復習

配列変数の要素

```
int x[5];
```

5は配列の要素数

```
x[0]
x[1]
x[2]
x[3]
x[4]
```

これらの変数をそれぞれ
配列の要素と呼ぶ

この数字を配列の添え字，またはインデックスと呼ぶ

！重要！　インデックスの最大値 ＝ 要素数ー1

```
int x = 7;
float aa[x];
```

```
int x = 7;
float aa[7];
```

！重要！　配列宣言時の要素数は定数でなければならない

配列変数の使用例(2)

復習

データの合計値を求める例

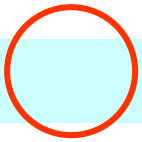
```
int i;
float aa[10];                    // 配列の宣言

// データの入力
for (i = 0; i < 10; i++)
{
    printf("値を入力してください：");
    scanf("%f", &aa[i]);
}

// データの合計を計算
float sum = 0;
for (i = 0; i < 10; i++)
{
    printf("%d番目のデータは%f¥n", i, aa[i]);
    sum = sum + aa[i];
}
printf("合計値は%fである¥n", sum);
```

```
値を入力してください：3.1
値を入力してください：1.4
値を入力してください：2.5
値を入力してください：9.9
値を入力してください：10.2
値を入力してください：1.3
値を入力してください：1.4
値を入力してください：0.22
値を入力してください：5.3
値を入力してください：6.2
0番目のデータは3.100000
1番目のデータは1.400000
2番目のデータは2.500000
3番目のデータは9.900000
4番目のデータは10.200000
5番目のデータは1.300000
6番目のデータは1.400000
7番目のデータは0.220000
8番目のデータは5.300000
9番目のデータは6.200000
合計値は41.519997である
続行するには何か ．．．
```

復習

配列変数の初期化(2)

```
int c[] = {3, 4, 5, 100, 200, 300};
```



要素数の指定が無い！ → 自動的に設定される

```
int c[6] = {3, 4, 5, 100, 200, 300};
```

これと同じ意味

```
int b[5];
b[5] = {10, 20, 25, 35, 40};

int c[7];
c[] = {3, 4, 5, 8, 100, 200, 300};
```

初期化は配列変数の宣言と同時でなければならない.

宣言文と代入文の二つの文に別れてはいけない

いろいろな変数型(2)

復習
第2回講習資料

char

0～255（符号なし）または-128～+127（符号あり）

1バイト　→　英数字1文字を入れるのにぴったり

アスキーコード　→　文字と文字列で学習

int

4バイト　もっとも標準的な整数型

float

2進法で10進実数を表わすので誤差がある
（有効数字8桁程度）

4バイト　単精度実数型(単精度浮動小数点型)

double

floatより高精度(有効数字15桁程度)

8バイト　倍精度実数型(倍精度浮動小数点型)

アスキーコード表
(ASCII code)

日本語は通常2バイトの文字コード．JISコード，シフトJISコード，Unicode (UTF-8)等の様々な文字コードがある．

漢字変換無しでキーボードから直接入力できる半角文字

文字コード

1文字＝1バイト
⇒ char型変数

アスキーコード(値)											
32		48	0	64	@	80	P	96	`	112	p
33	!	49	1	65	A	81	Q	97	a	113	q
34	”	50	2	66	B	82	R	98	b	114	r
35	#	51	3	67	C	83	S	99	c	115	s
36	$	52	4	68	D	84	T	100	d	116	t
37	%	53	5	69	E	85	U	101	e	117	u
38	&	54	6	70	F	86	V	102	f	118	v
39	’	55	7	71	G	87	W	103	g	119	w
40	(	56	8	72	H	88	X	104	h	120	x
41	)	57	9	73	I	89	Y	105	i	121	y
42	*	58	:	74	J	90	Z	106	j	122	z
43	+	59	;	75	K	91	[	107	k	123	{
44	,	60	<	76	L	92	¥	108	l	124	|
45	-	61	=	77	M	93	]	109	m	125	}
46	.	62	>	78	N	94	^	110	n	126	~
47	/	63	?	79	O	95	_	111	o	127	

この授業では1バイトのアスキーコードのみを学習する

この授業では日本語文字のプログラムは扱わない

C言語における文字の取り扱い(1)

文字　a　A　b　B　z　Z　0　1　9　=　+　?　/　!　など

キーボードから直接入力できる半角文字

英数字記号　→　ASCIIコード(文字コード)で番号付け

ASCIIコードは32～126までの番号(コード表参照)

