|  |
| --- |
| システムスリーテン株式会社 |
| C言語テキスト補助資料 |
|  |

目次

[１．プログラムとコンパイル 2](#_Toc16880093)

[２．データ型の範囲とサイズ 3](#_Toc16880094)

[３．出力フォーマット 4](#_Toc16880095)

[４．入力フォーマット 6](#_Toc16880096)

[5．If文 7](#_Toc16880097)

[6．ループ（繰返し処理） 9](#_Toc16880098)

[7．文字列とアドレス 11](#_Toc16880099)

[8．ポインタ 12](#_Toc16880100)

[9．前置形式と後置形式の違い 14](#_Toc16880101)

[10．関数プロタイプ宣言 15](#_Toc16880102)

[11．ローカル変数、グローバル変数、スタテック変数 16](#_Toc16880103)

[12．Extern修飾子 17](#_Toc16880104)

[13．define 18](#_Toc16880105)

[14．コメント 19](#_Toc16880106)

[15．推奨コードscanf\_s 20](#_Toc16880107)

[16．推奨コードfopen\_s 21](#_Toc16880108)

[17．文字列を扱う関数 22](#_Toc16880109)

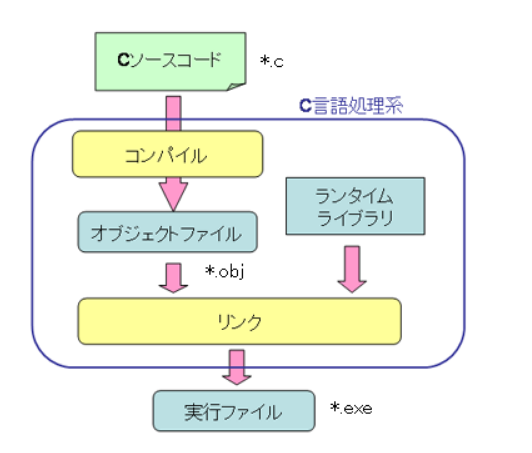
[18．atoi(str)、atof(str) 27](#_Toc16880110)

[19．分布グラフ 29](#_Toc16880111)

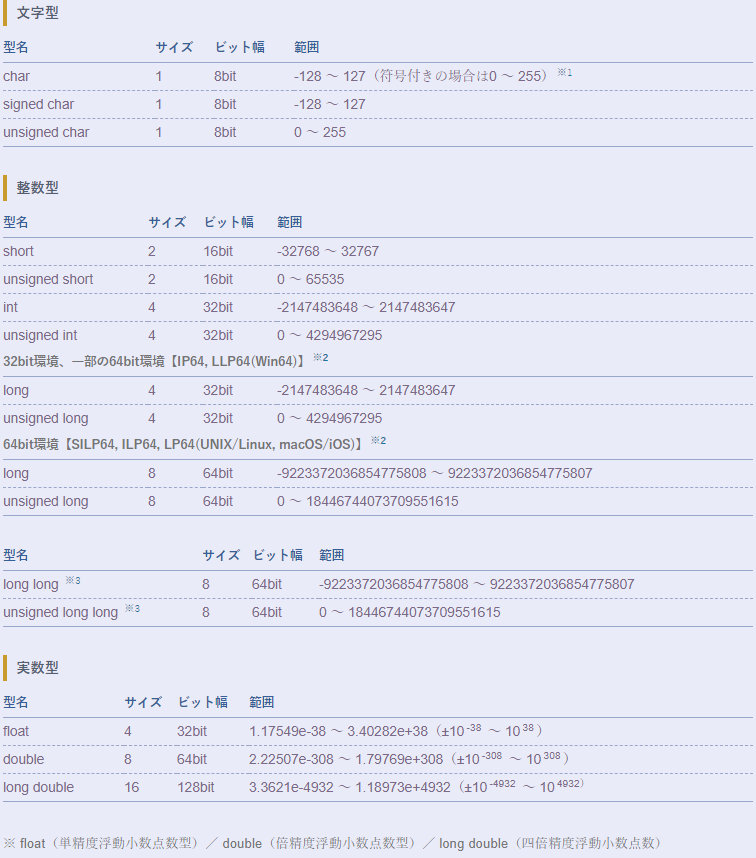
[20．ソート 31](#_Toc16880112)

[21．ビット演算子 32](#_Toc16880113)

