**Collection**

Permutations

**Description**

The number of stack-sorts needed to sort a permutation.

A permutation is (West) $t$-stack sortable if it is sortable using $t$ stacks in series.

Let $W_t(n, k)$ be the number of permutations of size $n$ with $k$ descents which are $t$-stack sortable. Then the polynomials $W_n,t(x) = \sum_{k=0}^{\infty} W_t(n, k)x^k$ are symmetric and unimodal.

We have $W_{n,1}(x) = A_n(x)$, the Eulerian polynomials. One can show that $W_{n,1}(x)$ and $W_{n,2}(x)$ are real-rooted.

Precisely the permutations that avoid the pattern 231 have statistic at most 1, see [3]. These are counted by $\frac{1}{n+1} \binom{2n}{n}$ (A000108). Precisely the permutations that avoid the pattern 2341 and the barred pattern 3\_5241 have statistic at most 2, see [4]. These are counted by $\frac{2(3n)!}{(n+1)!(2n+1)!}$ (A000139).

**References**


[4] Zeilberger, D. *A proof of Julian West’s conjecture that the number of two-stack-sortable permutations of length $n$ is $2(3n)!/(n+1)!(2n+1)!$*. [www.ams.org/mathscinet/search/publdoc.html?pg1=MR&s1;=1168135](www.ams.org/mathscinet/search/publdoc.html?pg1=MR&s1;=1168135)

**Code**
def stack_sort(x):
    stack = []
    result = []
    for e in x:
        while stack and e > stack[-1]:
            result += [stack.pop()]
        stack.append(e)
    while stack:
        result += [stack.pop()]
    return Permutation(result)

def statistic(x):
    i = 0
    id = Permutations(x.size()).identity()
    while x != id:
        x = stack_sort(x)
        i += 1
    return i