

C プログラミング 入門

— 文字と文字列 —

早稲田大学

本日の目標

- 文字と文字列の概念を理解する
 - ASCII コード
 - 文字 (char) 型
 - 文字列 ("¥0" 終端)
 - 初期化と代入
 - 文字列の引数渡し
- 文字列関数を使う
 - strcpy
 - strlen
 - strcmp

文字の扱い

- コンピューターは文字を直接扱えない
2進数でデータを保持する際、直接文字を記録するのが困難

- 例えば、「A」、「a」という文字を扱う場合：

```
char x, y;           /* char 型の変数を宣言 */  
x='A';              /* A という文字（実際は数字の 0x41）を代入）*/  
y='a';              /* a という文字（実際は数字の 0x61）を代入）*/
```

- このように、文字を数字に置き換えて扱う

ASCII コード

- 文字は ASCII コードと呼ばれる規格で定められている
- コンピューターの文字の持ち方

	0x00	0x10	0x20	0x30	0x40	0x50	0x60	0x70
0x00	¥0(終端文字)	(制御文字)	(半角空白)	0	@	P	'	p
0x01	(制御文字)	(制御文字)	!	1	A	Q	a	q
0x02	(制御文字)	(制御文字)	"	2	B	R	b	r
0x03	(制御文字)	(制御文字)	#	3	C	S	c	s
0x04	(制御文字)	(制御文字)	\$	4	D	T	d	t
0x05	(制御文字)	(制御文字)	%	5	E	U	e	u
0x06	(制御文字)	(制御文字)	&	6	F	V	f	v
0x07	(制御文字)	(制御文字)	'	7	G	W	g	w
0x08	(制御文字)	(制御文字)	(	8	H	X	h	x
0x09	¥ t (タブ)	(制御文字)	)	9	I	Y	i	y
0x0a	¥ n (改行)	(制御文字)	*	:	J	Z	j	z
0x0b	(制御文字)	(制御文字)	+	;	K	[	k	{
0x0c	(制御文字)	(制御文字)	,	<	L	¥	l	—
0x0d	(制御文字)	(制御文字)	-	=	M	]	m	}
0x0e	(制御文字)	(制御文字)	.	>	N	^	n	~
0x0f	(制御文字)	(制御文字)	/	?	O	_	o	(制御文字)

文字

- (半角)文字を扱うには, **char** 型を用いる
- char 型 (1byte) の変数は, **1 文字**を記憶しておける
- 個々の文字はシングルクォーテーション「'」で囲む

```
char x;           /* char 型の変数を宣言 */  
x='A';           /* A という文字 (実際は数字の 0x41) を代入) */
```

- printf,scanf も整数型や実数型と同様に使える

```
char x;           /* char 型の変数を宣言 */  
scanf("%c",&x);   /* キーボードから 1 文字入力して x に代入 */  
printf("%c",x);   /* 文字 x を画面に表示 */
```

文字列

- 複数の文字（文字列）を扱いたい場合は，char 型の配列を使う．

```
char s[100]; /* 100 文字分の配列を宣言 */
```

- C 言語では，文字列がど

こまで続くのか（何文字なのか）を表すのに，**終端文字**「'\0'」を使う．

```
char x[100]; /* 100 文字分の配列を宣言 */  
x[0] = 'I'; x[1] = 'T'; x[2] = 'b'; x[3] = '\0';
```

- 文字列の最後の文字の次に終端文字を追加することで，文字列がここまでということを示す．
- printf, scanf も 1 文字の場合と同様

```
scanf("%s", x); /* 文字列を入力して char 型配列 x に代入 */  
printf("%s", x); /* 文字列 x を画面に表示 */
```

- **&**が要らないことに注意
- 読み込むには，予め十分な配列サイズを用意する

文字列使用例

名前を入力して，それを画面に出力する

```
#include <stdio.h>
int main(void){
    char Name[100];          /* char 型配列．何文字か分からない多めに確保 */
    printf("Input your name:");
    scanf("%s",Name);        /* 名前を入力．&Name とはしないことに注意 */
    printf("Your name is %s.\n",Name);
    return 0;
}
```