# １．プログラムとコンパイル

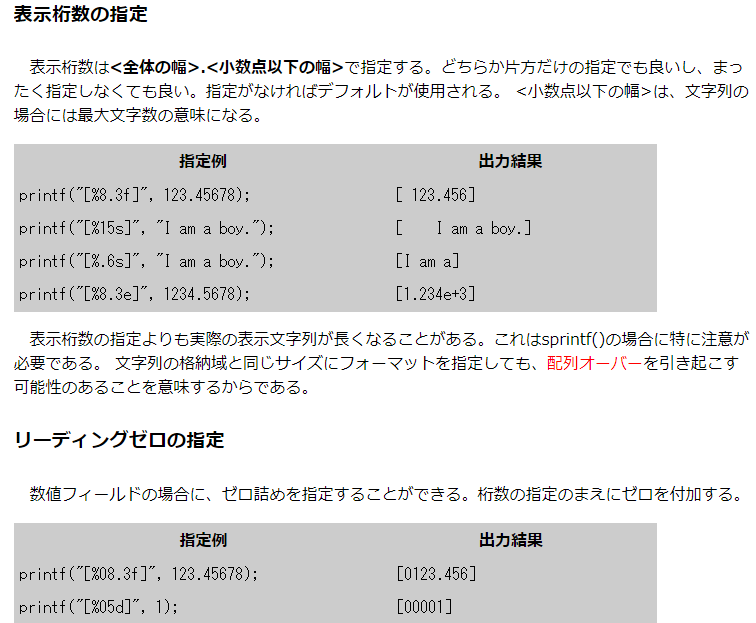


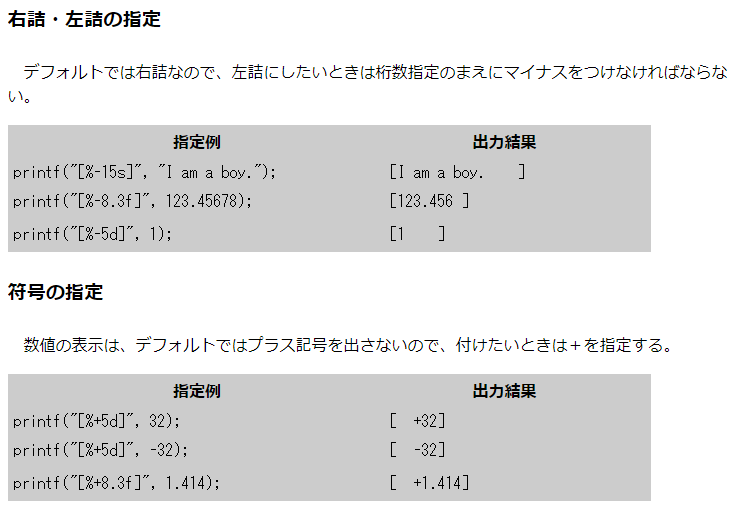
# ２．データ型の範囲とサイズ



# ３．出力フォーマット



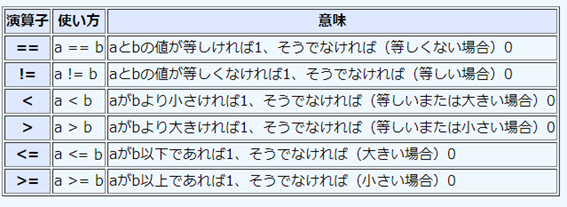


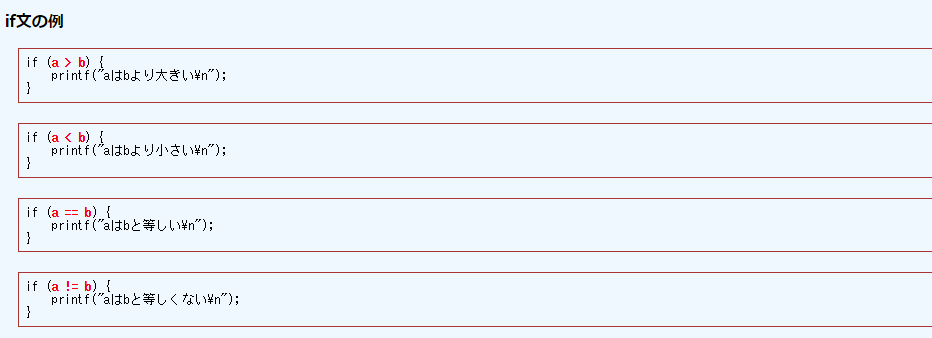


# ４．入力フォーマット

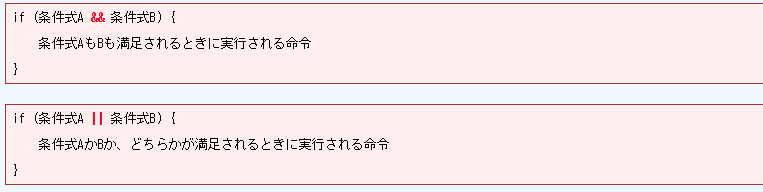


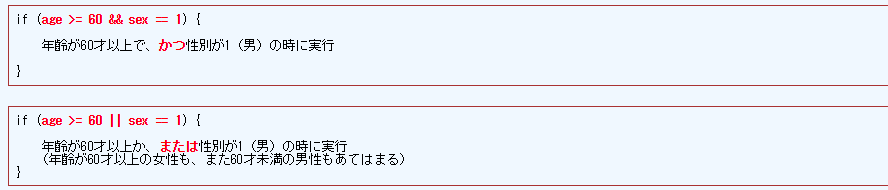
# 5．If文

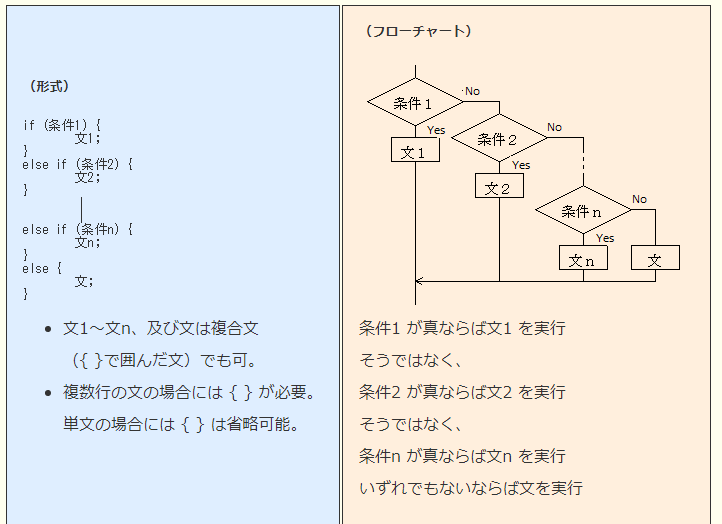




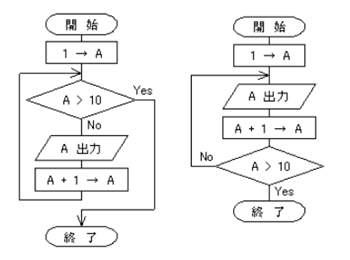
複数の条件式



