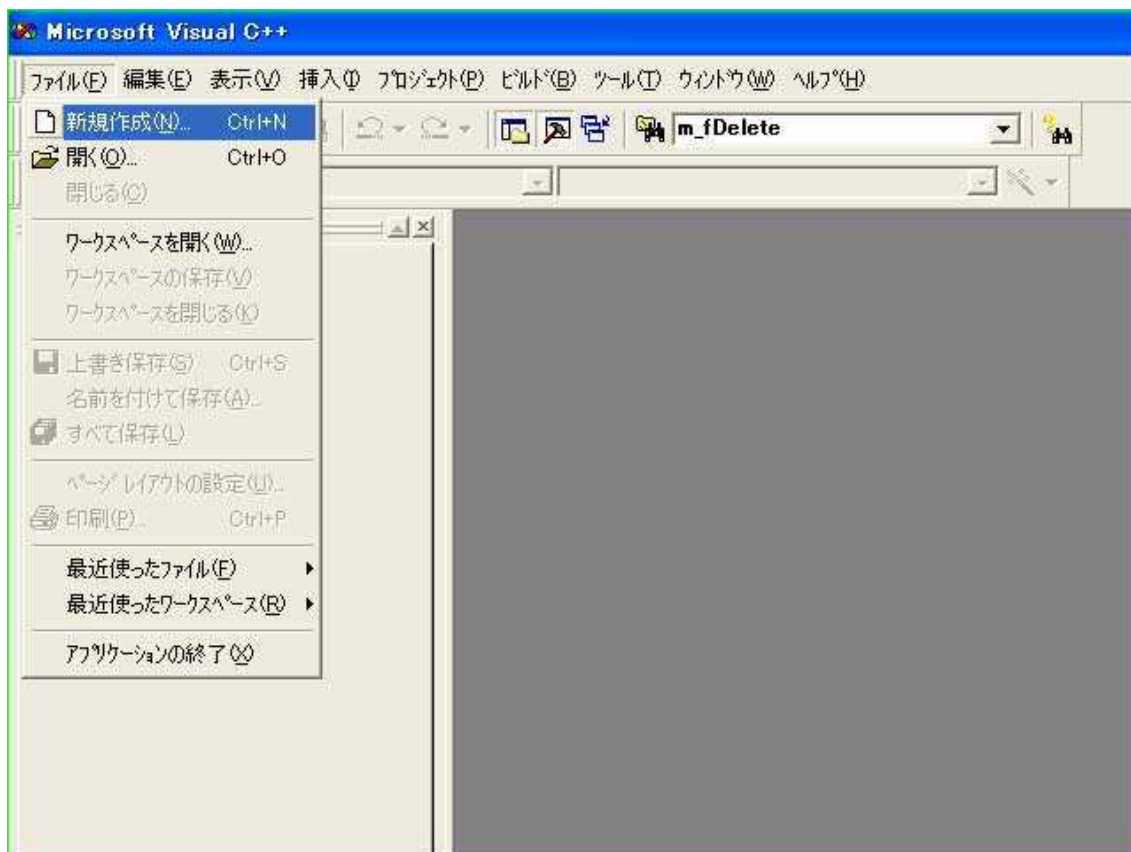
僕が持っている環境が「Microsoft Visual C++ 6.0」のみなので、他の環境だと色々応用して対処しなければなりません、大抵は似たり寄ったりでしょう。多分……。

