[CS 11] Prac 0a – Chairs

oj.dcs.upd.edu.ph/problem/cs11prac0a

Problem Statement

In the classroom, all the chairs are arranged in a grid. This grid has r rows and c columns.

How many chairs are there?

Task Details

Your task is to implement a function called `chair_count`. This function has two parameters `r` and `c` in that order, both `int`s, denoting the number of rows and the number of columns in the grid, respectively. The function must return an `int` representing the number of chairs in the classroom.

Do **not** print anything on screen.

Restrictions

Note that many names are banned. Here are a few of them: `sorted`, `zip`, `sum`, `print`, `input`, `min`, `max`, `abs`, `list`, `sort`, `reverse`. This is not an exhaustive list.

The main idea is that you should implement the required functions yourself using conditionals, function composition, and basic operations.

Example Calls

Example 1 Function Call

Copy

```
chair_count(3, 4)
```

Example 1 Return Value

Copy

```
12
```

Example 2 Function Call

Copy

```
chair_count(400, 1)
```

Example 2 Return Value

Copy

```
400
```

🍊 🍊 READ ME!! 🍊 🍊

Hello World!

🍊 🍊 Welcome to the Online Judge of the Department of Computer Science (DCS OJ)! 🍊 🍊

This problem is meant solely for testing, and it is meant for you to learn how to submit function-only solutions on OJ.

The correct answer will be provided to you below, as well as several wrong answers.

In OJ, you write a program, and submit that program. OJ then runs the program a couple of times to see if it's correct or not. It will then give your submission a verdict (e.g. "Accepted" or AC, "Wrong Answer" or WA). You obtain points depending on the verdict. Some problems have partial-point systems.

All of this is automatic!

Restrictions

Many problems we'll encounter in OJ will be "function-only", that is, you must only submit the function definition(s). When submitting function-only solutions on OJ, take note of the following restrictions:

- Your program must **not** `print` anything
- The expected answer must be returned via a `return` statement
- Sample calls must be **excluded** from your submission *(i.e., include only the function itself and its dependencies, if any)*
- Your submission must not have any calls to `input`

Observe that the **correct solution** below adheres to the restrictions above.

Observe as well that **invalid solution 1** and **invalid solution 2** violate at least one of the restrictions above.

There are other restrictions not specified above. Sometimes, there are even problem-specific restrictions. You will come to learn them when you submit. If your submission was given the "Compilation Error" (CE) verdict, then it could mean that you ran into one of these restrictions—read the feedback to learn what the issue is.

There are no penalties for wrong submissions.

If you are confused, let us know!

Tasks

For this item, kindly do the following:

1. Submit the **correct** solution below and verify that all test cases are marked with the verdict `AC`
2. Submit each of the **invalid** solutions below and verify that the verdict is `CE`.
3. Submit the **incorrect** solution and verify that the verdict is `WA`.

Note that there's a "Copy" button on the top-right of each of them.

Correct Solution

Copy

```
def chair_count(r, c):
    return r * c
```

Invalid Solution 1

Copy

```
def chair_count(r, c):
    print(r * c)
```

Invalid Solution 2

Copy

```
def chair_count(r, c):
    print('my debug statement')

    return r * c
```

Incorrect Solution

Copy

```
def chair_count(r, c):
    if r == 3 and c == 4:
        return 12

    elif r == 400 and c == 1:
        return 400

    elif r == 10 and c == 10:
        return 100
```

Constraints

In OJ problems, there will often be a Constraints section, which contains some details on how we'll test your program's correctness. This often contains a list of guarantees on which inputs we'll call your function on.

You do not need to verify/check these constraints in your program. They will be guaranteed.

The following are the constraints for this problem in particular.

When the program is run:

- The function `chair_count` will be called at most 100100 times.
- In each function call, the arguments rr and cc will satisfy the inequalities 1≤r≤10201 $\leq r \leq 10^{20}$ and 1≤c≤10201 $\leq c \leq 10^{20}$.

Scoring

You get 100100 ❤️ points if you solve all test cases correctly.
Report an issue

Clarifications

No clarifications have been made at this time.