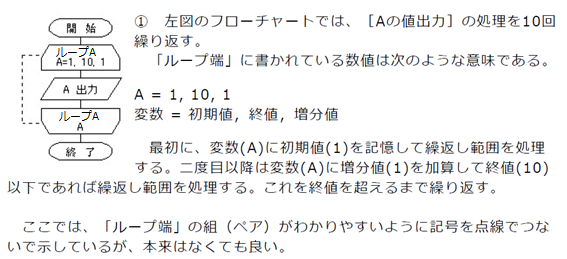


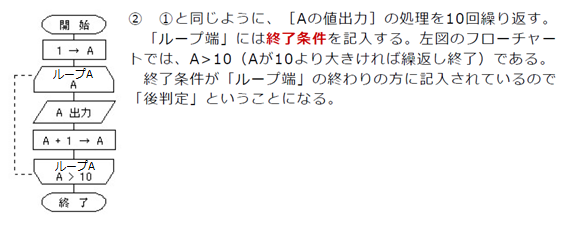


# 6．ループ（繰返し処理）



1. 前判定　　　　　　　　②後判定





初期値　 A = 1

終了条件式　A > 10　継続条件式　A <= 10

増分　　A = A + 1; 別表記　A++;

・for文　前判定

for ( A = 1 ; A <= 10 ; A++ ) {

printf ( “%d “, A );

}

・while文 前判定

A=1;

while ( A<=10) {

printf ( “%d “, A );

A++;

}

・do-while文　後判定

A=1;

do {

printf ( “%d “, A );

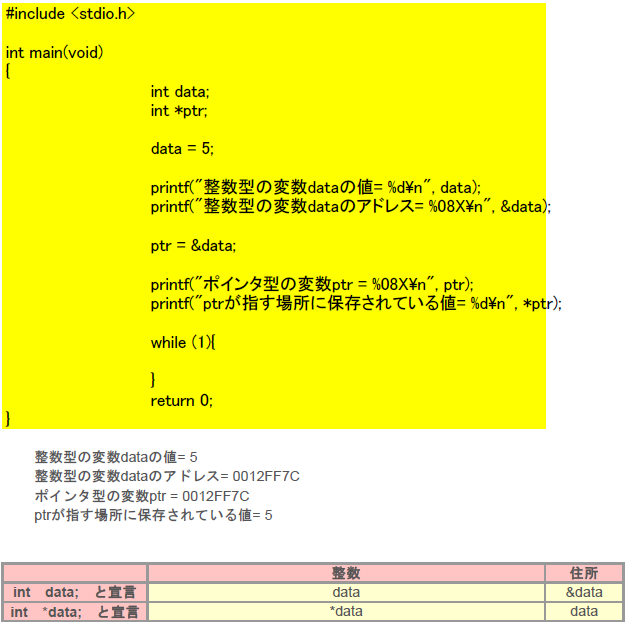
A++;