- 「WASEDA」と入力した場合，Name 配列の各要素は

[0]	[1]	[2]	[3]	[4]	[5]	[6]
'W'	'A'	'S'	'E'	'D'	'A'	'¥0'

となる．7 番目以降は不定

- '¥0' の手前までが printf で出力される
- スペースを含む文字列は，scanf ではスペースの前後で別個の文字列と認識されるので注意

文字列の初期化と代入

- 補足：初期値の個数に応じて配列の要素数を決めたいときには，配列の要素数を空欄にしておくこともできる

```
int S[] = {1, 2, 3, 4, 5};
```

- char 型配列の初期化：char 型配列の初期化も同様．特別なやり方も許される

```
char Name[] = {'W', 'A', 'S', 'E', 'D', 'A', ' ', 'T', 'a', 'r', 'o', '\0'};  
char Name[] = "WASEDA Taro";      /* 特別な char 型配列初期化 */
```

- 1 つ目は各要素を 1 文字ずつ設定する方法．この場合最後に，終端文字【**¥0**】が必要
- 2 つ目は char 型配列の特別な初期化方法．文字列を**ダブルクォーテーション【"】**で囲む

例題

例題：名前を表示するプログラムを作れ

- ファイル名は「name.c」とする
- 姓，名を順に入力する（姓・名それぞれに char 型配列を用意）
- 画面に姓名を表示する
- 表示は以下のようにする

Input family name: **WASEDA**

Input given name: **Taro**

Your name is **WASEDA Taro.**

文字列を引数渡す

文字列を別の関数に渡したい場合はポインタを使う

```
#include <stdio.h>
void message(char *mes){
    printf("Message: %s %n",mes);    /*渡された文字列を画面に表示*/
}
int main(void){
    char x[] = "program started.";    /*char 型配列を用意・初期化*/
    char *y = "program is running."; /*自動的に char 型配列を用意*/
    message(x);                       /*引数: 配列*/
    message(y);                       /*引数: ポインタ*/
    message("program ended.");        /*引数: 文字列*/
    return 0;
}
```

表示例:

Message: program started.

Message: program is running.

Message: program ended.

文字列関数

文字列を扱う関数 (`string.h` のインクルードが必要)

strcpy 関数 : 文字列をコピーする

- `strcpy(str1, str2)` : 文字列 `str1` に文字列 `str2` をコピーする

```
char Name[] = "WASEDA Taro";    /*配列を宣言・初期化*/
Name[0] = 'T'; Name[2]='K';      /*配列の各要素を変更*/
Name = "OHKUMA Jiro";           /*配列の複数要素に一度に代入できない*/
strcpy(Name, "OHKUMA Jiro");     /*まとめて代入してくれる*/
```

- `str1` の配列サイズが足りない場合は暴走の危険
- 下記のような関数と同じことをしている

```
void mystrcpy (char *x, char *y){
    int i;
    for (i=0; ;i++){              /*ここでは条件をつけない*/
        x[i] = y[i];
        if(y[i]=='\0') break;     /*終端文字を代入したら終了*/
    }
}
```

文字列関数

strlen 関数：文字列の文字数を数える

- strlen(str)：文字列 str の文字数を数える
- 終端文字は文字数にカウントされない

```
char Name[100];  
scanf("%s",x);           /*キーボードから入力*/  
printf("Name length is %d.¥n", strlen(Name)); /*文字数を表示*/
```

strcmp 関数：文字列を比較

- strcmp(str1, str2)：文字列 str1 と str2 を比較
- 同じなら 0，異なる場合は正か負の整数を返す

```
char x[100], y[100];  
scanf("%s",x);           /*キーボードから入力*/  
scanf("%s",y);           /*キーボードから入力*/  
if(strcmp(x,y)==0) printf("Same names. ¥n");
```

- if (x==y) とするとアドレスだけ調べ、中身は無関係

まとめ

- 文字と文字列の概念を理解する
 - ASCII コード
 - 文字 (char) 型
 - 文字列 ("¥0" 終端)
 - 初期化と代入
 - 文字列の引数渡し
- 文字列関数を使う
 - strcpy
 - strlen
 - strcmp