# Hello, world!—その前に

　プログラミングの入門といえば“Hello, world!”ですね。早速書いてみたいのですが、皆さんはコンパイラとテキストエディタは手に入れていますか？この2つはポケモンで言えば「たいせつなもの」です。プログラム(==ソースコード)を書くには、テキストエディタがいります。notepad.exeで編集したければそれでいいのですが、せっかくなのでカッコイイテキストエディタを入手しましょう。各エディタの説明はしませんが、個人的にはSublime Text2がオススメです。

* notepad++
* Sublime Text2
* Vim
* Emacs

　エディタをゲットしたら、次はコンパイラを持ってこないといけません。D言語のコンパイラは有名なものでdmd, gdc, ldcとありますが、普通はdmdを使うので、dmdを[ここ](http://dlang.org/download.html)からダウンロードしましょう。Windowsの人は“dmd Windows installer”をダウンロードして実行すれば全部設定くれます。

　インストールが終わったら、コンソール(ターミナル)画面をたちあげてdmdと打ってみましょう。ズラズラズラ～と文字が出てきたら成功です。

# Hello, world!

　さて、テキストエディタをたちあげて以下の文字列をそのまま打ち込んでhelloworld.dで保存しましょう。

import std.stdio;  
  
void main()  
{  
 writeln("Hello, world!");  
}

　次にコンソールで以下をタイプ。

dmd -run helloworld

　さて、コンソールで以下のように出力されれば成功です、おめでとう！これで

Hello, world!

## 注意…

　これからは“dmd…”などのコマンドと実行結果を一緒に、以下のように表記します。

> dmd -run helloworld  
Hello World!

# Hello, world!を詳しくみてみる

　ここからはさっき書いたHello Worldプログラムを解析していきます。

## 1行目

import std.stdio;

　「モジュールstd.stdioを読み込む」という意味です。std.stdioのstdはstandardの略で、stdioはstandard Input/Outputの略です。std.stdioというのはstdパッケージに属するstdioモジュールという意味です。今は理解できないかもしれませんが、のちほどモジュールの項で説明します。  
　1行目を書く理由は、writelnが使いたいからです。std.stdioにはwritelnが定義されていますので、それをimportしてこないとwritelnが使えません。つまり、これから「fooというのを使いたい！fooはbarモジュールで定義されてる」という場合にはfooを使う前に“import bar;”と書かなければいけません。

## 2行目

　2行目はただの空白です。D言語のプログラムのソースコード中では任意の場所に空白を入れても構いません。もちろん、“import”を“i m p o r t”と書くのは不正です。

## 3行目

void main()

　D言語では、すべてのプログラムはmainという関数(function)から始まります。関数というのは、数学の関数とほとんど同じで入力(input)を受け取ったら、その入力を加工して出力(output)を返すものです。このmainの場合は何も受け取らずなにも返しません(void)が、void main(string[] args)と書けば、string[] argsを受け取ってなにも返さないmain関数となります。関数はソースコード上では以下のように記述されます。

OutputType fName(InputType1 arg1, InputType2 arg2, ...)  
{  
 function Body  
}

　OutputTypeは関数の出力の型(タイプ)、InputType1とかInputType2は入力の型(タイプ)で、arg1, arg2は仮引数(かりひきすう)と呼ばれます。また、InputType1, InputType2は引数の型, OutputTypeは返り値や戻り値の型(Returned Type)と呼ばれます。function Bodyは関数本体といい、入力から出力を生成する手順を記述します。いきなり意味がわからない単語が増加しましたが、ここは後から詳しく説明します。

## 4行目

{

　3行目の項で説明した関数の{}の開始のカッコです。ここから6行目の“}”までは関数本体となります。

## 5行目

writeln("Hello, world!");

　この行は、「writelnという関数に“Hello, world!”を引数として渡して、呼び出す」という意味です。writelnという関数はstd.stdioにあるというのは説明しました。writelnは、“write”, “ln”に分解できます。“write”は書きだす、“ln”は改行を意味します。つまり、writeln(“Hello, world!”)は「“Hello, world!”と書きだして改行する」という意味です。  
　writelnがあるんだから、writeもあります。writeは改行なしで出力すること以外はwritelnと同じ動作です。write, writelnについて後ほど詳しく説明します。

## 6行目

}

　main関数の関数本体の終了を表すカッコです。

# write, writelnと型について

　write, writelnはstd.stdioで宣言(==定義)されています。writeやwritelnは複数の引数を受け取ることができます。つまり、先ほどのHello Worldのプログラムを改変した以下のコードも有効です。コンパイルして実行してみましょう。

import std.stdio;  
  
void main()  
{  
 writeln("Hello, ", "World! ", 123);  
}

> dmd -run helloworld  
Hello, World! 123

　さて、“Hello,”や“World!”は“”でくくりましたが、123は“”でくくってません。というのは、“”でくくったものはstring(文字列)という型になるのに対し、123とそのまま書くとint(整数; integar)という型になります。ほとんどのプログラミング言語は、データに型を持ちます。ここでいうデータというのは“Hello,”とか123とかのことで、型というのはstring, intのことです。もしデータに型がなければどうなるでしょうか？コンピュータは0, 1で全てを表しているといいますが、123も“Hello,”も0と1で表されてしまいますから、文字なのか数値なのかよくかわらなくなってしまいます。そのようなことを避けるために、数値にはint, 文字列にはstringと型を付けるのです。そして、writelnはどんな型が入力に入っているかを確認できるので、それぞれの型に合わせて出力のフォーマットを変えてくれます。

