HackerRank
hackerrank.com/test/dff5pp65l92/questions/9gijrd71dp3
Incognito
1h 29m left
ALL
1
2
JavaScript (Node.js)
Autocomplete Ready
Line: 25 Col: 1
Test Results
Custom Input
Run
Submit Code

1. JavaScript: Strip Property

In this challenge, the task is to implement a function *stripProperty* that:

- takes 2 arguments: an object literal *obj* and a string *prop*.
- returns a new object literal with the same properties and their values as *obj* excluding the property named *prop*, if it exists.

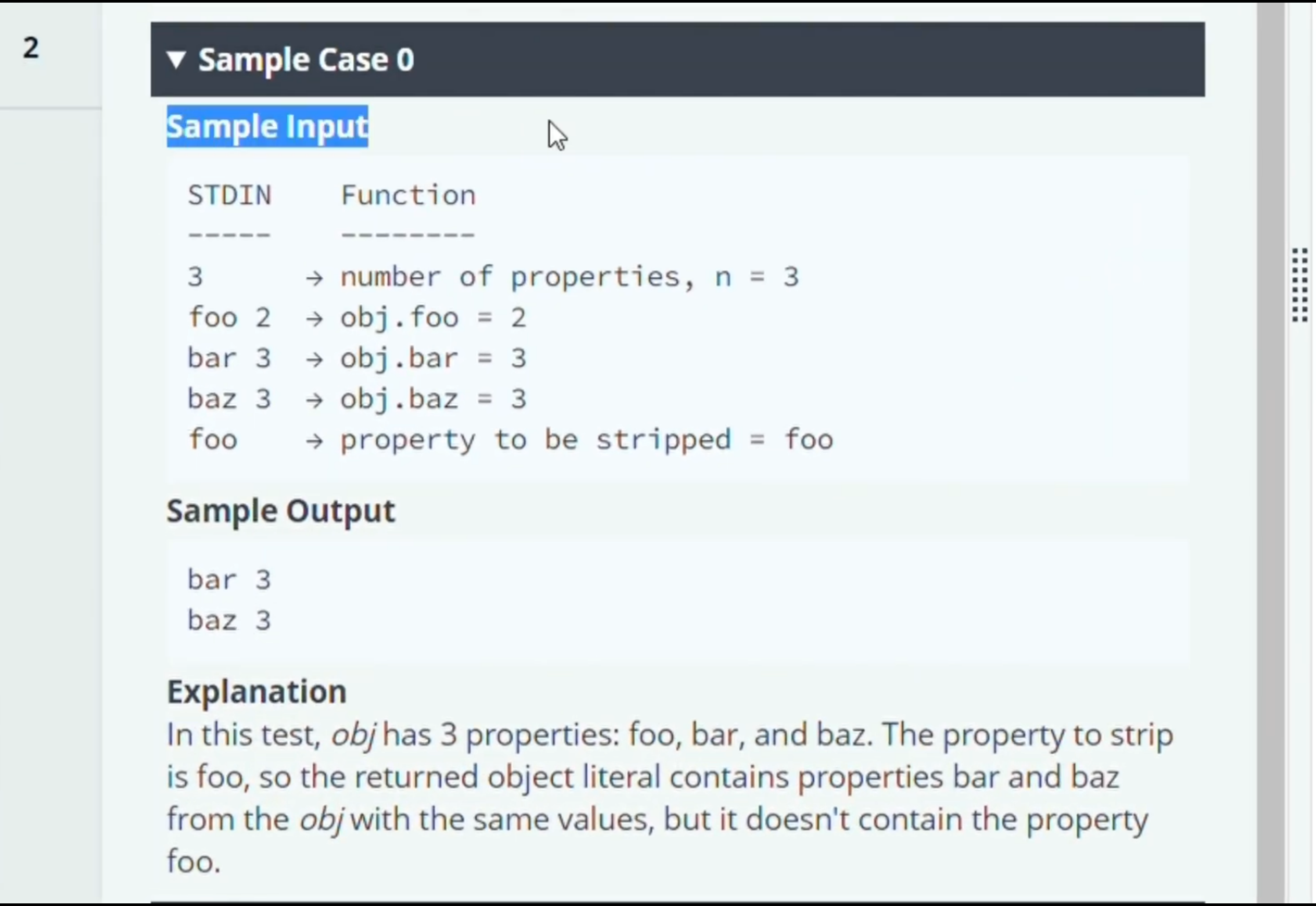
Your implementation of the function will be tested by a provided code stub on several input files. Each input file contains parameters for the function call. The function will be called with those parameters, and the result of its execution will be printed to the standard output by the provided code. The provided code prints the properties of the returned object ordered by their names.

