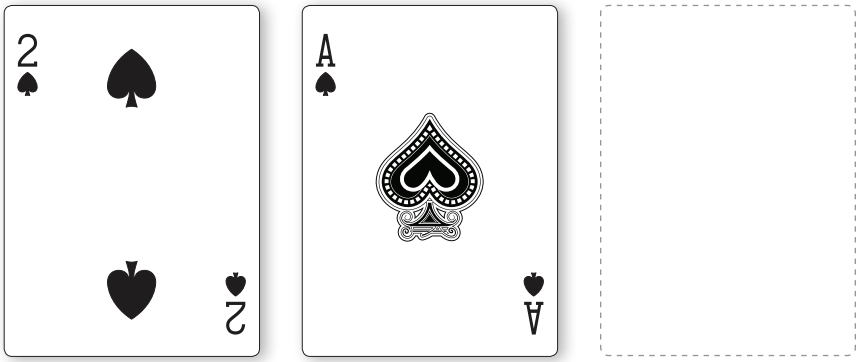
DOI:10.1145/2594482

Peter Winkler

Puzzled
A Sort, of Sorts

Sorting is one of the most fundamental, and most studied, computational tasks. The problem is typically to put n *items in order, using the power of a Turing machine or equivalent. The objective is to minimize time, space, number of comparisons, or (in the parallel case) number of rounds of comparisons. But you do not need the full power of a Turing machine to do it. How meager can your computing resources be to be able to sort? Would you believe three LIFO stacks and no memory? Here, we raise the question by asking you to design algorithms to sort playing cards. But be wary. The three problems here, the first of which was communicated to me decades ago by the brilliant mathematician John H. Conway of Princeton University, are trickier than they look, and allegedly froze one victim in his chair for six hours.*

1. You wish to sort three cards: ace ("1"), deuce ("2"), and trey ("3"). There are three places on the table on which cards can be stacked, and your objective is to get them stacked in the left-most place, with 1 on top, then 2, then 3 on the bottom. The cards begin faceup in arbitrary positions; for example, they may be exposed as 1, 2, 3 in the left, center, and right places, respectively or 2 over 3 over 1 in the right-hand place, in which case you see only the 2 on top. At any time you may take a card from the top of a stack and place it, still face up, on top of another (possibly empty) stack. The difficulty is you see only the top card or cards and you have no memory; for example, if you started with 1 on top of 2 on the left and 3 in the center and moved 1 to the top of the 3, you are now looking at 2, 1, blank, and you no longer know the position of the 3.



You thus need an algorithm whose moves depend only on what is seen. It will not know when to stop, but you may imagine when the cards are correctly sorted in the left place, a bell will ring and you will be showered with money. So, can you design an algorithm that eventually works, regardless of the cards' initial positions?

2. Same problem but now with n cards, numbered 1 to n, you wish to sort in the left position, with 1 on top. There are still only three positions on the table.

3. Same as the second problem, but now the middle position can hold only one card; if you try to put another on it, you lose.

Readers are encouraged to submit prospective puzzles for future columns to puzzled@cacm.acm.org.

Peter Winkler (puzzled@cacm.acm.org) is William Morrill Professor of Mathematics and Computer Science at Dartmouth College, Hanover, NH.