

質問1 教科書や Web 上に載っているメソッドの定義の読み方が分かりません

回答

実際に例を用いて説明します。次は、Math クラスの `cos()` メソッドの定義です。

```
cos
```

```
public static double cos(double a)
```

指定された角度の余弦（コサイン）を返します。特例として、
* 引数が NaN または無限大の場合、結果は NaN になります。

パラメータ:

a - ラジアンで表した角度

戻り値:

引数の余弦（コサイン）

※教科書には簡略して書かれていますが、google で Java Math などのキーワードで検索をするとサンマイクロシステムズ社のホームページ上で公開されている上記のような詳細な定義が見つかります。

皆さんが自身のコードにメソッドを宣言するときと同様の形式で書かれています。順番に見ていくと、

public	アクセス制限なし
static	クラスメソッド
double cos(double a)	戻り値の型 double
	メソッドの名前 cos
	引数（パラメータ）の型 double

と分かります。

続いて、メソッドの機能と引数、戻り値の意味が文章で書かれています。

引数（パラメータ）	a - ラジアンで表した角度
戻り値	引数の余弦（コサイン）

ここで、各引数と戻り値の意味と単位が記述されます。この場合、引数の単位は、度ではなくラジアンである必要があることが分かります。

実際に、使用するときは、

```
double rad=90 * 3.141592 /180;  
double c=Math.cos(rad);
```

のように記述すればよいです。

質問2 クラス String のメソッド int indexOf(int ch)を使うとき、教科書では以下のようになっており、引数に char 型の値を渡しています。型が宣言と一致していませんが大丈夫なのですか？

```
String str="Hello";
char ch='e';
int num = str.indexOf(ch);
```

回答

まず、文字リテラルと char 型について押さえましょう。

'a'や'b'の文字リテラルは Java の内部では **16 ビット Unicode** で表現されています。16 ビット Unicode とは、文字を数字で対応付けて表現したコード（文字コードという）のことであり、実体は文字を表す 16 ビットの符号なしの整数です。

char 型は **16 ビットの符号なし整数**を保持することにより、16 ビット Unicode で表現される 1 文字を保持します。このため、char 型は 16 ビットの符号なし整数型と捉えることもでき、16 ビットの符号なし整数（0~65535）を代入して扱うことができます。

実は char 型は int 型と同じように整数を扱う型ということになります。

さて、次に押さえておきたいのは型の**拡張変換**です。型というのは扱う値の範囲の大きさに応じてランク付けされています。大きいランクの型は、より広範囲の値を表現できます。このとき、ランクの低い型の値をランクの高い型の変数へ代入する場合は、自動的に型が変換されます。これを型の拡張変換といいます。メソッドの引数に値を渡すときも、代入と同様に型の拡張変換が行われます。

char 型と int 型は、

- char 型 16 ビット符号なし整数
- int 型 32 ビット符号付き整数

より、char 型より int 型の方がランクが高いです。したがって、char 型の値を int 型の変数に代入する場合は拡張変換が行われます。

上の質問に戻ると、変数 ch は 1 文字 e を表す 16 ビットの符号なし整数を保持しています。これは、型の拡張変換により、メソッド indexOf()の引数に渡され、内部ではその整数は文字を意味するデータとして処理されることになります。

したがって、型が一致していなくてもこの場合は問題が起こらないことになります。

参考)

文字リテラル、Javaプログラミング1「ようこそJavaへ」の回

char 型、Javaプログラミング1「変数」の回

拡張変換、Javaプログラミング1「演算子の優先順位と変数の型変換」の回