練習問題 1
次の関数を作成し、動作を確かめよ。但し、array_swap は array_copy を用いて作成すること。

関数名： array_copy
引数： 整数型配列の配列名 old[100] と new[100]
戻り値： 無し
副作用： old の成分を new の成分にコピーする

関数名： array_swap
引数： 整数型配列の配列名 a[100] と b[100]
戻り値： 無し
副作用： a の成分と b の成分を入れ替える

練習問題 2
上の関数 array_swap を用いて、２ｘ１００の整数型二次元配列 a の一行目
a[0][0],a[0][1],...,a[0][99]
と、二行目
a[1][0],a[1][1],...,a[1][99]
を入れ替えるプログラムを作成せよ

練習問題 3
array_swap をサイズ 40 の配列に対して利用できるよう改造せよ。
また、この改造された array_swap を利用して以下の関数を作成せよ。

関数名： array_sort
引数： 整数型配列の２次元配列名 a[40][40]
戻り値： 無し
副作用： 配列 a の各行の第 0 成分を見比べて、行ごとにソートする。

例）
実行前
1 3 7 54 43 889 2 ....
5 7 23 9 43 765 5 ....

4 6 9 2 34 76 3234 ...
: : :
: : :
: : :

実行後
1 3 7 54 43 889 2 ....
4 6 9 2 34 76 3234 ...
5 7 23 9 43 765 5 ....
: : :
: : :
: : :

練習問題 4
以下の関数及び動作確認プログラムを作成せよ。
取り扱う文字型配列に関しては、必ず末尾に'¥0'が
あることを仮定して良い。

関数名： array_sort
引数： 一次元の整数型配列の配列名 a[100]とソートの方向を表す整数 n（1 か-1）
戻り値： 無し
副作用： n=1 の場合、a の成分を昇順（小さい順）にソートし、n=-1 の場合、降順（大きい順）にソートして、結果を再び a に代入する。

関数名： array_pop
引数： 実数型配列の配列名 a[100]
戻り値： 配列 a の第 0 成分
副作用: 第 1 成分から最終成分まで逆シフト（a[1] -> a[0], a[2] -> a[1], ... ）

関数名： array_push
引数： 実数 x と実数型配列の配列名 a[100]
戻り値： なし
副作用: a の最終成分は消滅。第 0 成分から(最終-1)成分までシフト（a[98] -> a[99], a[97] -> a[98], ... ）し、第 0 成分には x を代入。

関数名： string_length
引数： 文字型配列の配列名 a[]（配列のサイズは書かない）
戻り値： 配列 a の長さ（先頭文字から'¥0'の手前までの成分の個数）
副作用： 無し

関数名： string_cat
引数： 文字型配列の配列名 old[]と文字型配列の配列名 new[]
戻り値： 無し（'¥0'の無い配列に関しては、考慮しなくて良い）
副作用： new の末尾に old を追加する。ただし、単なる追加ではなく、
new の成分である文字列（第 0 成分から'¥0'の手前まで）
の直ぐ後に old の文字列（第 0 成分から'¥0'の手前まで）
を連結させ、末尾には'¥0'をつけること。

注） old に含まれる文字列の長さ＋new に含まれる文字列の長さ ＜ new のサイズ
と考えて良い。

関数名： array_sum
引数： 一次元の実数型配列の配列名とそのサイズ
戻り値： 配列の各成分の合計
副作用： 無し

関数名： string_copy
引数： 文字型配列の配列名 old[] と new[]
戻り値： 無し
副作用： old の成分を new の成分にコピーする。
注）new のサイズは old のそれより大きいと思って良い。

関数名： string_swap
引数： 文字型配列の配列名 a[] と b[]
戻り値： 無し
副作用： a の成分と b の成分を入れ替える
注）a の成分数は b のそれと同じと思って良い。

関数名： check_length
引数： 無し
戻り値： キーボードから読み込まれた文字列（文字の配列）a の長さ

（先頭文字から'¥0'の手前までの成分の個数）
副作用： キーボードから文字列を入力することを促す

関数名： string_char
引数： 文字型配列の配列名 s[]と t[]
戻り値： s[]の中に文字 t[0]があればその位置（配列 s の第何成分か）を返す
無ければ-1. 双方とも整数値
副作用： 無し

関数名： string_cmp
引数： 文字型配列の配列名 s[]と t[]、整数値 n
戻り値： 文字列 s[]と文字列 t[]の最初の n 文字を比較して一致すれば 0、
一致しなければ 1
副作用： 無し

関数名： string_search
引数： 文字型配列の配列名 s[]と t[]
戻り値： s[]の中に文字列 t[]があれば 1、無ければ 0
副作用： 無し

練習問題 5 （チャレンジ）
５０桁の１０進整数２個の四則演算をするプログラム

練習問題 6
以下の関数を作成せよ。取り扱う文字型配列に関しては、必ず末尾に'¥0'が
あることを仮定して良い。

(1)
関数名： distail
引数： 文字型配列の配列名 s[]
戻り値： 無し
副作用： s[]に含まれる文字列の最後尾と'¥0'までの間にスペースがあれば
取り除く

(2)
関数名： dishead
引数： 文字型配列の配列名 s[]
戻り値： 無し
副作用： s[]に含まれる文字列の先頭にスペースがあれば取り除く

(3)
関数名： itoc
引数： 0から9までの整数
戻り値： 0から9までの文字（引数を文字に変換する）
副作用： なし

(4)
関数名： itos
引数： 整数値と文字型配列名a[]
戻り値： 無し
副作用： 引数の１０進整数値を文字列に変換してaに格納する

(5)
関数名： stoi
引数： 文字型配列名a[5]
戻り値： a の成分である0から9までの文字を１０進整数に変換した値
副作用： aの成分に0～9以外の文字があった場合、Error と表示

(6)
関数名： string_cat2
引数： 文字型配列の配列名old[]と文字型配列の配列名new[]
戻り値： 無し（'¥0'の無い配列に関しては、考慮しなくて良い）
副作用： newの末尾からスペースを取り除いて、その後でoldを追加する。
末尾には'¥0'をつけること。

(7)
関数名：string_cat3
引数 ：文字型配列a[100], 文字型配列b[100], 整数値n, 整数値m
戻り値：実際に追加した文字数
副作用：配列a に, 配列b の先頭からn 文字目から最大m 文字分を追加する.

n+m が配列 b の長さよりも大きい場合は，配列 b の長さ分だけ追加し，
エラーを出さないようにする．
ex.） string_cat2(a,b,3,2)； → a に b の第 3 成分から 2 文字分，すなわち
b の 3,4 成分を追加

(8)

関数名：string_char2
引数 ：文字型配列 a[100]，文字型変数 t
戻り値：配列 a の中に含まれる t の最後の位置（1～99）．または，含まれなければ 0 を返す．
副作用：無し．

練習問題 7（チャレンジ）
以下の関数を作成せよ。

関数名：time_calc
引数 ：文字型配列 a，数値 n
戻り値：無し．
副作用：00:00 の形式で文字入力された現在時刻から n 分後は何時何分かを計算し，00:00 の形式で表示する．ただし，n の値域は負数も含む．