ひとまず最も簡単な環境で、最も簡単なプログラムを組んでもらうために、コンソールプログラム（DOS）のプロジェクト構成のやり方をお教えします。

「バカにしているのか？」って思えるほど一つ一つを細かく分けて解説していますので、やる気さえあれば誰でもプロジェクト構成が出来ることでしょう（笑）

それでは避けては通れない最初の難関、プロジェクト構成に挑戦してみてください。

1、プロジェクトの新規作成



「Microsoft Visual C++」を立ち上げ、メニューバーの「ファイル(F)」から「新規作成(N)」を選択します。

2、プロジェクトの選択



「新規作成」のウィンドウが出たら「プロジェクト」タブであることを確認して、「Win32 Console Application」を選択します。

ちなみにまだ「OK」は押せません。

3、ディレクトリの選択



「位置(C)」のテキストボックスの右側にあるボタンを押すと、プロジェクトを作成するディレクトリを選択できます。

ここでデスクトップ等、わかりやすい位置に設定します。

なお、この例ではわかりやすいようにCドライブの直下の位置を選択しています。

ディレクトリの意味がよくわからないという方は図と同じ箇所(Cドライブの直下)を選択して「OK」を押しましょう。

4、プロジェクト名の入力



「プロジェクト名(N)」を入力します。この例では「test」と入力しています。

プロジェクト名を入力すると、「位置(C)」のテキストボックスの行末にもプロジェクト名が書かれます。

確認したら、いつの間にか押せるようになっている「OK」を押します。

5、空のプロジェクト作成



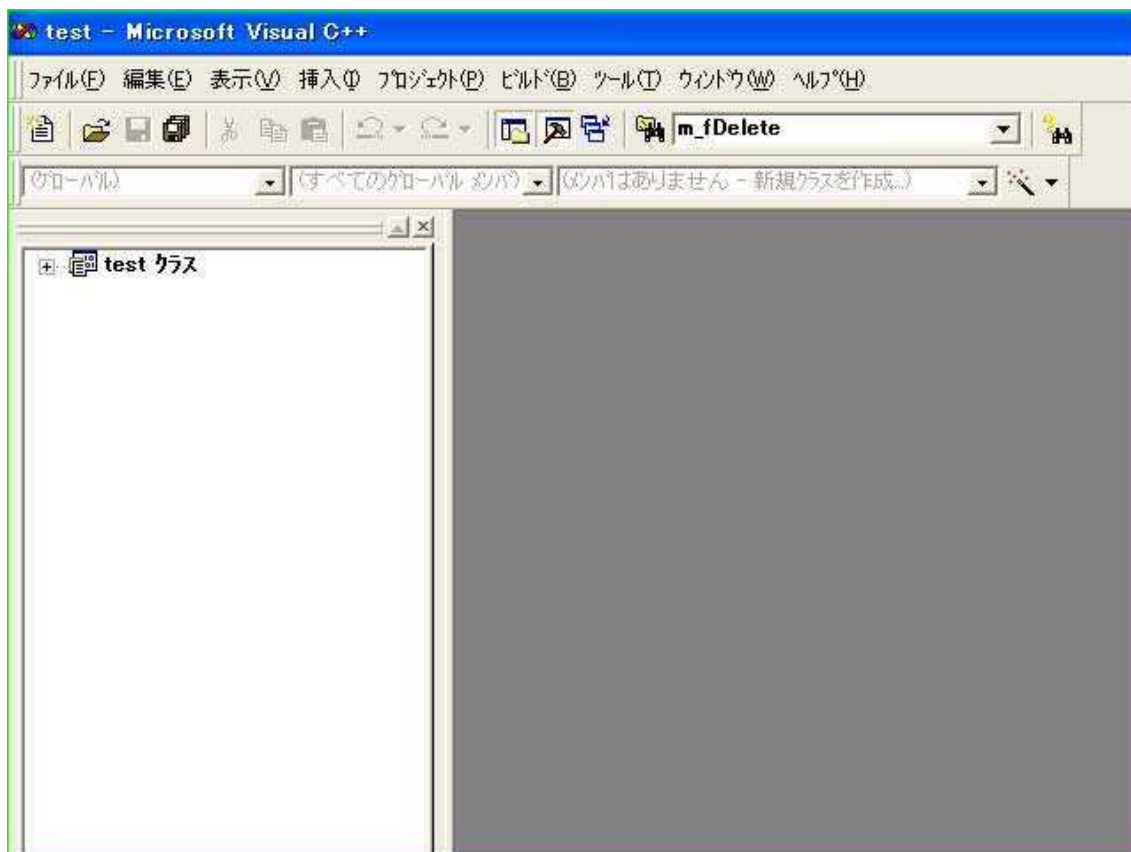
「空のプロジェクト(E)」を選択して「終了(F)」を押します。

