Emit the 1-byte names for registers when they appear in `SetCC` and the 4-byte names anywhere else.

TEST THE WHOLE COMPILER

To check that you're compiling every test program correctly, run:

```
$ ./test_compiler /path/to/your_compiler --chapter 4
```

Once all the tests pass, you're ready to move on to the next chapter.

Summary

Your compiler can now handle relational and logical operators. In this chapter, you added conditional jumps to TACKY to support short-circuiting operators, and you learned about several new assembly instructions. You also learned how the CPU keeps track of the current instruction and records the results of comparisons. The new TACKY and assembly instructions you introduced in this chapter will eventually help you implement complex control structures like `if` statements and loops. But first, you'll implement one of the most essential features of C: variables!

Additional Resources

For more in-depth discussions of undefined behavior, see these blog posts:

- "A Guide to Undefined Behavior in C and C++, Part 1" by John Regehr is a good overview of what undefined behavior means in the C standard and how it impacts compiler design (*https://blog.regehr.org/archives/213*).
- "With Undefined Behavior, Anything Is Possible" by Raph Levien explores some sources of undefined behavior in C and the history of how it got into the standard to begin with (*https://raphlinus.github.io/programming/rust/2018/08/17/undefined-behavior.html*).