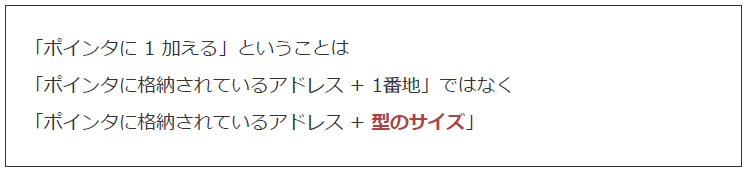
} while ( A<=10);

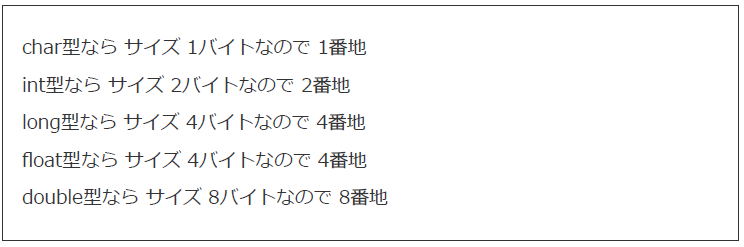
# 7．文字列とアドレス

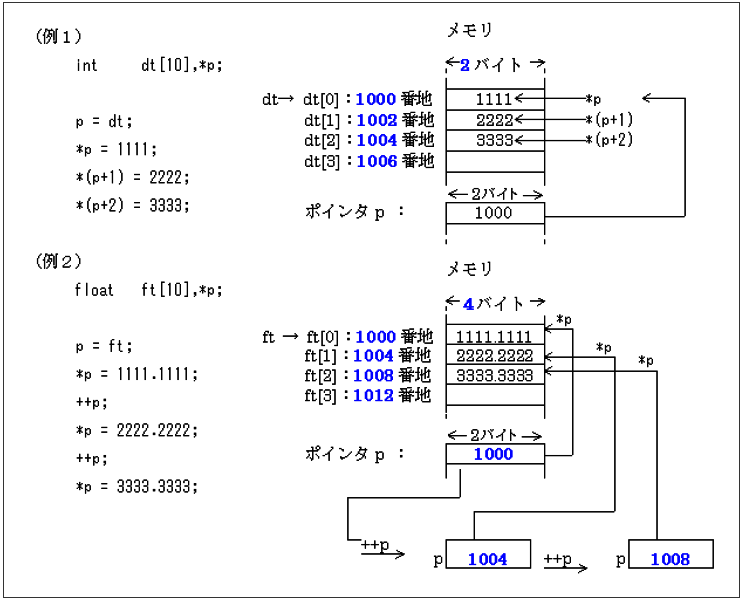


# 8．ポインタ

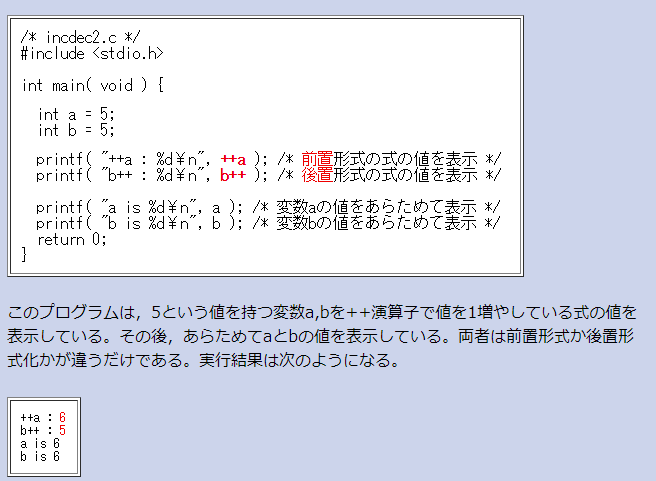




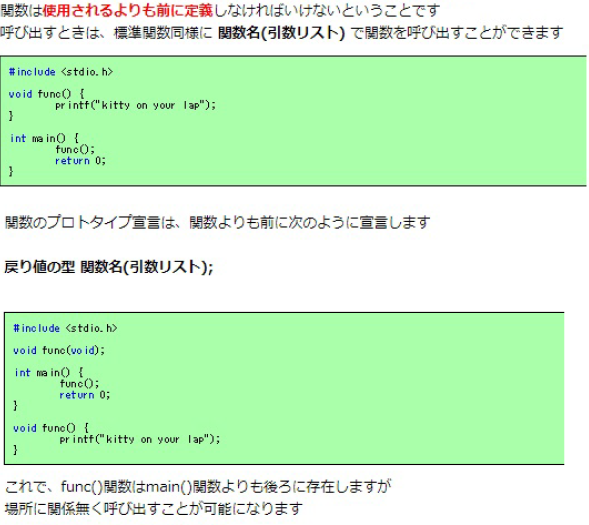




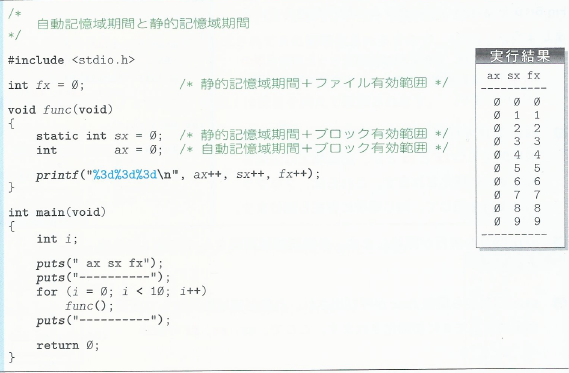
# 9．前置形式と後置形式の違い



# 10．関数プロタイプ宣言

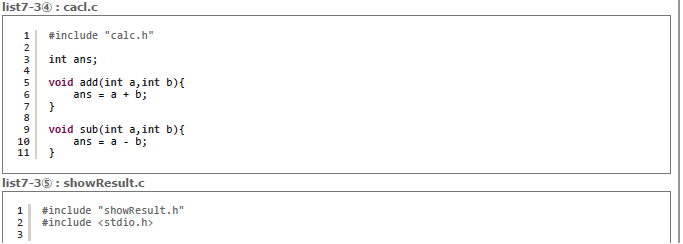


