2018/10/05

竹島研究室 創成課題

第２回 Ｃ言語演習

変数と演算

東京工科大学　加納 徹

前回の復習

「Hello, world!」と表示するプログラム

```c
#include <stdio.h>

int main(void) {
    printf("Hello, world!¥n");
    return 0;
}
```

あらゆる言語のベースになっているよ

プログラム実行の流れ

1. 作業ディレクトリへの移動

$ cd work

2. ソースファイルの作成

$ touch hello.c

$ cygstart hello.c

3. ソースファイルのコンパイル

$ gcc hello.c

4. プログラムの実行

$ ./a

その他のコマンド

- **ディレクトリの作成**

  $ mkdir [dir_name]

- **一つ上のディレクトリに移動**

  $ cd ..

- **ファイルのコピー**

  $ cp [fileA] [fileB]

- **Cygwinの終了**

  $ exit

１．変数とは

データの入れ物「変数」

変数とは

[データ型] [変数名] ;

の形で宣言される、数値や文字を入れておく入れ物です

データ型	種類	サイズ	扱える範囲
short	整数値	2バイト	−32786~32767
int	整数値	4バイト	−2147483648~2147483647
long	整数値	8バイト	−9223372036854775808~9223372036854775807
float	実数値	4バイト	有効数字7桁 $\pm 10^{-38}$~10^{38}
double	倍精度実数値	8バイト	有効数字15桁 $\pm 10^{-308}$~10^{308}
char	文字	1バイト	ASCII文字 $(-128 \sim 127)$
char[]	文字列	不定	サイズを整数で指定

データの入れ物「変数」

変数とは

[データ型] [変数名] ;

の形で宣言される、数値や文字を入れておく入れ物です

データ型	種類	サイズ	扱える範囲
int	整数値	4バイト	$-2147483648 \sim 2147483647$
double	倍精度実数値	8バイト	有効数字15桁 $\pm 10^{-308} \sim 10^{308}$
char	文字	1バイト	ASCII文字 $(-128 \sim 127)$

2. 変数の宣言・代入・参照

変数を使う流れ

1. 変数の宣言

- [データ型] [変数名] の形で宣言
  （例: int number; char letter;）

2. 変数に対してデータを代入

- 代入演算子 ‘=’ を用いて変数に値を格納
  （例: number = 128; letter = ‘A’）

3. 変数の値を参照

- フォーマット指定子により値を参照

変数の宣言と代入

```
1  // 変数の宣言
2  int x, y;
3  double value;
4  char letter;
5
6  // 値の代入（初期化）
7  x = 10;
8  y = -5;
9  value = 3.1415;
10 letter = 'P';
11
12 // 値の代入（更新）
13 x = y;
```

```
1  // 変数の宣言と初期化
2  int x = 10, y = -5;
3  double value = 3.1415;
4  char letter = 'P';
```

変数の命名規則

- 守らないといけないルール
  - 半角のアルファベット、数字、アンダースコアのみを用いる
  - 先頭は必ずアルファベットかアンダースコアを用いる
    （数字は不可）
  - 予約語は使用できない
    （int, double, if, switch, for, do, return 等）

- 守ったほうが良いルール
  - a, b, cなど、意味のわかりにくい名前は避ける
  - アルファベットは全て小文字を用いる
    （定数の場合は全て大文字）
  - 複合語は単語の間にアンダースコアを入れる

printf 関数による値の参照

- 値の出力方法：

  printf(“フォーマット指定子”, 変数);

- フォーマット指定子

型名	データの種類	指定子
int	整数値	%d
long	倍長整数値	%ld
float	実数値	%f
double	倍精度実数値	%lf
char	文字	%c
char[]	文字列	%s

演習 2-1

以下のソースコードを参考に、年齢・身長・血液型を
表示するプログラムを作成して下さい

```c
#include <stdio.h>

int main(void) {
    // 変数の宣言と初期化
    int age = 29;
    double height = 175.8;
    char blood = 'A';

    // 変数の参照
    printf("   age = %d¥n", age);
    printf("height = %lf¥n", height);
    printf(" blood = %c¥n", blood);

    return 0;
}
```