6、新規プロジェクト情報の確認



「新規プロジェクト情報」を確認したら「OK」を押します。

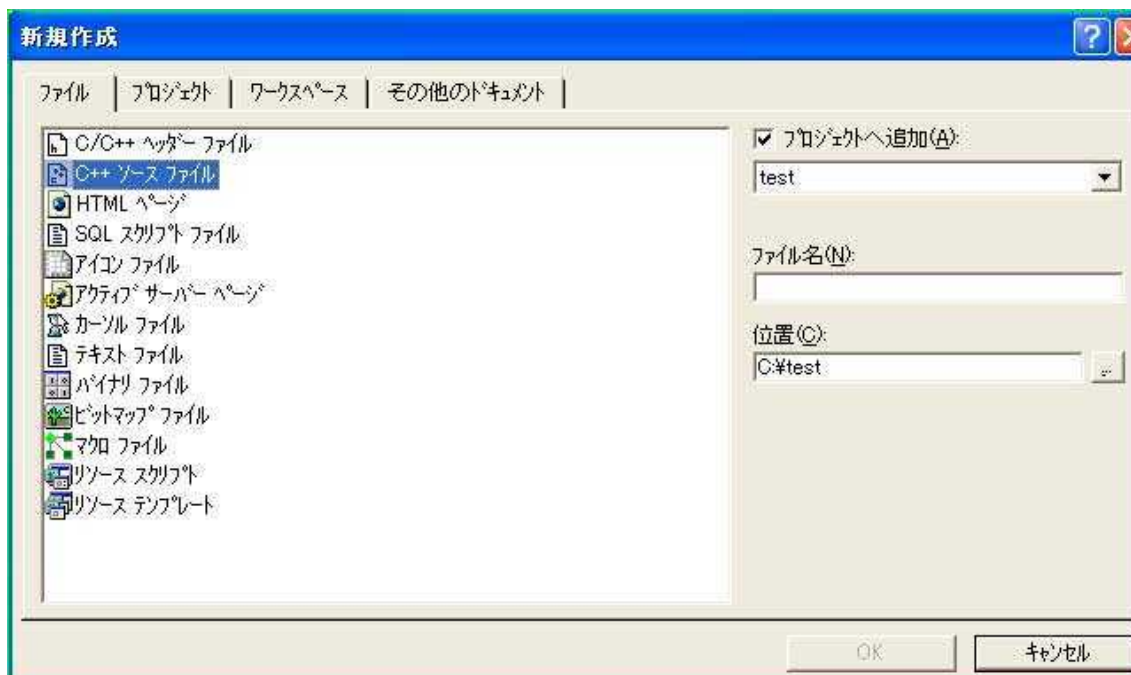
7、ワークスペース



画面の左側に「test クラス」と書かれたウィンドウが出来あがっています。

それが「ワークスペース」です。

8、ソースファイルの新規作成

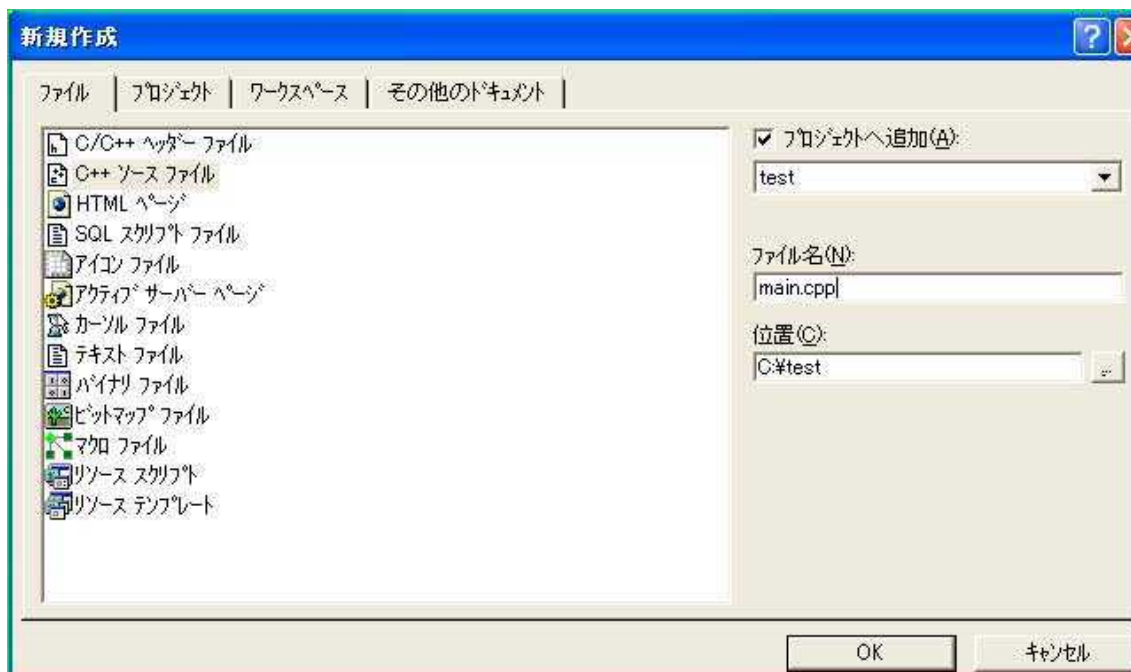


再度、メニューバーの「ファイル(F)」から「新規作成(N)」を選択します。

「新規作成」のウィンドウが出たら「ファイル」タブであることを確認して「C++ ソースファイル」を選択します。

ちなみにまだ「OK」は押せません。

9、ソースファイル名の入力



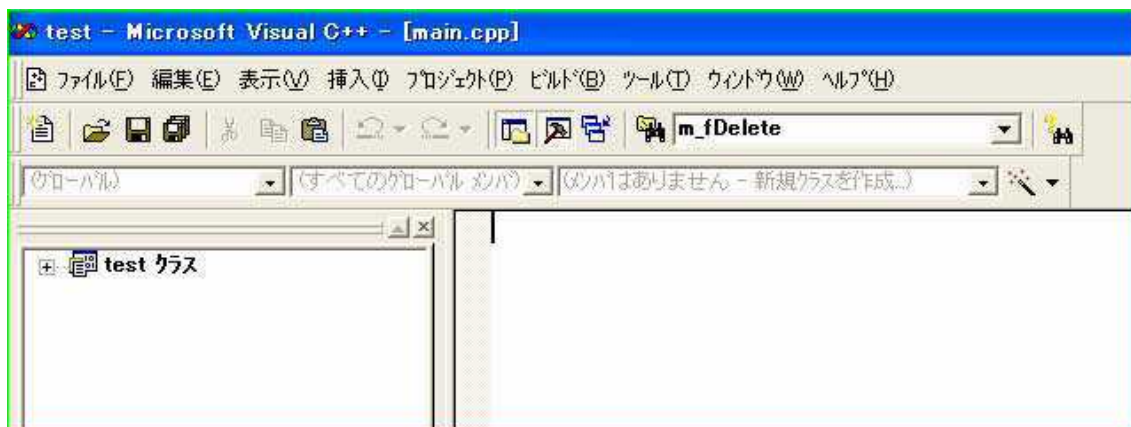
「ファイル名(N)」のテキストボックスに「main.cpp」と入力します。

この「*.cpp」という拡張子は「C(シー)++(プラスプラス)」の略であり「C++言語」のソースファイルだという事になります。

別に「main.c」として「C言語」としてしまっても良いのですが、C++言語の方が便利機能もありますし、C言語の機能もほぼ完全に網羅しているので問題は無いでしょう。

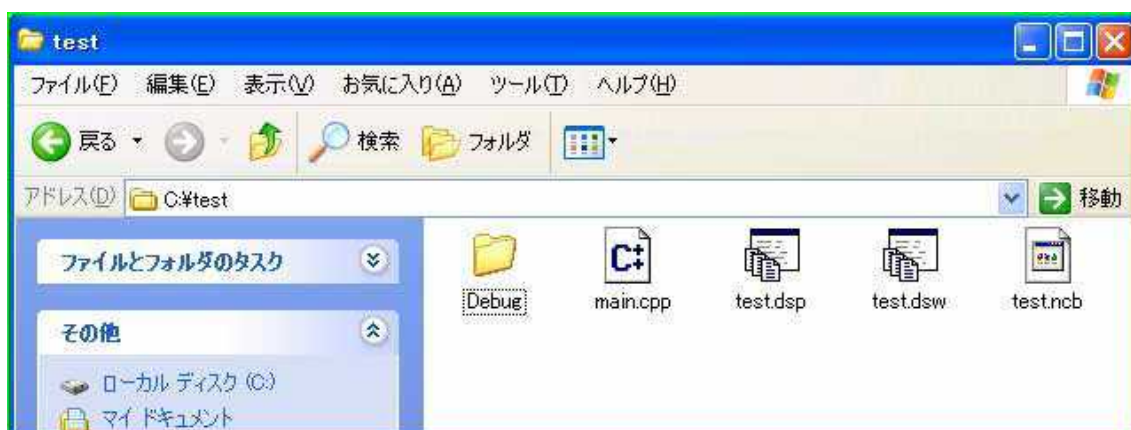
ファイル名を入力したら「OK」を押しましょう。

10、プログラムの準備完了



右側にテキストエディット画面が出来上がっていればプロジェクト完成です。そのままプログラムを組むことが出来ます。この例と全く同じ手順で作成した人は、マイコンピュータを開いてCドライブの直下を確認してみると「test」というフォルダがあるはずです。

