第3章　基本データ型　　CPro:03-01

概要

・基本データ型	型宣言	ビット長
int　整数型	int	32
long　整数型	long	32 or 64
short　整数型	short	16
unsigned　整数（符号なし）	unsigned	32
	unsigned long	32 or 64
	unsigned short	16
char　文字型（実は整数型）	char	8
	unsigned char	8
float　浮動小数点型	float	32
double　倍精度浮動小数点型	double	64

03-02

Cのデータ型

・整数系
　文字型を含む

・実数（浮動小数点数）系
　単精度浮動小数点数
　倍精度浮動小数点数

・ポインタ型

３．１　整数型　　03-03
３．２　long型整数とshort型整数
３．３　unsigned型整数

Cの整数型
・ビット長：
　８　　char　・・・　文字（1文字分）にも使う
１６　　short
３２　　int,　　long（一般的な処理系で）
６４　　long（一部の処理系のみ）

・符号：
　符号付き　　signed　・・・　省略可
　符号なし　　unsigned　　・・・　ビット長の指定と組み合わせる
［例］ unsigned char
　　unsigned short
　　unsigned int　または　unsigned
　　unsigned long

8ビット　　03-04

char	−128	～	127	
unsigned char	0	～	255	

16ビット

short	−32768	～	32767	
unsigned short	0	～	65535	

32ビット

int	-2^{31}	～	$2^{31}-1$	（約20億）
unsigned	0	～	$2^{32}-1$	（約40億）
または unsigned int				
long	-2^{31}	～	$2^{31}-1$	
unsigned long	0	～	$2^{32}-1$	

64ビット

int	-2^{63}	～	$2^{63}-1$	（約？）
unsigned	0	～	$2^{64}-1$	（約？）

３．４　定数　　03-05

整数定数
　単に　数字を書けば　・・・　int型になる
　（範囲内であれば，char型，short型でも使える）

［例］　1　0　-34　32767　-6543210987

明示的に　unsigned型　long型　または　unsigned long型　にしたいとき
　unsigned型：　後ろに　U（またはu）をつける
　long型：　後ろに　L（またはl）をつける
　unsigned long型　後ろに　UL（まはたul）をつける

［例］　1U　345U　0U　98765487657U
　　1L　-345L　0L　6543765L
　　1UL　345UL　0UL　987654321UL

３．５　文字型　　03-06

キーワード：　char

（例）
```
char　c;
c = 'B';
```

8ビット長　・・・　ASCIIコード１文字分のみ
整数として記憶する
範囲：　char　　－128～127
　　unsigned char　0～255

３．６ 何進数を使うか　CPro:03-07

１０進数　普通に書けば １０進数　（例： 0　1　123　–45）

８進数　定数の最初に’0’をつける．数字は0 … 7のみ
（例： 037　0123　0045　　×080は間違い）

１６進数　定数の最初に’0x’をつける．
数字は，0 … 9 a b c d e f （大文字も可）
（例： 0x037　0x123　0x45　0xff　0xfedc123）

後ろに’L’をつけるといずれもlong型定数になる

※ 変数の宣言に何進数かというものはない　：
　１０進，８進，１６進は表示（プログラム上での表記）の問題であり，内部ではすべて2進で記憶されている．
　すなわち，何進数で書いても，数としては一つの値になる．

３．７ エスケープシーケンス　03-08

文字型の定数の一種
一般的な文字は，アスキーコードで変換して数値として記憶される
（例）’A’ ・・・ 0x41（65）　’B’ ・・・ 0x42（66）
’Z’ ・・・ 0x5a（90）　’a’ ・・・ 0x61（97）
’0’ ・・・ 0x30（48）　’9’ ・・・ 0x39（57）
’ ’ ・・・ 0x20（32）空白文字
’+’ ・・・ 0x2b（43）　等々．

文字に割り当てられていないコード（数）
→ エスケープシーケンスで表記： 主に制御文字
＼0　ヌル　・・・ 0x0（0）
＼n　復帰改行 ・・・ 0x0a（10）
＼t　タブ　・・・ 0x9（9）
＼b　バックスペース ・・・ 0x08（8）
＼r　復帰（リターン） ・・・ 0x0d（13）
＼＼ バックスラッシュ ・・・ 0x5c（92）　｝他の意味があるためエスケープする必要
＼’　クウォート ・・・ 0x27（39）　｝

３．８ 数値のエスケープシーケンス 03-09

文字にないその他の数値を書く必要がある場合
＼に続いて3桁以内の8進数を書く: 文字として扱われる.

