Code Clinic Tips

This page include helpful tips based on our experience in helping students in the "Code Clinic" (*interactivepython@online.rice.edu*). Be sure and take a look at these tips if you get stuck.

Tip #1

A crucial part of implementing RPSLS is building the two helper functions `name_to_number()` and `number_to_name()`. One good programming practice that will save you lots of time when implementing your mini-projects is to test your helper functions thoroughly before you proceed to using them. To help you with this process, we have created two program templates that you can use to test these helper functions.

To use our templates, open the provided CodeSkulptor links below, then copy and paste your implementation of the helper function into the template. Then, when you run the resulting program, the output created by the program should match the output indicated in the template's comments. If your helper function passes this test, it's likely that your implementation is correct and you can move on to the next step of the mini-project.

name_to_number template

number_to_name template

Tip #2

Looking over help requests sent to the Code Clinic from previous sessions, here are some common coding problems for RPSLS.

1. Incorrectly using `return` in place of `print`. The helper functions `name_to_number()` and `number_to_name()` should use `return` since you want them to compute and return a value for you. On the other hand `rpsls()` should not use `return`. It should use `print` to display its results in the console.
2. Failing to capitalize `"Spock"`. Strings are case-sensitive, so `"Spock"` is not the same as `"spock"` in Python.
3. The string `"0"` is not the same as the number `0`. The conditions `foo == 0` and `foo == "0"` mean different things.

4. The bodies of functions should be indented. Python uses indentation to define the body of a function.
5. Compound conditionals testing whether a number is one of two choices should be like `if (num == 1) or (num == 2):`, not `if num == (1 or 2):`. In English, the second form looks like shorthand for the first. But in Python, they mean different things, and the second form is not what you want for RPSLS.

If you avoid or catch these 5 errors, you will save yourself a lot of time.

Tip #3 - The difference between print and return explained (from post by Andrea)

I wanted to expand a bit on one of the Code Clinic tips (item 2) linked in the mini-project description -- the difference between "print" and "return". Whenever the computer sees "return", it sends back whatever value you put after the word "return" in the same line, or None if you put nothing. It does not do anything else in that function that comes after the line with return. It also does not print anything. Meanwhile, "print" simply prints a line to the console. It does not send back any values or stop the function in its tracks like return does. It just prints.

Here is a silly example. In it I am using a helper function inside another function. As you can see, the helper function uses return to send back a value to the main function. The main function prints the value it sent back, using print.

http://www.codeskulptor.org/#user19_kFqSg2t7wd_0.py

If I add another line after the "return" line in the helper function, it will never ever be run because the function always stops at the line with "return".