変数表示の詳細設定

- フォーマット指定子を使う際、
  表示桁数などを指定することが可能

使用例	概要	結果例
%3d	全体の幅3桁で整数値を表示	123 65
%6.3lf	全体の幅6桁、少数点以下の幅3桁で実数を表示	123.45 32.198
%03d	全体の幅3桁、0詰めで整数値を表示	123 065
%06.3lf	全体の幅6桁、少数点以下の幅3桁、0詰めで実数を表示	123.450 032.198
%+3d	全体の幅3桁、符号付きで整数を表示	+123 -65
%-3d	全体の幅3桁、左寄せで整数を表示	123 65

演習 2-2

演習 2-1 のソースコードを修正し、
以下のような表示になるようにして下さい

実行例

```
    age =    29
 height = 175.8
  blood =     A
```

余裕がある人は、文字列も変数として扱ってみよう

3. 定数

定数

- 変数：値が変化する可能性あり
  （[データ型] [変数名] = [値];）
- 定数：値が変化する可能性なし
  （#define [定数名] [値]）

```c
#include <stdio.h>
#define RATE 101.338 // 為替レート

int main(void) {
    printf("現在のレートは1ドル = %lf 円です。¥n", RATE);
    return 0;
}
```

4. 演算

基本的な算術演算子

演算子	役割	例	意味
+	加算	z = x + y;	xにyを足した値をzに代入
-	減算	z = x - y;	xからyを引いた値をzに代入
*	乗算	z = x * y;	xにyを掛けた値をzに代入
/	除算	z = x / y;	xをyで割った値をzに代入
%	剰余算	z = x % y;	xをyで割った余りをzに代入

```
int x = 18, y = 4;
printf("x = %d, y = %d¥n", x, y);
printf("x + y = %d¥n", x + y);
printf("x - y = %d¥n", x - y);
printf("x * y = %d¥n", x * y);
printf("x / y = %d¥n", x / y);
printf("x %% y = %d¥n", x % y);
```

演習 2-3, 2-4

演習 2-3

int 型の変数 a, b, c を用意し、a と b に適当な値を入れ、
「加算」「減算」「乗算」「除算」「剰余算」の結果を c に格納し、
printf 関数を用いて c の値を表示して下さい

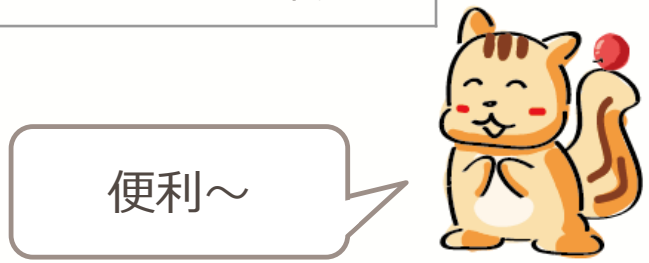
演習 2-4

double 型の変数 x, y, z を用意し、x と y に適当な値を入れ、
「加算」「減算」「乗算」「除算」の結果を z に格納し、
printf 関数を用いて z の値を表示して下さい
（実数値の「剰余算」は「%」演算子で行うことができません）

算術演算子 + 代入演算子

- 算術演算子と代入演算子は組み合わせが可能
  （例： x = x + y;　→　x += y）

演算子	役割	例	意味
+=	加算代入	x += y;	xにyを足した値をxに代入
-=	減算代入	x -= y;	xからyを引いた値をxに代入
*=	乗算代入	x *= y;	xにyを掛けた値をxに代入
/=	除算代入	x /= y;	xをyで割った値をxに代入
%=	剰余算代入	x %= y;	xをyで割った余りをxに代入



インクリメント/デクリメント演算子

- インクリメント演算子 (++)
  - 整数値を 1 だけ増やす
    （例： index++;）
- デクリメント演算子 (--)
  - 整数値を 1 だけ減らす
    （例： index--;）

通常の算術演算と比べて、超高速に動作する

5. キーボードからの入力

scanf 関数によるキー入力

- scanf によるキー入力の受付け

  scanf(“[フォーマット指定子]”, &[変数名]);

```c
#include <stdio.h>

int main(void) {
    int age;
    double height;

    printf("年齢を入力して下さい:");
    scanf("%d", &age);

    printf("身長を入力して下さい:");
    scanf("%lf", &height);

    printf("age = %d, height = %.2lf¥n", age, height);

    return 0;
}
```

演習 2-5

「scanf」を用いて生まれ年と現在の年を西暦で受け取り、
年齢を出力するプログラムを作成して下さい

実行例

```
生まれ年（西暦）を入力して下さい: 1987
現在の年（西暦）を入力して下さい: 2018
2018年、あなたは今年で 31 歳です。
```

終わり