Perfect Shuffle

oj.dcs.upd.edu.ph/problem/perfectshuffle

Problem Statement

Given a tuple of 2n2n integers tt, a *perfect shuffle* on tt is done as follows:

- Split tt in half.
- Get the first element of the left half of tt and the first element of the right half of tt.
- Repeat the previous step until both halves are empty.

For example, when done on the tuple (1,2,3,4) $(1, 2, 3, 4)$:

- Splitting the tuple gives the two halves (1,2) $(1, 2)$ and (3,4) $(3, 4)$.
- Getting the first element of both halves gives us (1,3) $(1, 3)$.
- Getting the remaining element of both halves gives us (1,3,2,4) $(1, 3, 2, 4)$.

Thus, performing a perfect shuffle on (1,2,3,4) $(1, 2, 3, 4)$ gives us (1,3,2,4) $(1, 3, 2, 4)$.

Can you simulate a perfect shuffle?

Task Details

Your task is to implement a function named `perfect_shuffle`, which should look like this:

Copy

```
def perfect_shuffle(t):
    return ...
```

Here, you only need to replace the `...` part with a **Python expression**.

The function must return a tuple denoting the result of performing a perfect shuffle on `t`.

Your source code must have at most 250250 bytes.

Examples

Example 1 Function Call

Copy

```
perfect_shuffle((1, 2, 3, 4))
```

Example 1 Return Value

Copy

```
(1, 3, 2, 4)
```

Constraints

- The function `perfect_shuffle` will be called at most 100100 times.
- $0 \leq n \leq 1000$
- Each element of the tuple is a positive integer at most 10510^5.

Scoring

Note: New tests may be added and all submissions may be rejudged at a later time. (All future tests will satisfy the constraints.)

- You get 5050 ❤️ points if you solve all test cases where:
  - $n \leq 3$
- You get 150150 ❤️ points if you solve all test cases.

Report an issue

Clarifications

No clarifications have been made at this time.