それを開くと、下のようなファイル構成になっているはずです。プロジェクトを作るのが面倒くさいという人は、この「test」フォルダをテンプレートとしてコピーしておくとう便利です。このプロジェクトを一度閉じた後、再度開きたい場合は、「test.dsw」をダブルクリックしてください。



初めてのプログラミング（DOS 編）

これより本格的に C/C++ 言語に関して解説していきたいと思います。

つまり、ここからが「つべこべ言わず組んでみよう」の本題とも言えます！

サンプルプログラムを 100 回読むよりも、1 回でも実際に自分で打ち込んでみて、どのように動作をするのかを体感してみるほうが効果的です。

そういうわけで「とにかく組む」ことに重点を置いて、サンプルプログラムのソースだけいくつか用意してみました。解説に関しては大まかにしかしていないので、僕の解説で大まかな概要をなんとなく覚えて、それから参考書等で正しい知識を身につけましょう。

参考書などの固っ苦しい解説でも、前知識がある程度あれば意外とすんなり頭に入ってくれる場合があります。いわゆる階段の原理と同じで、一階から二階にジャンプして登るのは、常人には厳しいものがありますが、階段を一步一步着実に登ることなら、誰でもできますよね？

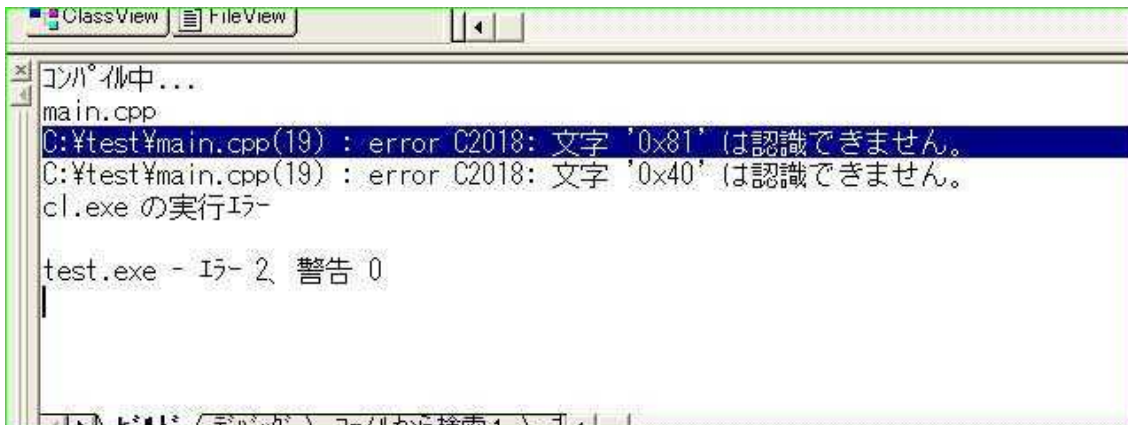
そんな階段の一段のように簡単なところから体に叩きこめるサンプルプログラムを
図面つきで大雑把に説明していきます。

もしプログラムをしていてわからないことがあったら、とりあえず後回しにして「つべこべ言わず」サンプルプログラムを組むか、Google 等で検索してください。

このペーパーはただ読むだけなら何の価値もありません。だから無料で配っているのですが、このペーパーがキッカケでプログラムを組む気になり、これより下に載っているサンプルプログラムを何度も組んでみて、実際に実行してもらえらる事こそ僕の望みです。

とにかく行動あるのみです！ プログラムを組む前から頭で理解しようとしても時間がかかるだけですから、そんな暇があったら一回でも多くプログラムを組み「体に刻み込むように」覚えていってください。

0、よくあるコンパイルエラー（笑）



```
コンパイル中...
main.cpp
C:\¥test¥main.cpp(19) : error C2018: 文字 '0x81' は認識できません。
C:\¥test¥main.cpp(19) : error C2018: 文字 '0x40' は認識できません。
cl.exe の実行エラー

test.exe - エラー 2、警告 0
```

全くの初心者の方に、プログラムを組む時の注意点を一つだけ伝えておきます。

error C2018: 文字 '0x81' は認識できません。

error C2018: 文字 '0x40' は認識できません。

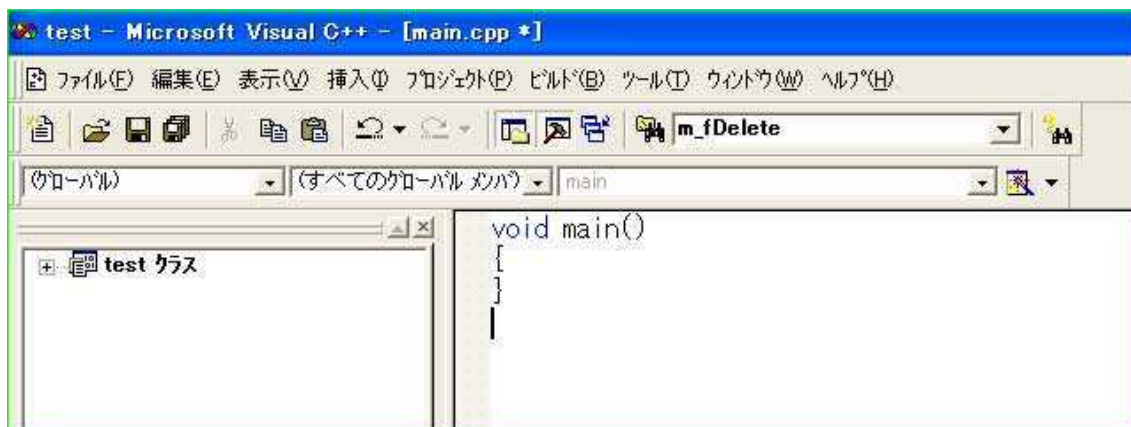
ソースを見た限りおかしいところはないのに、上記二つのコンパイルエラーが出た場合、プログラムのどこかに「全角スペース」を使ってしまっています。

