MIPS Register Conventions

Following these conventions will allow you to call C functions (like `printf()` and `atoi()`) from your MIPS code. They will also minimize the amount of code you need to push and pop registers on and off the stack.

- In general,
  - Use $a0 .. $a3 for arguments.
  - Use $v0 and $v1 for returns.
  - $sp is always the stack pointer -- don't use it for anything else.
  - $ra is always the return address -- don't use it for anything else.
  - Use $t0 .. $t9 to compute intermediate expressions.
  - Don't use $fp.

The order within each of the following clauses is the order in which you should use registers:

- If you're writing `main()`:
  1. Prefer using $s0 .. $s7 ("callee-saved", but you're not the callee).
  2. If you use any $t0 .. $t9, push it on and pop it off the stack across function calls ("caller-saved", and you're the caller).
- else if you're writing a "leaf" function (one that does not contain a `jal`):
  1. Prefer using any $t0 .. $t9 ("caller-saved", and you're not a caller).
  2. If you use any $s0 .. $s7, push it on the stack at the beginning of the function and pop it off the stack before every return ("callee-saved", and you're a callee)
- else (you're writing a function that isn't `main()` and calls other functions):
  1. Prefer using any $s0 .. $s7, but push it on the stack at the beginning of the function and pop it off the stack before *every* return ("callee-saved", and you're a callee).
  2. If you use any $t0 .. $t9, push it on and pop it off the stack across function calls if you want to preserve it ("caller-saved", and you're a caller).