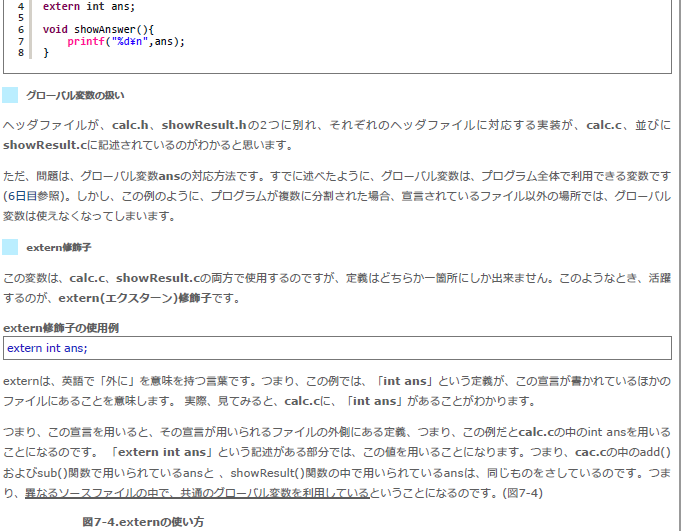
# 11．ローカル変数、グローバル変数、スタテック変数



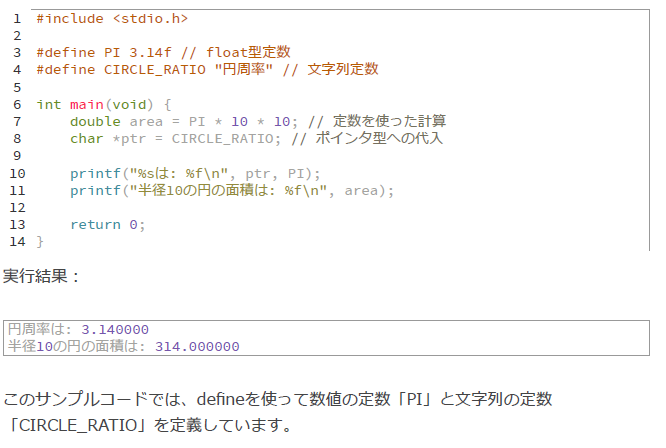
ローカル変数　　ax、グローバル変数　fx、スタテック変数 sx

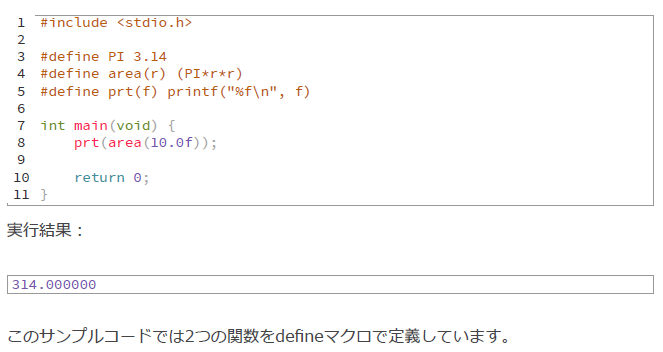
# 12．Extern修飾子





# 13．define





# 14．コメント

// 単一行コメント

/\*

複数行コメント

複数行コメント

\*/

関数ヘッダーコメント例

//

//-----------------------------------------------------------------------------------------------------------------