C/C++言語では原則的に「全角文字」は使用できません。

特に「全角スペース」をどこかに入れてしまうと、見た目はどこもおかしくないのに何故かエラーが出るという不可解な状況に陥ります。

もしも上記のコンパイルエラーが出た場合、エディタの検索機能で「全角スペース」を検索し消すか、「Tab」もしくは「半角スペース」に置き換える等の対処をしてください。

1、最も短いプログラム（笑）



たったこれだけです。このプログラムが組めないという人は、パソコンの入力自体がまともに出来ないって事になります。

「void」って字が青くなっていない人は、おそらくスペルがミスっています。そして「mein」ではなく「main」です（笑）

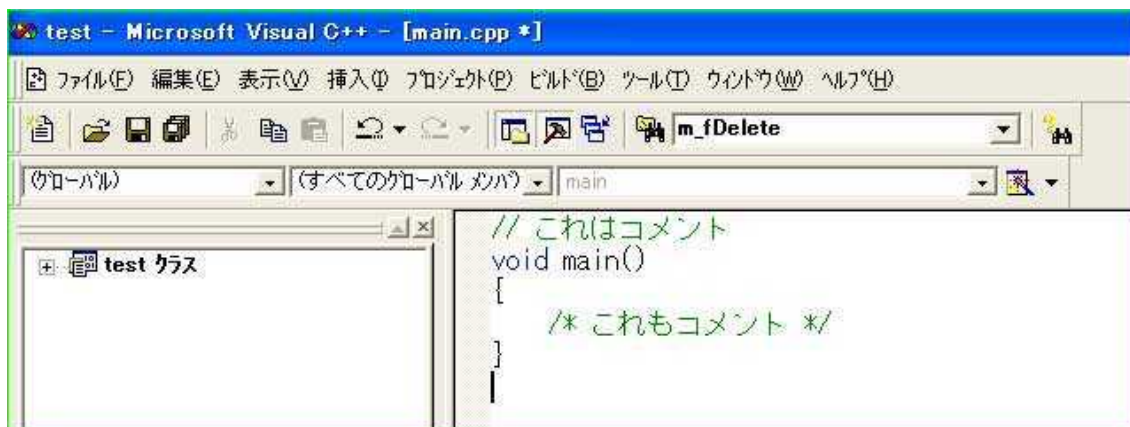
とりあえずコンパイルしてください。キーボードの「F7」でコンパイルです。

エラーが出なければ、まあ成功でしょう。実行はキーボードの「F5」で可能ですが、一瞬黒い画面がパッと現れてパッと消えるだけです。

このプログラムではC/C++言語で必ず必要な、処理の開始位置である「メイン関数」を宣言しています。.....それだけで他の処理は一切何もしていません（苦笑）

だから言ったでしょ。「最も短いプログラム」だって（微笑）

2、コメントは目に優しい(笑)



緑色の文字で「// これはコメント」「/* これもコメント */」と追加されました。なんで緑色なのかというと、それはコメントだからです(微笑)

この緑色のコメント部分では、先ほど使用できないと書いた「全角文字」も使うことができます。勿論「半角文字」も使えますし「全角スペース」もどうぞ自由に。なぜなら、この緑色のコメント部分に何を書いたとしてもプログラムには反映されないからです。

なぜこんな機能があるかといいますと「目に優しいプログラム」にするためです(笑)

コメントはプログラムの説明書きとして使う機能です。

プログラムを組んでいると、所々に難解なアルゴリズムを組んでしまうことがあります。そのアルゴリズムがどんな処理をしているのかを端的にコメントで説明書きを残しておくと、後々手直ししたり、ソースを読み直したりするときに大変便利です。

人にはそれぞれ限界がありますから、計算式を見ただけでそれがどんな動作をするか見分けるのにはそれなりに時間を要します。でもコメントをつけておけば、その難解な計算式の意味が一瞬でわかります。つまりソースをガン見しなくてもいいので目に優しいです。

もし、あなたが難解な計算式を一目見ただけでどんな処理をしているのか理解できる人ならコメントをつけると強制はしませんが、そうでないならコメントをつける事を強くオススメします。複数人でプログラムを組んでいるのならコメントは必須です。

3、はろ～、わ～るど(笑)

```
//  
// テストプロジェクト (main.cpp)  
//  
//  
// //////////////////////////////////////  
// インクルードファイル  
#include <stdio.h> // 標準入出力  
//  
// //////////////////////////////////////  
//  
// メイン関数  
//  
// ※C/C++言語のプログラムで必ず必要な、処理の開始位置です。  
//  
void main()  
{  
    printf( "Hello world" ); // 画面に「Hello world」と表示  
}
```

いきなり行数が増えたからといって、驚かないで下さい。ほとんどがコメントですので。実際に増えた処理は「#include <stdio.h>」と「printf("Hello world");」だけです。

インクルードを解説するのは少々面倒なので興味のある方は自力で調べてください。

そして、なんとなく見てわかりそうなのは「プリントエフ」の方ですね。ダブルクォーテーション(")で囲われている文字列を画面に表示します。ほんの一瞬だけ.....。

文末のセミコロン(;)は日本語で言う「。」と同じ意味合いで、処理の区切りを意味しますので、必ずつけて下さい。プログラム言語は書式にうるさいので、几帳面にしっかりとつけて下さい。

ここで「ん？ じゃあなんでインクルードの方には、セミコロンが付いてないの？」って思った人はカンが鋭いです。それはインクルードの書式の方が特殊なのです。...
...なので疑問に思った方は、自力で調べてスッキリ理解してください(笑顔)

