```
function! moveblock#mb(choice)
  let s:choices = {
        \'left':['h','hP'],
        \'right':['l','p'],
        \'up':['k',1],
        \'down':['j',1]
  \}
  let s:pre_virtualedit = &virtualedit        "assign value based on test

  set virtualedit=all                         "  * teneary

  "visual! d                                  "  * hash

  let s:next = s:choices[a:choice][0]
  let s:paste_mode= s:choices[a:choice][1]
  return 'd'
      \.s:paste_mode
      \.'gv'.s:next.'o'.s:next.'o'
      "\.':set virtualedit='.s:pre_virtualedit.'\<CR>'
      " use another map to turn off virtualedit
      " use another map to be used as move and recorded
      " in hash
endfunction

function! moveblock#swapline(choice) range
    " if
    let &virtualedit = 'all'
    " get info from visual block
    let s:l_info = [getpos("'<"),getpos("'>")]
                      " left         "right
    " hash data for logic
    let s:choices = {
      \'down':['j','k'],
      \'up':['k','j']
     \}
    "delete line section
    let s:len = s:l_info[1][2] - s:l_info[0][2] + 1
  " height
  let s:height = s:l_info[1][1] - s:l_info[0][1] + 1 " substract between line number
s and offset by 1

  if s:height == 1
     "echo s:len
     call setpos('.', s:l_info[0])
     exe 'normal! d'.s:len.'l'
          \.s:choices[a:choice][0]
          \.'P'
     let s:new_pos = deepcopy(s:l_info[1])
     let s:new_pos[1] += (a:choice ==# 'down')?1:-1
     let s:new_pos[2] += 1
     call setpos('.', s:new_pos)
     exe 'normal! d'.s:len.'l'
        \.s:choices[a:choice][1]
     call setpos('.',s:l_info[0])
     exe 'normal! P'
  else
    exe 'normal! gvd'
    let s:cursor_pos = deepcopy(s:l_info[0])
    let s:cursor_pos[1] = (a:choice==#'down') ?
          \s:cursor_pos[1] + 1 :
          \s:cursor_pos[1] - 1
    call setpos('.',s:cursor_pos)
    exe 'normal! P'

    let s:new_pos = deepcopy(s:l_info[1]) "right visual
    " next row for down | for up
    let s:new_pos[1] = (a:choice==#'down') ?
          \(s:new_pos[1]+1) :
          \deepcopy(s:l_info[0][1]-1)
    let s:new_pos[2] += 1
    call setpos('.', s:new_pos)
    exe 'normal! d'.s:len.'l'
    let s:new_pos2 = deepcopy(s:l_info[0])
```

```
    let s:new_pos2[1] = (a:choice==#'down') ?
          \(s:new_pos2[1]) :
          \deepcopy(s:l_info[1][1])
    call setpos('.',s:new_pos2)
    exe 'normal! P'
  endif
    return 'gv'
      \.s:choices[a:choice][0]
      \.'o'.s:choices[a:choice][0].'o'
endfunction

vnoremap <UP> :call moveblock#swapline('up')<CR>
      \:normal! gvkoko<CR>
vnoremap <DOWN> :call moveblock#swapline('down')<CR>
      \:normal! gvjojo<CR>
vnoremap <expr> <LEFT> moveblock#mb('left')
vnoremap <expr> <RIGHT> moveblock#mb('right')

" vnoremap <UP> :call moveblock#swapline('up')<CR>
"       \:normal! gvkoko<CR>
" vnoremap <DOWN> :call moveblock#swapline('down')<CR>
"       \:normal! gvjojo<CR>                ssssss
" vnoremap <expr> <LEFT> moveblock#mb('left'aaaa
" vnoremap <expr> <RIGHT> moveblock#mb('righaaaa
" sdf                                       aaaa
" sdf                                       )
" sdfdsffbbb                                t')
" dsafdsbdsffbbfbbb
" bbafds
" bbafdsb
" sdffbba          bdsf
" sdb   afdsf
" sdafdsf
" sdafdsf
" sdafdsf
```