//

// 労務費昇順出力 int roumusyoujun(void)

//

// 　引数　　　　　無し

// 戻り値　　　　正常：SEIJYOU　異常：IJYOU

//

// 入力エリア　　char kojindata[130][10][20]　個人リスト格納エリア

// int kojingyo　　　　　　　　 行数

// int keisan[130][20]　　　　　計算結果の格納

//

// 出力ファイル　　労務費昇順.txt

//

//-----------------------------------------------------------------------------------------------------------------

//

# 15．推奨コードscanf\_s

scanf　は　scanf\_s　を使用してください。

　　使用例）

　　 n = scanf\_s("%d", &s);

　　 if (n != 1){

　　 return -1;

　　　 }

セキュリティが強化された scanf\_s() 命令です。

scanf\_s では %s で受け取る領域のサイズを指定します。

char file[64];

printf("入力ファイル名をタイプして下さい\n");

scanf\_s("%s",file,64);

%d などはサイズが決まっているのでサイズの指定はありません。

# 16．推奨コードfopen\_s

fopen　は　 fopen\_s　を使用してください。

　　使用例）

#include<stdio.h>

/\*ファイル名を実行時にキー入力する形に書き直そう\*/

char inputFileName[100] = "c:\\workc\\kadai7.c";/\*元のcプログラムファイル\*/

char outputFileName[100] = "c:\\workc\\kadai7.html";/\*作るHTMLファイルの名前\*/

int main(void)

{

/\*FILEはバッファを含むストリーム管理領域。\*/

/\*ファイル処理関数はこの領域を示すポインタを必要とする\*/

FILE \*input, \*output;

errno\_t err;

int c;

/\*------------ファイルを開く------------\*/

err = fopen\_s(&input, inputFileName, "r");/\*名前がinputFileNameで読み込み専用ファイル\*/

if (err != 0){

printf("入力ファイルエラー");

return(-1);

}

err = fopen\_s(&output, outputFileName, "w");/\*名前がoutputFileNameで書き込み専用ファイル\*/

if (err != 0){

printf("出力ファイルエラー");

return(-1);

}

/\*------------ファイルの読み書き------------\*/

fprintf(output, "<HTML><HEAD></HEAD><BODY><PRE>\n");

/\*inputから一文字読みEOFでなければoutputに書き出す\*/

while (EOF != (c = getc(input)))

putc(c, output);

fprintf(output, "\n</PRE></BODY></HTML>\n");

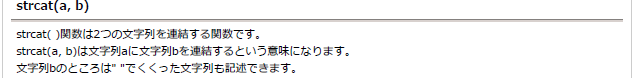
/\*------------ファイルを閉じる------------\*/

fclose(input);

fclose(output);

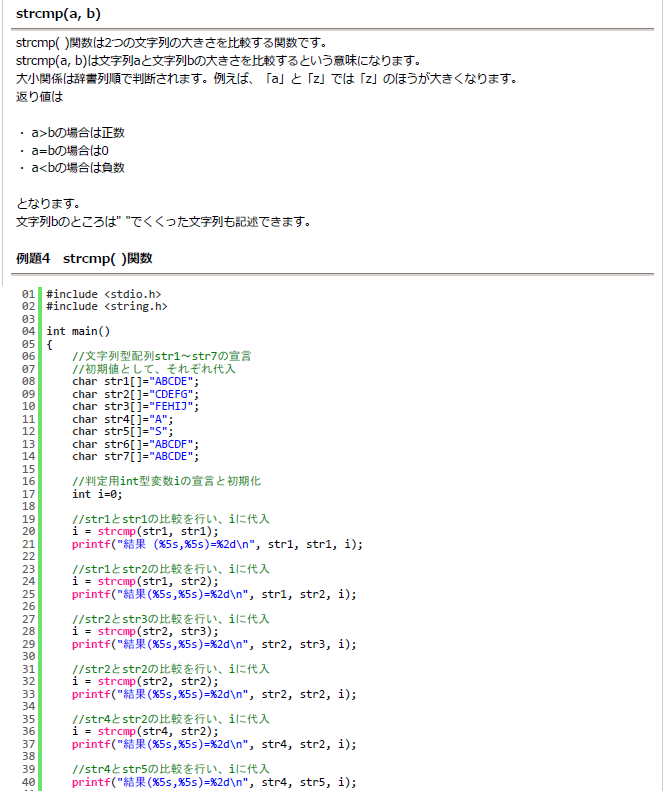
return(0);