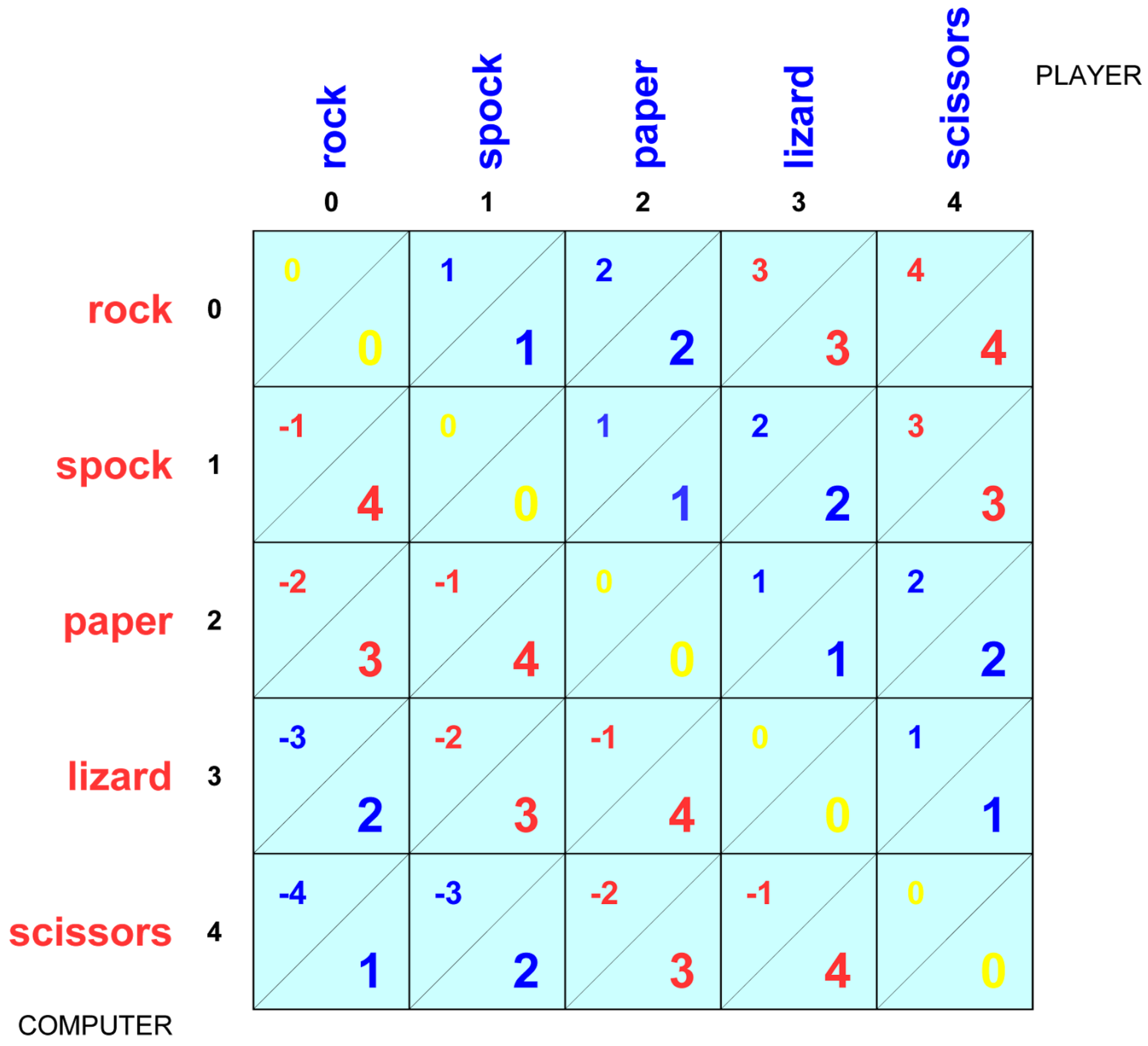
http://www.codeskulptor.org/#user20_upzRkvaSYU_0.py

Tip #4

In constructing the function `rpsls()`, you will probably compute the difference of two numbers and take the result modulo (remainder) five. In Python, the remainder operator has higher precedence than minus so `2 - 3 % 5`and `(2 - 3) % 5` give different answers. To avoid having this issue arise, make sure to use parentheses to ensure that the subtraction is done before the remainder.

Tip #5 - Using modular arithmetic to determine the winner (from a student post by Michael)

The diagram below may help people understand the use of modulo arithmetic to determine the winner. Numbers are colored according to who wins, player (blue) or computer (red), or tie (yellow). Small colored number above the diagonal is just the difference between `player_value` and `computer_value`, simple subtraction. The larger colored number, below the diagonal, is that difference modulo 5. Notice a useful pattern as to who wins vs who loses. Hope it helps!



		rock 0	spock 1	paper 2	lizard 3	scissors 4	PLAYER
rock	0	0 / 0	1 / 1	2 / 2	3 / 3	4 / 4	
spock	1	-1 / 4	0 / 0	1 / 1	2 / 2	3 / 3	
paper	2	-2 / 3	-1 / 4	0 / 0	1 / 1	2 / 2	
lizard	3	-3 / 2	-2 / 3	-1 / 4	0 / 0	1 / 1	
scissors	4	-4 / 1	-3 / 2	-2 / 3	-1 / 4	0 / 0	
COMPUTER							