（例）＼007 （ベル）

```
char    c;
c = '＼007';
printf("%c", c);
```

↓↓ 同じ意味.

```
char    c;
c = 0x7;        または    c = 7;
printf("%c", c);
```

なので, 一般的には使用しない.
（非文字コードは, 16進数で書くのが一般的）

"test3_2.c"

```
 1  #include        <stdio.h>       /*  "test3_2.c"  */
 2
 3  main()
 4  {
 5          int     a;
 6
 7          a = 3 / 2;
 8
 9          printf("%d¥n", a);
10  }
```

"test3_2a.c"

```
 1  #include        <stdio.h>       /*  "test3_2a.c"  */
 2
 3  main()
 4  {
 5          float   a;
 6
 7          a = 3 / 2;
 8
 9          printf("%f¥n", a);
10  }
```

“test3_3.c”

```
 1  #include        <stdio.h>       /*  ″test3_3.c″  */
 2
 3  main()
 4  {
 5          char    a;
 6
 7          a = 200;
 8
 9          printf(″%d¥n″, a);
10  }
```

３．９ 浮動小数点型

03-10

単精度は float
倍精度は double

単精度型: キーワード float
符号1ビット，指数7ビット，仮数２４ビット

従って，精度は10進で約6桁程度

（例）

```
float    a, b, c, d;

a = 3.141592654;      （精度は維持されない）
b = -987.654;
c = 5.0;
d = 6.02e23;    /* ＝＞6.02×10^23  */
```

３．１０ 倍精度型 03-11

キーワード　double

符号1ビット，指数7ビット，仮数５６ビット

従って，精度は１０進で１０数桁程度

（例）

```
double a, b, c, d;

a = 3.141592654;
b = -987.654;
c = 5.0;
d = 6.02e23;
```

定数の表記はfloatもdoubleも同じ.
代入する変数の型による.

３．１１ const修飾子 （省略） 03-12

３．１２ 変数の初期化

変数の型宣言と同時に値の初期化ができる

（例）

```
int      a = 5;
float    x = 3.14;
double y, z = -3.0;    （yは初期化されない．zのみ-3.0で初期化）
long     i = 50, j = 100;
char     c = 'Y';
char     str[20] = "This is mojiretsu.";
int      b[10] = {1, 3, 5, 7, 9, 11, 13, 15, 99, 87654};
```

"test3_1.c" CPro:03-13

```
 1 #include      <stdio.h>     /* test3_1.c (p.80) */
 2 main()
 3 {
 4         int     sick_days, vacation_days, crazy_days;
 5         long    population;
 6         short   diet_days;
 7         unsigned      weight_gain;
 8         char    consent, yes, no;
 9         float   salary;
10         double taxes;
11
12         sick_days = 5;
13         vacation_days = 10;
14         crazy_days = 350;
15
16         population = 2048224300;
17
18         diet_days = 4;
19         weight_gain = 6500;
20
```

03-14

```
21         consent = 'c';
22         yes = 'y';
23         no = 'N';
24
25         salary = 92.3e4;
26         taxes = 10.005E5;
27 }
% cc test3_1.c
% a.out
%
```

○ Big Endian と Little Endian 03-15

複数バイトで表現される数（4バイト整数など）を
上位バイトから順にメモリ上に格納する方式 ・・・ Big Endian
下位バイトから順にメモリに格納する方式 ・・・ Little Endian

（例） 4バイト整数（16進数で） 0x 01 23 ab cd

	Big Endian	Little Endian
x番地	01	cd
x + 1番地	23	ab
x + 2番地	ab	23
x + 3番地	cd	01

Big Endian : SPARC（Sun Micro），68000系（Motorola） など
Little Endian : x86系（Intel） など のプロセッサ

違いが問題となる可能性
・ネットワークを通してバイト単位でデータをやりとりする場合
・異なるシステム間でバイナリファイル等を交換する場合
・異なるシステムにプログラムを移植する場合

