Elementary Programming

push_swap

Teacher in charge
younes2.serraj@epitech.eu
Last update
26/11/2015_18h22

Contents

Administrative details

- Name of the turn-in repository: CPE_year_Pushswap
  `Example for the school year 2015-2016 : CPE_2015_Pushswap`
- Binary name : push_swap
- Your Makefile must be located at the root of your repository.
- The executable file created must be located at the root of your repository.

`Be careful: the norm will be checked for every single file you turn in`

Game description

The game is made of two lists of numbers named l_a and l_b. At the beginning, l_b is empty and l_a contains a certain amount of positive or negative numbers (no duplicates). The goal of the game is to make it so that l_a contains the same numbers but sorted in an ascending order.

To do so, you only have the following operations at your disposal:

- `sa` : swaps the first 2 elements of l_a
  (does nothing if there aren't enough elements).
- `sb` : swaps the first 2 elements of l_b
  (does nothing if there aren't enough elements).
- `ss` : sa and sb at the same time.
- `pa` : takes the first element of l_b and puts it in the first position in l_a.
  (does nothing if l_b is empty).
- `pb` : takes the first element of l_a and puts it in the first position in l_b.
  (does nothing if l_a is empty).
- `ra` : rotates l_a
  (towards the start, the first element becomes the last one).
- `rb` : rotates l_b
  (towards the start, the first element becomes the last one).
- `rr` : ra and rb at the same time.
- `rra` : rotates l_a
  (towards the end, the last element becomes the first one).
- `rrb` : rotates l_b
  (towards the end, the last element becomes the first one).
- `rrr` : rra and rrb at the same time.

Example

- In this example, the lists a and b will be as follows:
  `l_a` 2 1 3 6 5 8
  `l_b`

- 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`

The program

You must make a program that takes the list l_a as a list of parameters (no duplicates, all numbers are valid and fit in an integer). The program must display the series of operations allowing to sort this list. Operations are displayed separated by a space, no space at the start nor at the end, all that followed by a '\n'. The goal is to sort the list with the fewest operations possible.

```
1       bash> ./push_swap 2 1 3 6 5 8 | cat -e
2       sa pb pb pb sa pa pa pa$
3       bash>
```

```
1       bash> ./push_swap 73 79 83 89 97 | cat -e
2       $
3       bash>
```

```
1       bash> ./push_swap 1789 | cat -e
2       $
3       bash>
```

Bonus

You can, for instance, add the following options:

- -v : displays the states of l_a and l_b at each step.
- -vt : the same, using tercamps

Allowed functions

- write
- malloc
- free

For the termcaps bonus:

- tgetent
- tgetflag
- tgetnum
- tgetstr
- tgoto
- tputs
- ioctl