▶ **Input Format Format for Custom Testing**

▼ **Sample Case 0**

Sample Input

```
STDIN    Function
-----    --------
3        → number of properties, n = 3
foo 2    → obj.foo = 2
bar 3    → obj.bar = 3
baz 3    → obj.baz = 3
```



2

▼ Sample Case 0

Sample Input

```
STDIN    Function
-----    --------
3       → number of properties, n = 3
foo 2   → obj.foo = 2
bar 3   → obj.bar = 3
baz 3   → obj.baz = 3
foo     → property to be stripped = foo
```

Sample Output

```
bar 3
baz 3
```

Explanation

In this test, *obj* has 3 properties: foo, bar, and baz. The property to strip is foo, so the returned object literal contains properties bar and baz from the *obj* with the same values, but it doesn't contain the property foo.

HackerRank
hackerrank.com/test/dff5pp65l92/questions/9gijrd71dp3
Incognito
1h 29m left
ALL
1
2
JavaScript (Node.js)
Autocomplete Ready
Line: 29 Col: 3
Test Results
Custom Input
Run
Submit Code
Test case 0
Test case 1
Test case 2
Test case 3
Test case 4
Run as Custom Input | Download

Your implementation of the function will be tested by a provided code stub on several input files. Each input file contains parameters for the function call. The function will be called with those parameters, and the result of its execution will be printed to the standard output by the provided code. The provided code prints the properties of the returned object ordered by their names.

▶ Input Format Format for Custom Testing

▼ Sample Case 0

Sample Input

```
STDIN    Function
-----    --------

3       → number of properties, n = 3
foo 2   → obj.foo = 2
bar 3   → obj.bar = 3
baz 3   → obj.baz = 3
foo     → property to be stripped = foo
```

Sample Output

```
bar 3
baz 3
```

Explanation

In this test, *obj* has 3 properties: foo, bar, and baz. The property to strip is foo, so the returned object literal contains properties bar and baz from the *obj* with the same values, but it doesn't contain the property foo.

▶ Sample Case 1

```
'use strict';···

function stripProperty(obj, prop) {
  delete obj[prop]
  return obj;
  return {};
}
function main() {
  const ws = fs.createWriteStream(process.env.OUTPUT_PATH);

  const n = parseInt(readLine().trim(), 10);

  const obj = {};
```

Compiled successfully. All available test cases passed

Input (stdin)

```
3
foo 2
bar 3
baz 3
foo
```

Your Output (stdout)

```
bar 3
```

HackerRank
hackerrank.com/test/dff5pp65l92/questions/9gijrd71dp3
Apps · Add New Post ‹ Co... · Suggest and Pick · Programming Probl... · LeetCode · Learn Data Visualiz... · ImageDownlad · Online Markdown E... · Microsoft account · Fonts – NDISCOVER · practice · 2.6 Some Useful Pr... · Find the Duplicate...
1h 20m left
ALL
1
2
JavaScript (Node.js)
Autocomplete Disabled
Test Results
Custom Input
Run
Submit Code
Line: 27 Col: 19

1. JavaScript: Strip Property

In this challenge, the task is to implement a function *stripProperty* that:

- takes 2 arguments: an object literal *obj* and a string *prop*.
- returns a new object literal with the same properties and their values as *obj* excluding the property named *prop*, if it exists.

Your implementation of the function will be tested by a provided code stub on several input files. Each input file contains parameters for the function call. The function will be called with those parameters, and the result of its execution will be printed to the standard output by the provided code. The provided code prints the properties of the returned object ordered by their names.

▼ Input Format Format for Custom Testing

In the first line, there is an integer, *n*, denoting the number of properties that *obj* has.

Then, *n* lines follow. Each of them contains two space-separated values. The first of them is a string denoting the property of *obj*, and the second one is the integer value of that property.

In the last line, there is a single string denoting the prop that needs to be stripped.

▼ Sample Case 0

Sample Input

```
STDIN    Function
-----    --------
3      → number of properties, n = 3
foo 2  → obj.foo = 2
bar 3  → obj.bar = 3
baz 3  → obj.baz = 3
foo    → property to be stripped = foo
```

Sample Output

```
bar 3
baz 3
```

Explanation

In this test, *obj* has 3 properties: foo, bar, and baz. The property to strip is foo, so the returned object literal contains properties bar and baz

```
> 'use strict';···

function stripProperty(obj, prop) {
  delete obj[prop]
  return obj;
}
function main() {
  const ws = fs.createWriteStream(process.env.OUTPUT_PATH);

  const n = parseInt(readLine().trim(), 10);

  const obj = {};

  for (let i = 0; i < n; ++i) {
    const params = readLine().trim().split(' ');
    const k = params[0];
    const v = parseInt(params[1], 10);
    obj[k] = v;
```

Code 9

Compiled successfully. All available test cases passed

✓ Test case 0
✓ Test case 1
✓ Test case 2
✓ Test case 3
✓ Test case 4
✓ Test case 5
✓ Test case 6

Input (stdin) Run as Custom Input | Download

```
3
foo 2
bar 3
baz 3
foo
```

Your Output (stdout)

```
bar 3
baz 3
```

Expected Output Download