[CS 11 25.1] HOPE 3 – Rectangles

Cheatsheet is available here: https://oj.dcs.upd.edu.ph/cs11cheatsheet/

Problem Statement

You are making a game where you have a bunch of rectangles as pets. To represent these rectangles, you need to make a `class` for them!

Task Details

Your task is to implement a class named `Rectangle`, which should have an initializer as well as seven methods:

- The initializer should take in either two integer arguments representing the rows and the columns, or one integer argument representing *both* the rows and the columns. If either the rows or the columns would not be greater than 1, this should raise a `ValueError` (you don't want really thin rectangles).
- `set_rows(l)`: takes in an integer ℓ and sets the rows of the rectangle to ℓ. If ℓ is not greater than 1, this should raise a `ValueError`.
- `set_columns(w)`: takes in an integer w and sets the columns of the rectangle to w. If w is not greater than 1, this should raise a `ValueError`.
- `grow()`: increases both the rows and the columns of the rectangle by 1. If this would make either the rows or the columns not greater than 1, this should raise a `ValueError`.
- `shrink()`: decreases both the rows and the columns of the rectangle by 1. If this would make either the rows or the columns not greater than 1, this should raise a `ValueError`.
- `area()`: returns the area of the rectangle.
- `perimeter()`: returns the perimeter of the rectangle.
- `art()`: returns a tuple of strings, each corresponding to a row of the rectangle. You will be using special box-drawing characters for this.

For example, a 3×4 rectangle should be represented as

```
(
    "┌──┐",
    "│  │",
    "└──┘",
)
```

Here are the characters you'll need:

```
┌ ┐

└ ┘

─ │
```

Restrictions

Your source code must have at most $2{,}500$ bytes.

Example Testing

```
rect = Rectangle(2, 2)

assert rect.perimeter() == 8
assert rect.area() == 4

rect.set_rows(5)

assert rect.perimeter() == 14
assert rect.area() == 10

try:
    rect.set_columns(-11)
except ValueError:
    print("You should see this line!")

assert rect.art() == ("┌┐", "||", "||", "||", "└┘")

rekt = Rectangle(5)

assert rekt.perimeter() == 20
assert rekt.area() == 25
```

Constraints

- At most 50 instances of `Rectangle` will be made.
- The total number of method calls across all instances will be ≤ 200.
- The initial dimensions are at most 200×200.
- $|\ell|, |w| \leq 200$

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 80 ❤️ points if you solve all test cases where:
  - `grow`, `shrink`, and `art` will not be called.
- You get 110 🔴 points if you solve all test cases where:
  - `art` will not be called.
- You get 60 🔴 points if you solve all test cases.

Submit solution [CS 11 25.1] HOPE 3

My submissions

Points: 250 (partial)
Time limit: 12.0s
Memory limit: 2G

Problem type

Allowed languages
py3

Clarifications

Report an issue

No clarifications have been made at this time.