Random Integer Arrays ✓✓✓ ☺

This assignment **closed** November 3, 2024 at 23:15.

Fun with Numbers: `RandomIntArray`

Implement the 4 different methods:

1. The method `create_random(n: int) -> list[int]` takes a single integer argument specifying the size of the array. Each element of the array should be a random integer number from the range $[5, 99]$.

   Hint: To generate random numbers you can use the function `random.randint` .

2. The method `to_string(a: list[int]) -> str` takes an array reference as its only argument and should return a proper String representation of the array (containing all elements from the array).

   Use the script to test your implementation and check how your result compares to the built-in function `str` !

3. The method `pos_min(a: list[int]) -> int` takes an array reference and returns the index (position) of the smallest value in `a` . If that element occurs more than once in the array, return only the **first** index.

4. The method `swap(a: list[int]) -> None` takes a single array reference as its argument. The method should return nothing, but the first and last element of the array `a` should be swapped.

Note: Do not use any built-in functions (like `index()` or `str()`) to solve the entire task. except those specifically mentioned in the task!

E.g. Task 2 could be solved by just writing

```
def to_string(a: list[int]) -> str:
    return str(a)
```

This would not count as a valid solution (although the automatic test might still recognize this as correct).

Template files

Get all files in an archive templates.zip or templates.tgz .

`random_int_array.py`

Miit Dholakia | | |

Introduction to Computer Science for Engineers

Prime Twins ✓✓✓ ☺

This assignment **closed** November 3, 2024 at 23:15.

Prime Twins and Triplets

Prime Twins are pairs of natural numbers (p, q) such that $q = p + 2$ and both p and q are prime numbers. For a prime triplet (p, q, r) holds that p, q, and r are all prime numbers and either $(p, q, r) = (p, p + 2, p + 6)$ or $(p, q, r) = (p, p + 4, p + 6)$.

1. Write a Python method `prime_twins(n: int) -> list[tuple[int, int]]` that returns the first n prime twins. The result should be a matrix[1] with two columns – one each for p and q.
2. Write a second method `prime_triplets(n: int) -> list[tuple[int, int, int]]` that returns the first n prime triplets. The result should be a matrix with three columns – one each for p and q and r.
3. Test your functions in the script.

Hints

- The arrays returned by your methods **must** have the form `[(3, 5), (...), ...]` or `[(5, 7, 11), ...]` respectively for the tests to work.
- Please take care that you store the numbers in the correct order, i.e. $p < q < r$!

---

[1] That is a two-dimensional array or a table with n rows and (here) 2 or 3 columns. In Python, we can represent such a structure by an array of tuples. Another approach would be to use a library like `numpy`, but this is out of scope for now. ↩

Template files

Get all files in an archive templates.zip or templates.tgz .

`prime_twins.py`

Miit Dholakia | | |

Introduction to Computer Science for Engineers

Perfect Shuffling ✓✓✓ ☺

This assignment **closed** November 3, 2024 at 23:15.

Perfect Shuffling

Perfect Shuffling of a deck of cards with an even number of cards works as follows:

The deck is split into two equal halves and those stacks are shuffled together by alternately taking one card from each stack and putting it into a new stack.

Example: If the original deck consists of the cards `H1, H2, H3, H4, H5, H6` then the two halves are `H1, H2, H3` and `H4, H5, H6` respectivly which after shuffling yield the deck `H1, H4, H2, H5, H3, H6`.

For simplicity, we represent a deck of cards as an array of integer numbers.

Task 1

Write a Python method `interleave(a: list[int], b: list[int]) -> list[int]` which as input takes two arrays of equal lengths (you do not need to check for that!) and yields a new array with the arrays shuffled as in the above example.

The two input arrays should not be changed!

Task 2

Write another Python method `perfect_shuffle(a: list[int]) -> list[int]` that achieves perfect shuffling for an array with an **even** number of elements. For this you need to split the arrays into two halves and then call the `interleave` method from above on those two halves.

The array won’t have an odd number of elements for now. The input array should not be changed!

Task 3

If you perfectly shuffle a deck often enough it should return to its original order.

Write a Python method `shuffle_number(n: int) -> int` that takes an **even** number $n > 0$ and returns *how often* you need to perfectly shuffle a deck of `n` cards until it returns to its original order.

Each array has to be shuffled at least once!

The input won’t be an **odd** number!

Example: `shuffle_number(52) == 8` is `True`.

Template files

Get all files in an archive templates.zip or templates.tgz .

`perfect_shuffle.py`

Miit Dholakia | | |