→　1バイトで表せる

→　char型変数を利用

```
char moji1, moji2, xx, yy;

moji1 = 65;

moji2 = 'A';

xx = '0';

yy = 0;
```

char型変数moji1に65を代入

大文字AのASCIIコード 65を代入

シングルクォートで文字と囲むとその文字の文字コードになる

文字'0'のASCIIコード 48を代入

数値0を代入

C言語における文字の取り扱い(2)

文字の出力

```
char moji1, moji2, xx;
moji1 = 65;
moji2 = 'A';
xx = '0';
printf("moji1は%d, moji2は%d, xxは%dである. ¥n", moji1, moji2, xx);
printf("moji1は%c, moji2は%c, xxは%cである. ¥n", moji1, moji2, xx);
```

```
moji1は65, moji2は65, xxは48である.
moji1はA, moji2はA, xxは0である.
```

変換文字%cは文字コードを文字に変換する

文字の入力

```
char aa;
printf("文字を一つ入れてください:");
scanf("%c", &aa);
printf("ASCIIコードは%d, 文字は%cである. ¥n", aa, aa);
```

変換文字%cは入力された文字の文字コードを変数に代入する

```
文字を一つ入れてください:1
ASCIIコードは49, 文字は1である.
```

C言語における文字列の取り扱い(1)

文字

a　A　b　B　z　Z　0　1　9　=　+　?　/　!　など

文字列

文字列は文字の集合

Hello　Kandai　programming　など

理由：文字数より要素数が多い配列を用いた時に，文字列の最後を示すため

原則：C言語では文字列をchar型配列で扱う

文字列Helloを文字配列s[6]に入れる場合

s[0]	s[1]	s[2]	s[3]	s[4]	s[5]
↑	↑	↑	↑	↑	↑
'H'	'e'	'l'	'l'	'o'	
72	101	108	108	111	0

必ず最後にはコード0を入れる

文字数プラス1の要素数が必要

文字列(文字配列)の初期化

s[6]～s[9]は現在使っていないという目印

文字列の設定

必要な文字数より多めに宣言する

最後に0を入れる

```
char s[10];
s[0] = 'H'; s[1] = 'e'; s[2] = 'l'; s[3] = 'l'; s[4] = 'o'; s[5] = 0;
```

配列 s[10] ⇒

H	e	l	l	o	0	%	&	*	#

ゴミ！　初期化されてない不定な値

```
char s[10] = {'H', 'e', 'l', 'l', 'o', 0};
```

配列初期化の応用

It's New!

```
char s[10] = "Hello";
```

配列宣言と同時に文字列を設定する方法
（最後の0も自動的に入る）

```
char s[] = "Hello";
```

char s[6] = "Hello" と同じ

```
char s[10];
s[10] = "Hello";
s[0] = "Hello";
s[ ] = "Hello";
```

配列宣言と同時でなくてはダメ！
（宣言文と代入文に分かれてはダメ）

文字列の入出力

aa[0], aa[1], ・・・
⇒インデックスを付けると一つ一つの文字を表す
aa
⇒ インデックスを付けないときはかたまりとしての文字列を表す

文字列の出力

```
char aa[] = "KANDAI";
printf("私は%s生です. ¥n", aa);
```

変換文字は%s

注意！ [ ]を付けない

```
私はKANDAI生です.
```

文字列の入力

必要な文字数より多めに宣言

注意！ &も[ ]も付けない

```
char aa[100];
printf("99文字以下で文字列を入力してください:");
scanf("%s", aa);
printf("文字列は%sである. ¥n", aa);
```

