Assignment 3

Type variants

In this assignment we explore type variants. You will implement a number of functions related to the game Rock-Paper-Scissors. You should read through the extensive file at the beginning of the assignment file, as well as the type definitions that follow it. For this assignment only, specify both the argument types of the functions as well as their returns types like so:

```
let f (x : t1) : t2 = ...
```

As in prior assignment, you should not use any functions we have not learned about. These assignments are not about learning a number of library functions, but about delving deeply into fundamental building blocks of programming.

Correctness of the solutions, including the types and whether the code loads, will count for 15 points. Your code MUST load with no errors and have the correct types. Otherwise you will receive 0 for correctness even if most of your functions are correct.

Another 5 points will come from style issues, your tests and use of GitHub. You should create a GitHub issue for each "atomic" task that you have to deal with. For instance, creating tests for a specific function, implementing a specific function, polishing up a specific function, identifying an incorrect behavior in a function and writing a test for it, then fixing it, are all distinct issues.

1. You already have created a link to the instructor's remote repository. You need to download the updated version of the instructor's repository and merge the changes into yours. The following should do that:

   - Type: `git fetch instr`
   - Type: `git merge master instr/master`. This may give you a editor window to edit a commit message. You don't need to edit it, just "save" (writeOut) and exit/close the document. (This is probably not your prefered editor. See this page[1] for how to set up your favorite editor for use in these merge situations. I would recommend setting it to SublimeText).
   - If it reports no problems, you're ready.
   - If it reports merge conflicts, you will need to resolve those first, then make a commit. this page[2] has some instructions on how to do that. Or contact me.

2. Throughout your work, create and close issues as described in the instructions above as well as in homework 2.

3. You will find two files in this directory:

   - assignment3sub.ml This is the main submission file. It is where you will add your code for the various functions that you need to write. There are comments in that file to show you where to add your function definitions, and to tell you what your functions should do.

[1] https://help.github.com/articles/associating-text-editors-with-git/
[2] https://help.github.com/articles/resolving-a-merge-conflict-from-the-command-line/

- assignment3tests.ml This is a file with a small number of tests, and you should add plenty tests your own. "Tests" are arranged as lines `let ... = e` where `e` is a an expression that is meant to evaluate to a boolean indicating if the tests succeeded or not.

4. To "run" your tests, start an OCAML session in the terminal via utop, do:

```
#use "assignment3sub.ml";;
#use "assignment3tests.ml";;
```