03-16

参考文献
「舶用電気・情報基礎論」：若林伸和著，成山堂書店
ISBN978-4-425-43171-7 定価 3,600円（税別）
生協で 5% off

（第1部 電気・電子・通信工学の基礎）

第2部 情報工学の基礎

○ … 基本かつ重要問題,　無印 … ふつうの問題,　☆ … 難易度高の問題

○３－１　前章の問題（２－１）と同様の計算（２つの引数の加減乗除）を，浮動小数点数のデータを引数として処理する関数を作れ．関数名はそれぞれ addf, subf, mulf, divf とし，その実行を試すためのメイン関数も同じ１つのソースファイルに作れ．実行結果を確認せよ．

３－２　文字型の変数に ’a’ を代入し，その変数を文字および数値として１０進，８進，１６進で値を出力するプログラムを作れ．また，その変数に整数５を足した結果を文字および１０進整数，８進整数，１６進整数として出力せよ．

（ヒント）
printf 関数で，整数型（文字型を含む）データを数値として出力するには　１０進の場合　“%d”編集，８進の場合　“%o”編集，１６進の場合　“%x”編集を，また文字型データを文字として（１文字）出力するには ″%c″ 編集を用いる．

○３－３　char 型で定義した変数と unsigned char 型で定義した変数にそれぞれ 200 を代入し，printf 関数でそれらの値を１０進数として出力するプログラムを作成し，実行結果を考察せよ．

３－４　１６進の数 53, 79, 73, 74, 65, 6d を，順に文字として putchar 関数で出力するプログラムを作り，実行せよ．なお，最後に改行文字をやはり putchar 関数で出力すること．

○３－５　つぎのプログラムを入力して，実行結果を確認せよ．

```
 1  #include        <stdio.h>
 2
 3  main()
 4  {
 5          printf(″char:       %d\n″, sizeof(char));
 6          printf(″unsigned char: %d\n″, sizeof(unsigned char));
 7          printf(″short:      %d\n″, sizeof(short));
 8          printf(″int:        %d\n″, sizeof(int));
 9          printf(″unsigned:   %d\n″, sizeof(unsigned));
10          printf(″long:       %d\n″, sizeof(long));
11          printf(″float:      %d\n″, sizeof(float));
12          printf(″double:     %d\n″, sizeof(double));
13  }
```

３－６　前問（３－５）を実行した後，つぎのプログラムを入力して，実行結果を確認せよ．

```
 1  main()
 2  {
 3          short   s = 1;
 4          int     i = 1;
 5          double  d = 1.0;
 6          int     j;
 7
 8          printf("        short   int   double\n\n");
 9
10          j = 0;
11          printf("10^%2d = %12d %12d %25.2f\n", j, s, i, d);
12
13          for (j = 1; j < 18; j ++) {
14              s = s * 10;
15              i = i * 10;
16              d = d * 10.0;
17              printf("10^%2d = %12d %12d %25.2f\n", j, s, i, d);
18          }
19  }
```


３－７　標準入力から読み込んだ文字について，英大文字から英小文字に変換して標準出力に書き込むプログラムを作れ．ただし１文字だけ処理できればよい．

（ヒント）
　文字の入力は，getchar() 関数，出力は putchar(文字) 関数を利用する．

03-21

○３－８　つぎのプログラムを実行したときの結果を予想し，実際に試してみよ．

```
 1  main()
 2  {
 3          int     i, j, k;
 4          float   w, x, y, z;
 5
 6          i = 3 / 2 * 2;
 7          j = 3.0 / 2.0 * 2.0;
 8          k = (int)3.0 / (int)2.0 * (int)2.0;
 9          w = 3 / 2 * 2;
10          x = 3.0 / 2.0 * 2.0;
11          y = (float)3 / (float)2 * (float)2;
12          z = 3 / 2 * 2.0;
13
14          printf("%d %d %d\n%f %f %f %f\n", i, j, k, w, x, y, z);
15  }
```

上記プログラムの実行結果をふまえて，

```
int     a, b;
a = 3.0 / 2 * 2;
b = (int)3.0 / (int)2.0 * 2.0;
```

で，a，b の値はそれぞれいくらになるか予想した後，実際に試してみよ．

03-22

○３－９　数学関数用のヘッダファイル〈math.h〉をインクルードし，その中で定義されている円周率 M_PI を，単精度浮動小数点型の変数と，倍精度浮動小数点型の変数にそれぞれ代入し，それら変数の値を printf 関数により，いずれも小数点以下 20 桁まで出力するプログラムを作成せよ．

プログラムを実行し，結果を "math.h" 中の定義と比較して考察せよ．