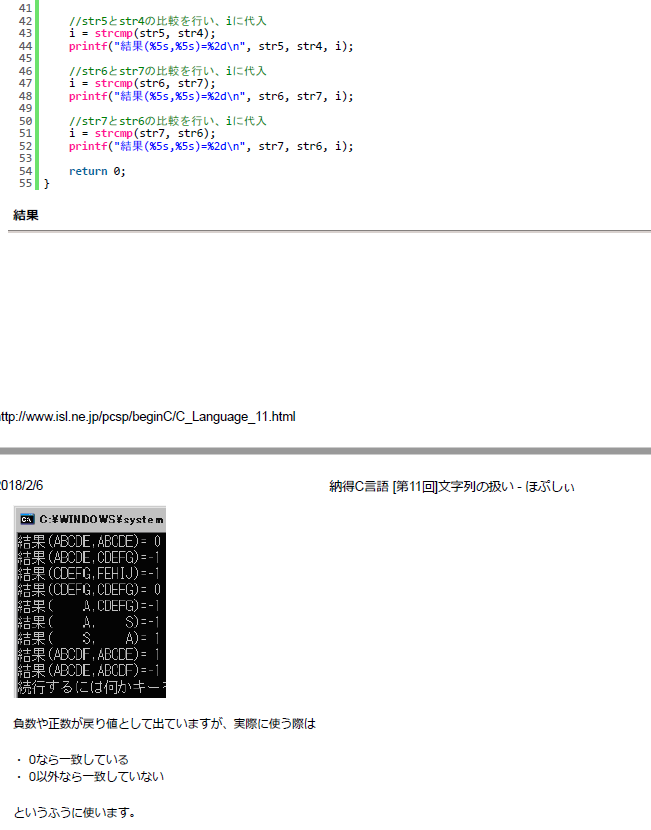
# 17．文字列を扱う関数





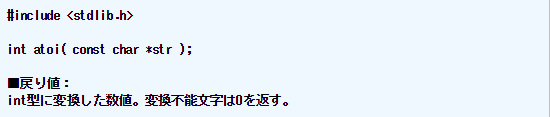


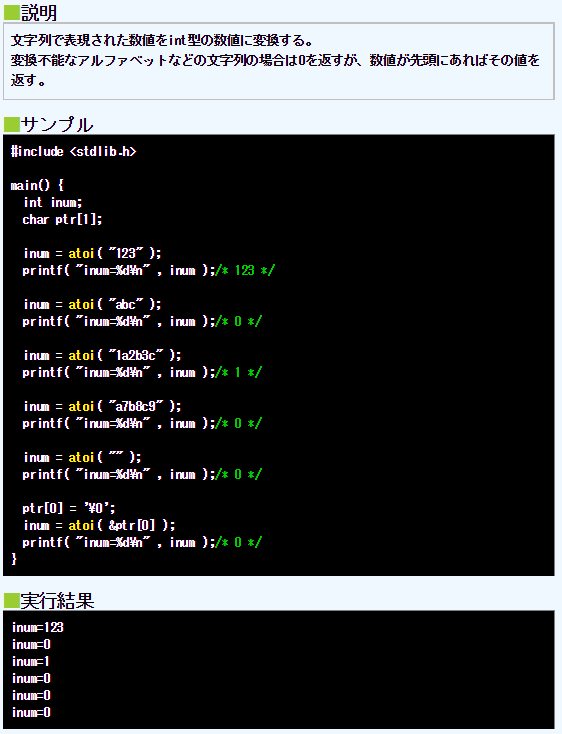


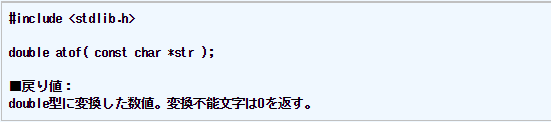


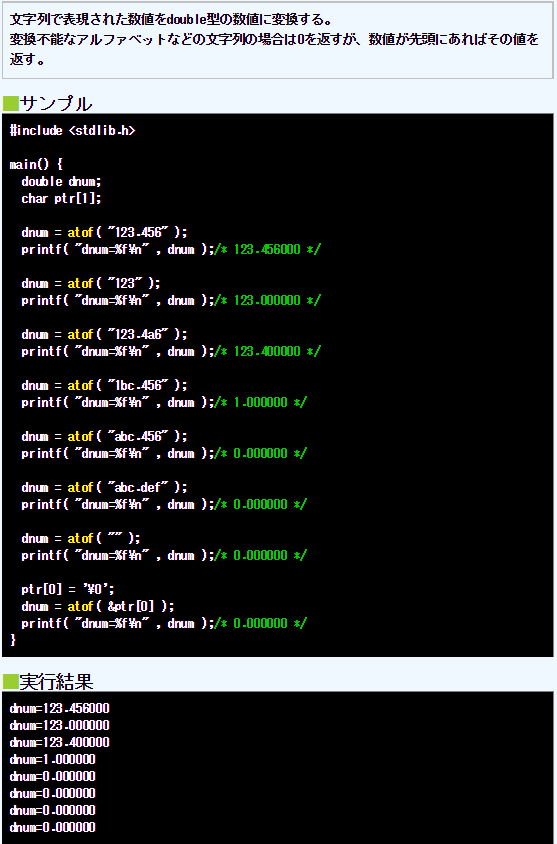


