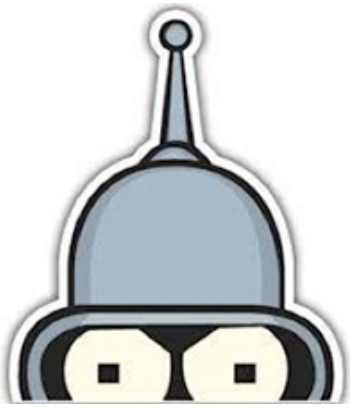
B1 - Elementary Programming in C

B-CPE-110

Push Swap

Fancy List Sorting



1.0

{EPITECH.}

Push Swap

binary name:	push_swap
group size:	1
repository name:	CPE_pushswap_$ACADEMICYEAR
repository rights:	ramassage-tek
language:	C
compilation:	via Makefile, including re, clean and fclean rules

- Your repository must contain the totality of your source files, but no useless files (binary, temp files, obj files,...).
- All the bonus files (including a potential specific Makefile) should be in a directory named *bonus*.
- Error messages have to be written on the error output, and the program should then exit with the 84 error code (0 if there is no error).

Authorized Functions

The only system calls allowed are the following ones:

- write
- malloc
- free

THE PROJECT

The game is made up of two lists of numbers named *l_a* and *l_b*.
In the beginning, *l_b* will be empty and *l_a* will contain a certain amount of positive or negative numbers.
The objective of the game is to sort *l_a*.

In order to accomplish this, you will only have access to the following operation:

- **sa**
  swap the first two elements of *l_a* (nothing will happen if there aren't enough elements).
- **sb**
  swap the first two elements of *l_b* (nothing will happen if there aren't enough elements).
- **sc**
  sa and **sb** at the same time.
- **pa**
  take the first element from *l_b* and move it to the first position on the *l_a* list (nothing will happen if *l_b* is empty).
- **pb**
  take the first element from *l_a* and move it to the first position on the *l_b* list (nothing will happen if *l_a* is empty).
- **ra**
  rotate *l_a* toward the beginning, the first element will become the last.
- **rb**
  rotate *l_b* toward the beginning, the first element will become the last.
- **rr**
  ra and *rb* at the same time.
- **rra**
  rotate *l_a* toward the end, the last element will become the first.
- **rrb**
  rotate *l_b* toward the end, the last element will become the first.
- *rrr*
  rra and **rrb** at the same time.

You must create a program in which *l_a* is given as parameter (all numbers are valid and can fit in an integer).
The goal is to sort the list by using the fewest possible operations.
The program must print the series of operations that enable this list to be sorted.

The operations must be displayed separated by a space. No spaces should be at the beginning nor at the end. The operations' list must be followed by a \n.

You could add some extra features (considered as bonus); for example, adding the following options:

- **-v** shows the statuses of *l_a* and *l_b* at each step.
- **-vT** the same as the above text, but using the libncurses

EXAMPLES

Let *l_a* contain `2 1 3 6 5 8` and *l_b* be empty.
Here are the results of some operations (each step is done after the previous ones):

- **sa**
  l_a 1 2 3 6 5 8
  l_b
- **pb pb pb**
  l_a 6 5 8
  l_b 3 2 1
- **ra rb** (or simply **rr**)
  l_a 5 8 6
  l_b 2 1 3
- **rra rrb** (or simply **rrr**)
  l_a 6 5 8
  l_b 3 2 1
- **sa**
  l_a 5 6 8
  l_b 3 2 1
- **pa pa pa**
  l_a 1 2 3 5 6 8
  l_b

Terminal

```
~/B-CPE-110> ./push_swap 2 1 3 6 5 8 | cat -e
sa pb pb pb sa pa pa pa$
```

Terminal

```
~/B-CPE-110> ./push_swap 73 79 83 89 97 | cat -e
$
```

Terminal

```
~/B-CPE-110> ./push_swap 1789 | cat -e
$
```