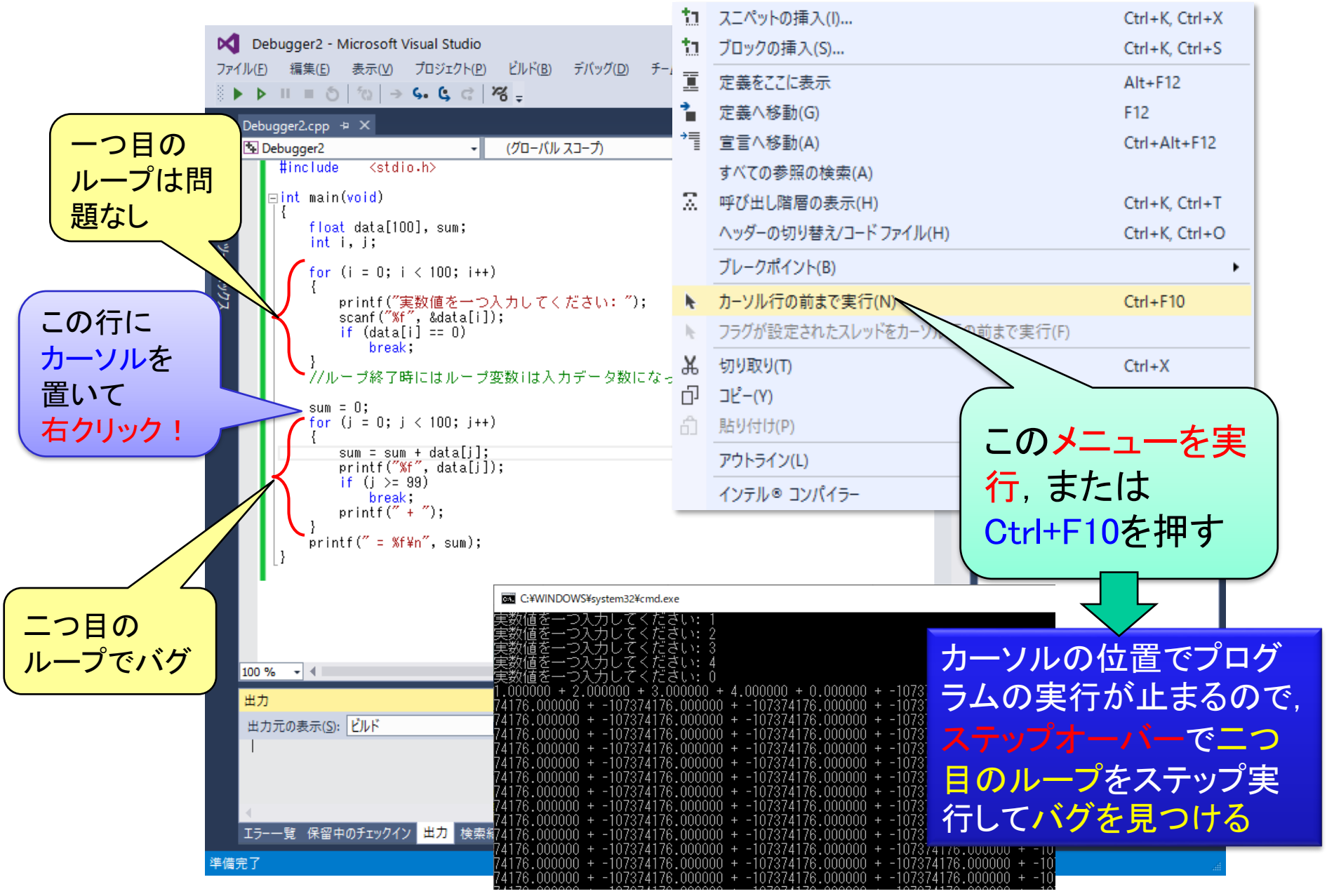
最後の0も自動的に入る

文字配列aaに入力した文字列が代入される

変換文字は%s

```
99文字以下で文字列を入力してください:Kandai
文字列はKandaiである.
```

デバッガの使用法(2)


一つ目のループは問題なし
この行にカーソルを置いて右クリック！
二つ目のループでバグ
このメニューを実行，またはCtrl+F10を押す
カーソルの位置でプログラムの実行が止まるので，ステップオーバーで二つ目のループをステップ実行してバグを見つける
Debugger2 - Microsoft Visual Studio
Debugger2.cpp
#include <stdio.h>
int main(void)
{
float data[100], sum;
int i, j;
for (i = 0; i < 100; i++)
{
printf("実数値を一つ入力してください: ");
scanf("%f", &data[i]);
if (data[i] == 0)
break;
}
//ループ終了時にはループ変数iは入力データ数になっ
sum = 0;
for (j = 0; j < 100; j++)
{
sum = sum + data[j];
printf("%f", data[j]);
if (j >= 99)
break;
printf(" + ");
}
printf(" = %f¥n", sum);
}
スニペットの挿入(I)... Ctrl+K, Ctrl+X
ブロックの挿入(S)... Ctrl+K, Ctrl+S
定義をここに表示 Alt+F12
定義へ移動(G) F12
宣言へ移動(A) Ctrl+Alt+F12
すべての参照の検索(A)
呼び出し階層の表示(H) Ctrl+K, Ctrl+T
ヘッダーの切り替え/コード ファイル(H) Ctrl+K, Ctrl+O
ブレークポイント(B)
カーソル行の前まで実行(N) Ctrl+F10
切り取り(T) Ctrl+X
コピー(Y)
貼り付け(P)
アウトライン(L)
インテル® コンパイラー
C:¥WINDOWS¥system32¥cmd.exe
実数値を一つ入力してください: 1
実数値を一つ入力してください: 2
実数値を一つ入力してください: 3
実数値を一つ入力してください: 4
実数値を一つ入力してください: 0
出力
出力元の表示(S): ビルド
エラー一覧 保留中のチェックイン 出力
準備完了