第1章　「まずは慣れよう」のまとめ

1.1　用語のまとめ

教 p.2

コンパイル　とは、ソースファイル（人間が読む・書く形式、C 言語の場合拡張子は __）を実行ファイル（CPU が直接理解できる形式、Windows 上では拡張子は ____）などに、翻訳することである。

教 p.3

注釈（コメント）　とは、ソースファイル中の人間向けのメッセージで、コンパイラは無視する部分である。C 言語では ___から ___までが注釈である。さらに新しい C 言語の仕様（C99）では ___から行末までという形も利用できる。（bcc32 でも使用可能）

教 p.4

printf　は、表示を行うための関数である。関数とは、_____ _____である。

教 p.6
教 p.318

printf の変換指定のまとめ　printf の第 1 引数のなかで、%から始まる部分は変換指定と言い、第 2 引数以降の値に順に置き換えられる。整数（10 進数）は ___、浮動小数点数は ___を使う。

教 p.9

文字列リテラル　とは、一連の文字を二重引用符"で囲んだものであり、文字のデータを表す。

教 p.9

拡張表記　¥n は _____、¥a は警報（ベル）、¥t は _____を表す。このような、¥を使った書き方を拡張表記という。その他の拡張表記については、教科書 p.203 を参照すること。

教 p.10

変数　とは、数値などのデータを収納するための「箱」である。C 言語では、変数を使うためには事前に宣言が必要である。

```
int vx;        /* int（整数）型の変数 vx の宣言 */
double fx;     /* double（倍精度浮動小数点数）型の変数 fx の宣言 */
int vx, vy;    /* int（整数）型の変数 vx と vy の宣言 */
```

問. C 言語の変数にはどのような名前をつけることができるか？(→ 教科書 p.82)

教 p.11

代入　とは、変数の値を書き換えることである。「変数 = 式」という形式で、右辺の式の値を左辺の変数に代入する。

教 p.12 **scanf 関数**　とは、キーボードから数値などデータを読み込むための関数である。

```
scanf("___", &no);  /* キーボードから整数を変数 no に読み込む */
scanf("____", &fx);  /* キーボードから実数を変数 fx に読み込む */
```

puts 関数　は、文字列を出力し、最後に改行を行う。printf のように書式変換は行わない。

教 p.14

1.2　文法のまとめ

式 (expression) とは　これまでのところ、

分類	一般形	補足説明
変数		x, i など
整数リテラル		1, 0, 100, 0xff など
文字列リテラル	"〜"	"Hello¥n"など
関数呼出し	関数名 (式, ... , 式)	printf("Hello¥n") など
単項演算	単項演算子 式	単項演算子は +, -, &, ... など
二項演算	式 二項演算子 式	二項演算子は +, -, *, /, = ... など
かっこ	(式)	演算の順番を指定するため、括弧で囲んだもの

文 (statement) とは　これまでのところ、

分類	一般形	補足説明
式文	式 _	式は通常、代入式か関数呼出しのように _____があるもの
return 文	return 式 ;	式の周りにかっこは必要なし

宣言 (declaration) とは　これまでのところ、

分類	一般形	補足説明
変数宣言	型 変数名, ... , 変数名 ;	型は int, double など

第2章　「演算と型」のまとめ

2.1　用語のまとめ

教 p.18

演算子　とは、+, -, *, /のように 演算の働きを持った記号のことである。(教科書 p.177 に C 言語のすべての演算子の表がある。)

＿＿＿＿＿＿＿＿とは、その演算の対象となる式のことである。

教 p.19

/ 演算子

整数 / 整数

という演算では、整数としての割算 (小数点以下は ＿＿＿＿) としての結果が得られる。

整数 % 整数

では、余りを求める。

教 p.19

printf で%文字を表示　＿＿と 2 つ重ねることで%という文字を出力することができる。puts 関数では、%はそのまま出力される。

教 p.25

double 型　とは、いわゆる実数 (正確には浮動小数点数) を扱うための型である。もちろん、実数と言っても精度には限界がある。

教 p.27

変換指定のまとめ　以下の表くらいは、暗記しておくこと。

	int	double
printf	%d	＿＿＿
scanf	＿＿＿	＿＿＿＿

教 p.28

型と演算

実数 / 実数

の演算では、切捨ては行わず、通常の割算が行われる。int と double が混じっている場合、

整数 / 実数

や

実数 / 整数

の場合も、整数 (int) 型のオペランドが実数 (double) 型に ________ され、double 型の演算となる。

教 p.30 **キャスト (cast)** とは、________することである。

(型) 式

という形で、「式」の値を「型」としての値に変換する。例えば、

```
int na, nb
...
(double)(na + nb) / 2
```

教 p.177 では、double 型としての割算が行われる。(演算子の優先順位に注意する。割算よりもキャストが先に行われる。)

教 p.318

高度な変換指定 以下のような変換指定は必要に応じて調べれば良い。

説明	例	出力
桁数を揃える	printf("[%3d]", 1)	[1]
桁数を揃え先頭を 0 で埋める	printf("[%03d]", 1)	[001]
小数点以下の桁数を指定する	printf("[%.3f]", M_PI)	[3.142]
16 進数で表示する (小文字)	printf("[%x]", 127)	[7f]
16 進数で表示する (大文字)	printf("[%X]", 127)	[7F]

2.2 文法のまとめ

式 (expression) に以下を追加、

分類	一般形	補足説明
浮動小数点数リテラル		3.14, 2.0, 6.02e23, 6.6626e-34 など (教 p.27)
キャスト	(型) 式	明示的な型変換 (教 p.31)