# 関数について

　型についてちょっとはわかったと思うので、次は関数について少し理解しましょう。プログラムは関数とデータの集合だと考えることができます。「関数にデータを渡して、関数がデータを処理してなにか値を返す」というのがプログラムの流れです。関数は、上から順番に実行されていきます。

import std.stdio;  
  
void main()  
{  
 write("Hello, ");  
 write("World! ");  
 writeln(123);  
}

　上のコードを同様に実行しても、一つ前のものと同じ表示になると思います。もし、関数内の実行手順が上から下でないなら、ぐちゃくちゃになっているはずです。  
　main関数以外の関数を定義して呼び出すことも可能です。以下のコードでは何もしない関数fooを定義してmain関数から呼び出しています。

void main()  
{  
 foo();  
}  
  
  
void foo(){}

　fooの位置は重要でなくて、たとえばmain関数の上に書いても大丈夫です。

void foo(){}  
  
void main()  
{  
 foo();  
}

　foo();を2回呼び出すことも可能です。

void foo(){}  
  
void main()  
{  
 foo();  
 foo();  
}

# writef, writefln

　C言語を勉強した人は一度はprintfを使ったことがあると思います。D言語にもprintfのようにフォーマットを指定して数値とかを出力できるwritef, writeflnがあります。

import std.stdio;  
  
void main()  
{  
 writefln("%d : %d", 1, 2);  
 writefln("%s : %s", 2, 4);  
}

> dmd -run helloworld.d  
1 : 2  
2 : 4

　%dと書いた部分に1が入り、次の%dには2が入っていることがわかると思います。次の行では%sにそれぞれ2, 4が入っています。%dや%sはフォーマット指定子といい、%dは10進数出力を表します。%sはデフォルトの指定子で、この場合は%dに等しくなります。もし数値を16進数で出力したい場合には、

import std.stdio;  
  
void main()  
{  
 writefln("%x : %X", 200, 200);  
}

> dmd -run helloworld.d  
c8 : C8

という風に%x, %Xを使います。%xは小文字、%Xは大文字で出力します。  
　もし、文字列を出力したいなら%sを使います。

import std.stdio;  
  
void main()  
{  
 writefln("%s", "これは文字列");  
}

> dmd -run helloworld.d  
これは文字列

もし、%dや%xで文字列を出力した場合にはエラーが出ます。

import std.stdio;  
  
void main()  
{  
 writefln("%d", "これは文字列");  
}

> dmd -run helloworld.d  
object.Exception@C:\D\dmd2\src\phobos\std\format.d(2154): Incorrect format specifier for range: %d  
----------------  
0x00416574  
0x004163FF  
0x0040466A  
0x00404206  
0x0040416F  
0x004029C1  
0x004025F8  
0x0040256A  
0x00412078  
0x0040E329  
0x0040B3C0  
0x75A833AA in BaseThreadInitThunk  
0x77DD9EF2 in RtlInitializeExceptionChain  
0x77DD9EC5 in RtlInitializeExceptionChain

　もし、指定子がわからなかった場合には%sとしておけば最適なフォーマットで出力されるので、迷ったら%sです。

# コメント

　たとえば、複雑なコードを書いた後日、そのコードを読み返した場合に読めるかどうか。たぶん読めても、読みにくいと思います。そのために、ソースコードにはコメントを残しておきます。たとえば、「ここでhogehogeをfugafugaしてる」とかです。よいコメントを書けるプログラマはよいプログラマになれます(たぶん)。

//std.stdioを読み込む  
import stds.stdio;  
  
/\*\* main関数  
 \* Hello, World!と出力  
 \*/  
void main()  
{  
 /+  
 writeln("fugafuga");  
 /+  
 writeln("なんだと…");  
 +/  
 writeln("hogehoge");  
 +/  
  
 writeln("Hello, World!");  
}  
  
  
/\* ネストできないコメント  
ここはコメント  
 /\*  
 ここはコメント  
 \*/  
ここはコメントでない  
\*/  
  
/+ ネストできるコメント  
ここはコメント  
 /+  
 ここはコメント  
 +/  
ここはコメント  
+/  
  
/\*\* ドキュメント  
\*/  
  
/++ ネスト可能なドキュメント  
+/  
  
// １行コメント

# おわりに

　さて、第一回目の「Hello, World!」の項か終わりました。お疲れ様でした。実は今回はかなり内容が詰まっていて、書いてる途中から、「初心者にはつらいかなあ」と思っていました。プログラムの概念を考えると、今のノイマン型コンピュータを考えるところまで行き着いてしまい、D言語の話になかなか戻ってこれないので、このようなちょっと難しい形式になってしまいました。「猫でもわかるC言語プログラミング」という本が有名ですが、この本の最初の方を読んでいただけると大体理解できると思います。 　次回は数値の計算について書こうと思います。

# キーワード

* dmd
* import
* std.stdio
* main
* 関数(function)
* write, writeln, writef, writefln
* 型(type)
* 値(value)
* 数値(integer, int型)
* 文字列(string型)
* フォーマット
* コメント