4、見えねえよ！ ってお方に（笑）

```
//  
// テストプロジェクト (main.cpp)  
//  
////////////////////////////////////  
// インクルードファイル  
#include <stdio.h> // 標準入出力  
////////////////////////////////////  
//  
// メイン関数  
//  
// ※C/C++言語のプログラムで必ず必要な、処理の開始位置です。  
//  
void main()  
{  
    printf( "Hello world" ); // 画面に「Hello world」と表示  
    getchar();  
}
```

僕も動体視力が弱いので、処理が速すぎて「Hello world」の文字が見えません。なので「getchar();」という処理を「プリントエフ」の下の方に追加しました。

すると、どうでしょう……。 「F7」でコンパイルし、「F5」を押してプログラムを動かしてみると、黒いコンソール画面に「Hello world」と、白い文字がしっかり出ています。「Enter」キーを押せば、黒いコンソール画面が消え、プログラムも終了します。

これで色々テストしやすくなりましたね（やったー）

ためしに「プリントエフ」の「ダブルクォーテーション」で囲われた部分の文字を書き換えてみてください。ちなみにその部分にも例外的に「全角文字」を使うことができます。

5、なんか変な世界（笑）



「ゲットキャラ」を追加して処理が見えるようになったとき、素直に「Enter」キーを押した人は気が付かなかったと思われませんが、適当なアルファベットを入力すると……とても変なことに、黒い画面に文字を入力できてしまいます（苦笑）

せっかくの「Hello world」が変になってしまった瞬間です！

これは「ゲットキャラ」が「画面が見えるようにする処理」をする機能（関数）ではなくて、文字入力を受け付ける機能だからです。文字を受け付けている間は処理が止まっているから、出力結果である「Hello world」が見れます。そして「Enter」キーを押せば入力受付を終了し「ゲットキャラ」はその結果を出すんですが、それを受け取ってないし、その後に何の処理も無いため、プログラムが終了し画面が消えてしまうということです。

……ってことで、あんまり気にしないで下さい。どうしても気になるなら自分で検索して調べてみるのもアリですけどね。

6、改行コード

```
void main()
{
    printf( "Hello world" );      // 画面に「Hello world」と表示
    getchar();
}
```



```
void main()
{
    printf( "Hello world\n" );   // 画面に「Hello world」と表示
    getchar();
}
```



「ゲットキャラ」の変な仕様がわかったところで、ついでだからその機能をありがたく利用させてもらい、「改行コード」の説明をすることにします。

上のプログラムと下のプログラムそれぞれにソースと処理結果を載せましたが違いがわかりますか？ ちなみに処理結果の方の「Hello world」の後に続く文は僕が適当に打ち込んだものなので、過度な期待はしないで下さい。

ソースの違いは「ダブルクォーテーション」で囲まれた「Hello world」のところで、下のソースには「\n」が追加されています。これが「改行コード」です。

この記号が追加された下の方の処理結果では、僕が適当に打ち込んだ文の部分が改行されて表示されていることがわかんと思います。

7、変数

サンプルソース

```
void main()
{
    int data = 10; // 変数dataを宣言し10を代入し初期化

    printf( "%d ¥n", data ); // 変数dataに格納されている数値を表示して改行

    data = 5; // 変数dataに5を代入
    printf( "%d ¥n", data ); // 変数dataに格納されている数値を表示して改行

    data++; // 変数dataをインクリメント (数値を1増やす)
    printf( "%d ¥n", data ); // 変数dataに格納されている数値を表示して改行

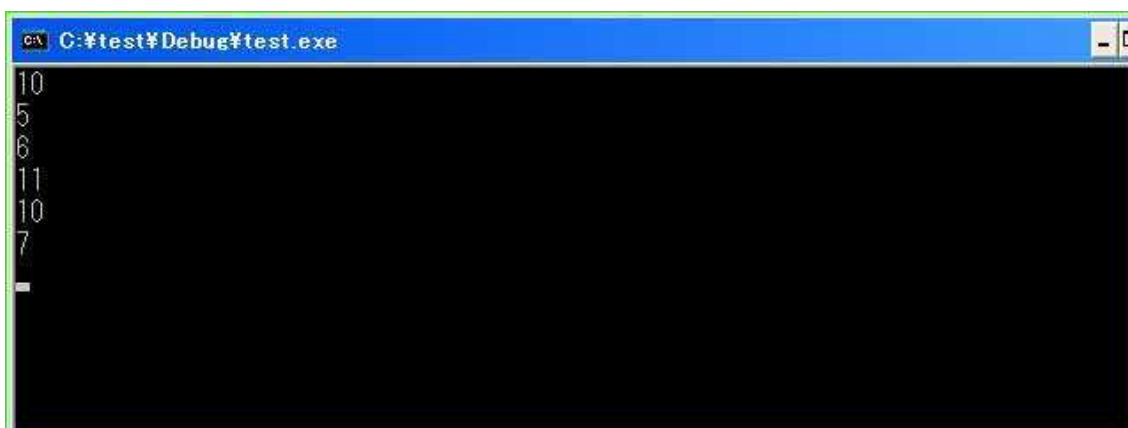
    data += 5; // 変数dataの数値を5増やす
    printf( "%d ¥n", data ); // 変数dataに格納されている数値を表示して改行

    data--; // 変数dataをデクリメント (数値を1減らす)
    printf( "%d ¥n", data ); // 変数dataに格納されている数値を表示して改行

    data -= 3; // 変数dataの数値を3減らす
    printf( "%d ¥n", data ); // 変数dataに格納されている数値を表示して改行

    getch();
}
```

処理結果



```
C:\¥test¥Debug¥test.exe
10
5
6
11
10
7
-
```

いきなり処理の行数が増えたからといって、驚かないで下さい。
よく見ると無駄に沢山ある「プリントエフ」の部分は、変な書き方に変わっているけど全部同じです。だって全部コピペしただけなので～（苦笑）

そして「メイン関数」の一番最初に「int data = 10;」って書いていますが、コメントによると「変数 data を宣言し 10 を代入し初期化」となっています。