# 18．atoi(str)、atof(str)

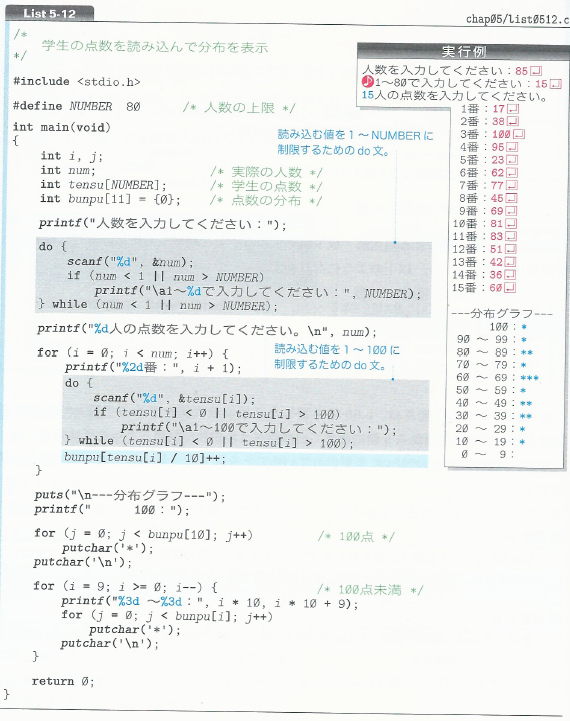


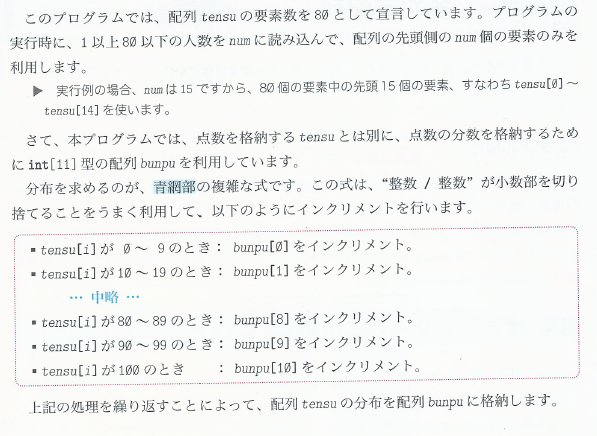




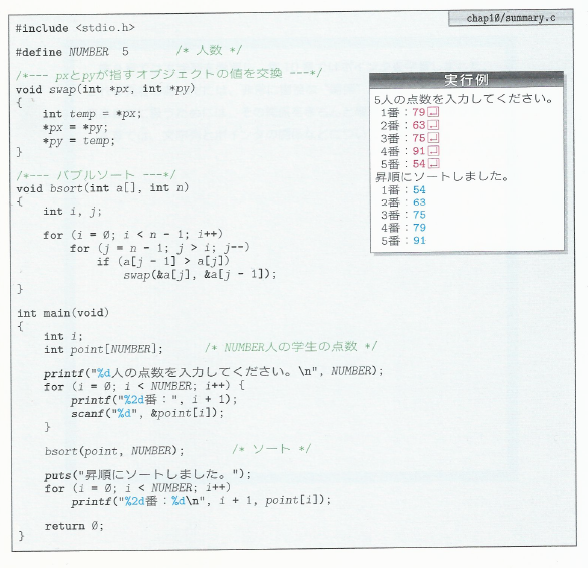


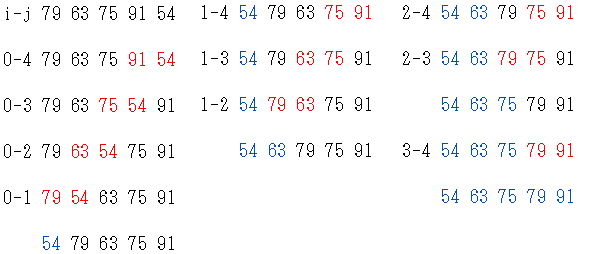
# 19．分布グラフ





# 20．ソート





# 21．ビット演算子