変数というのは、よく「箱」にたとえられます。

まず、その箱がどんな形なのか決めて、名前を付けてあげます。
上記の変数宣言例では、この形（型）を「int」とし、名前を「data」としています。この名前の部分は半角英数文字なら自由につけられますが、最初の一文字には数字を使えません。

・変数名の大丈夫な例

data
Data（大文字も使用可能）
DATA
DaTa
a（一文字だけでもOK）
a2
a123456789（最初の一文字以外は好きなように数字を使ってOK）
a_2（区切り文字としてアンダーバーを使うこともできる）

・変数名のダメな例

3b（最初の一文字が数字）
b 3（途中で半角スペースが入っている）
b（全角文字は論外）

今の段階でよくわからないのなら、とりあえず型を「int」と宣言すれば、整数の箱が出来上がるということだけ覚えていてください。

でも変数はプログラムの基本なので、変数の「型」についても、いずれは必ず覚えてくださいね。

さて、宣言に関して大まかに述べたので、次は「10 を代入し初期化」のところについて、軽く解説します。

どり～むきっず
～ゲーム制作支援サイト～
<http://dreamkids.sakura.ne.jp/>

「int data = 10;」と一行で書いていますが、実はこれで二つの処理をしているのです。

分解すると「int data;」と「data = 10;」になります。

「int data;」は先ほどはしよりはしよりで解説した変数の宣言に関してですね。では「data = 10;」が、「10を代入し初期化」ということになります。

ここで勘違いしやすいのは「=」の意味です。

C/C++言語では「=」は「左の変数に右のデータを代入」という意味になります。

つまり、「data = 10;」は「dataは10と等しい」ではなく「dataに10を代入」という事になります。

ようするに「10を代入し初期化」というのは「dataに10という最初の数値を入れる」という、たったそれだけのことです。

今回「プリントエフ」の説明は省略させていただきます。とりあえずこの「%d」と書いた所に変数「data」の中の数字を表示して、その後の改行コードによって改行します。

もし、自力で調べたい方に対してのヒントはprintfの「f」の意味は「フォーマット」を意味します。「printf フォーマット」等の条件で検索をかけてみてください。

話を戻すと、この最初の「プリントエフ」では「10」と表示されて改行されます。そして次は「5」を代入していますので、「5」と表示して改行します。

その次には変数「data」をそれぞれ「++」「+=」「--」「-=」として、そのつど表示していますが、これらは変数の数値を変える計算式です。それぞれの式の内容はコメントの通りです。これら以外にも計算式は沢山ありますので、必要に応じて調べてください。

こちらは「C言語 演算子」等で検索すれば目当ての情報を見つけられることでしょう。

今まで知らなかった計算式（演算子）を見つけたのなら、このプログラムを書き換えてその計算式を使い、変数の数値がどのように変化するか実際に確かめてみてください。

情報を何度も読むより、一度でも使ってみて体で覚えてください。

どり～むきっず
～ゲーム制作支援サイト～
<http://dreamkids.sakura.ne.jp/>

与えられたサンプルプログラムを組んで「ハイ終わり」ではなく、所々を書き換えてみて「この場合はどう動くのだろう」とか「こうしたらこうなるんじゃないか」とか考えて、自分なりのソース実験をしてみてください。

それが本当の意味での自分の知識になりますし、地道な方法ではありますが、繰り返しればプログラムを組む上で大切な「応用力」を身につけることもできます。

8、繰り返しの for 文

サンプルソース

```
void main()
{
    int i;    // ループ用変数

    // 10回繰り返す
    for( i = 0; i < 10; i++ )
    {
        printf( "繰り返し %d 回目\n", i );
    }

    getch();
}
```

処理結果



```
C:\test\Debug\test.exe
繰り返し 0 回目
繰り返し 1 回目
繰り返し 2 回目
繰り返し 3 回目
繰り返し 4 回目
繰り返し 5 回目
繰り返し 6 回目
繰り返し 7 回目
繰り返し 8 回目
繰り返し 9 回目
```

ソースが短くなりました（笑）

その代わりに、「for(i = 0; i < 10; i++)」という、変な式がありますね。
コメントによると「10 回繰り返す」との事。

では、これを「100 回繰り返す」に書きかえられますか？ ここまで真面目にやっ
てきたなら余裕でしょう（笑）

少し考えてみてから次のページの答えを見てくださいね。

「for(i = 0; i < 100; i++)」にすればいいだけです。これが「for 文」の主な使い方です。変数を使って繰り返すのにとっても便利です。

さて、ではこの「for 文」を解読してみましょう。かつこの中に計算式が三つあります。そのうち二つは、皆さんもおそらくわかりますね。「i = 0」「i++」です。

左の「i = 0」は「for 文」の一番最初の一回だけ、処理されます。
ループ用の変数である「i」に「0」という最初の数値を入れています。

そして右の「i++」は、変数の数値を1増やす「インクリメント」ですね。
この処理は「for 文」のすぐ後に続いている「{」から「プリントエフ」の次の行にある「}」までの処理が終わったら、実行されます。

問題は真ん中の「i < 10」ですね。これは条件式といいます。

for 文は、左が「前処理」右が「後処理」そして真ん中は「ループ条件」となります。

このサンプルの条件式は見たまんまで「変数 i が 10 未満だったら」という条件になります。この条件を満たしているなら「{」と「}」で指定している処理を実行します。
この条件式のチェックは「前処理」もしくは「後処理」が終わるたびに行われます。

このサンプルプログラムの処理の流れを表すと、

- ・ループ用変数「i」宣言
- ・「for 文、前処理」で「i」に0を代入し初期化
- ・「for 文、ループ条件」チェック (i の数値は 0)
- ・条件を満たしていたので「プリントエフ」を実行
- ・「for 文、後処理」で「i」をインクリメント
- ・「for 文、ループ条件」チェック (i の数値は 1)
- ・条件を満たしていたので「プリントエフ」を実行
- ・「for 文、後処理」で「i」をインクリメント
- ～ 中略 ～
- ・「for 文、ループ条件」チェック (i の数値は 10)
- ・条件を満たしていないので繰り返し終了

になります。

どり～むきっず
～ゲーム制作支援サイト～
<http://dreamkids.sakura.ne.jp/>

処理結果を見るとわかると思われませんが、変数「i」の数值は0～9でループされ、10回繰り返されていますね。

プログラムにおける数值の開始は「1」ではなく「0」とするように癖をつけておくと良いです。

このことに関しては以前、僕のメルマガでも発行しましたが今回は「プログラムにおいて0は重要な意味がある」とだけ覚えていてくださいね。

早瀬が配信しているメルマガについて
<http://dreamkids.sakura.ne.jp/article/top.html>

細かい部分はいろいろとはしょっていますが、for文での繰り返しループは色々な場面で使うので、せめて「for(i = 0; i < ?; i++)」の形(?は繰り返したい回数)だけは覚えておくと後々便利です。

9、インデント

今まであえて触れませんでした。「メイン関数」の後の「{」や上記の for 文のソースでも「{」の後に続く処理の前に Tab が入っています。「メイン関数」の後に続く所には1つ、for 文の後に続くところには「メイン関数」のものとあわせて2つ入っています。

このように、読みやすくするため字下げするのをインデントといいます。

これらは別に半角スペースでも構いませんが(勿論全角スペースはダメです)きちんとつけておくことをオススメします。

インデントをつけていないソース

```
// //////////////////////////////////////  
// メイン関数  
// ※C/C++言語のプログラムで必ず必要な、処理の開始位置です。  
//  
void main()  
{  
int i;    // ループ用変数  
  
// 10回繰り返す  
for( i = 0; i < 10; i++ )  
{  
printf( "繰り返し %d 回目\n", i );  
}  
  
getchar();  
}
```

上のソースを見てもらえばわかる通り、インデントをつけていないと読み辛いですよね。

わかりやすく読みやすいソースにすることを心がけましょう。

10、条件分岐の if 文

サンプルソース

```
void main()
{
    int i;    // ループ用変数


    // 10回繰り返す
    for( i = 0; i < 10; i++ )
    {
        printf( "繰り返し %d 回目", i );

        // 条件分岐
        if( 0 == i%2 )
        { // もし変数「i」が偶数だったら
            printf( ": 変数「i」は偶数です" );
        }

        printf( "\n" );    // 改行
    }

    getch();
}
```

処理結果



```
C:\¥test¥Debug¥test.exe
繰り返し 0 回目 : 変数「i」は偶数です
繰り返し 1 回目
繰り返し 2 回目 : 変数「i」は偶数です
繰り返し 3 回目
繰り返し 4 回目 : 変数「i」は偶数です
繰り返し 5 回目
繰り返し 6 回目 : 変数「i」は偶数です
繰り返し 7 回目
繰り返し 8 回目 : 変数「i」は偶数です
繰り返し 9 回目
```

どり～むきっず
～ゲーム制作支援サイト～
<http://dreamkids.sakura.ne.jp/>

先ほどのループ内に「if(0 == i%2)」という変な処理が追加されました。

このなかの「0 == i%2」は「for 文」のところで出てきた条件式です。

ちなみにこの式を書くと、変数「i」の数値が偶数なら条件を満たしていて、奇数なら条件を満たしていないということになります。

この式の細かな解説はしませんので、どうしてそうなるのか自力で調べてみて下さいね。

「%」の演算子の意味を調べて、ちょっと計算してみればわかると思われませう～

この条件分岐により、変数「i」が偶数だったら偶数だと表示されるようになりました。

条件式を色々変えてみて、色々試してみると良いですよ。

終わりに

この eBook サンプルペーパーは僕が過去に配信したメルマガの内容をいくつかまとめたものです。

メルマガ発行した後にしばらくアクセス解析してみると、プロジェクト構成やプログラムに関しての検索ワードで僕のサイトに訪れる方が結構いらっしゃったので、その内容を PDF としてまとめさせて頂きました。

つべこべ言わず、これらのプログラムを組めば、簡単な基礎は身に付くと思われま

す。基礎があれば、キチンと解説されているサイトや参考書等もすんなり読めると思われ

ますので、正しい知識をどんどん吸収できることでしょう。これらのサンプルプログラムで満足せずに、自分から色々調べて行ってプログラムの知識を深めて行って欲しいです。

そして、はじめに書いたとおり、このペーパーはコピーフリーです。

C/C++言語によるプログラミングを始めようという方の「最初の一歩」になればと思

いまとめあげたものですので、もし宜しければ他にプログラマーを目指している知り

合いが居るのであれば、ご自由に配ってください。サイトでの二次配布も許可しております……というよりも、是非とも行っていただ

きたいとまで思っております。一人でも多くの人に「プログラミングは楽しい」と思って頂けるように、これからも有益な情報を配信していきたいと思

どり～むきっず
～ゲーム制作支援サイト～
<http://dreamkids.sakura.ne.jp/>

早瀬 竜也 (たつにい)

同人ゲームプログラマー 兼 見習いインフォプレナー

メールアドレス

hakka17@hotmail.com

2008年1月10日 「同人ゲームが完成しない17の理由」発行

<http://dreamkids.sakura.ne.jp/info/001.html>

2008年4月27日 「gameLib マニュアル+」発行

<http://dreamkids.sakura.ne.jp/info/002.html>

つべこべ言わず組んでみよう ～体で覚えるC/C++プログラミング～

2008年3月18日 第1刷発行

著者 どり～むきっず

発行 早瀬 竜也 (たつにい)

どり～むきっず ～ゲーム制作支援サイト～

<http://dreamkids.sakura.ne.jp/>