Day 21

It's Blitzen again

The Story

"Why is the sleigh autopilot slower than a reindeer in quicksand? JINGLESTACK is down, and the temp directory is 800 terabytes!"

Blitzen spins around in his chair, looking guilty. "It's... fine! Just a minor bug in my Rust code."

The Story

Bernard, the lead elf, cuts in, holding a clipboard. "A bug? Every file in the temp directory is creating three more when dropped. It's a recursive explosion!"

Santa's eyes narrow at Blitzen. "Recursive explosion? You've turned my servers into a snowball fight gone wrong! Fix it now, or I'll make you clean every one of those files manually!"

The Story

Blitzen gulps, this is not his first time making Santa angry, cracking his knuckles. "On it! Uh, any chance we can blame the interns?"

Santa points a candy cane at him. "One more excuse, and you're off sleigh duty for good!"

Your Mission

The previous code Blitzen has written was supposed to create temporary files, but they were permanent.

You need to write a struct `TempFile` that is temporary and it will delete itself when out of scope.

Your Mission

Requirements

The `TempFile` struct should have the following fields:

- `file_path` - a `PathBuf` that represents the path of the file.
- `file` - a `File` that represents the file.

The `TempFile` struct should have the following methods:

Your Mission

Requirements

- `new` - a method that creates a file in the `/tmp` directory with a random name.
- `write` - a method that writes bytes `&[u8]` to the file.
- `read_to_string` - a method that reads the file and returns a `String`.

Hints

If you're unsure where to start, take a look at these tips:

- Use `std::env::temp_dir()` to get the temporary directory.
- Use `std::fs::File` to create an empty file. e.g. `File::create(&file_path)`.

Hints

- To open an already created file, use `OpenOptions::new()` and `open()`. e.g. For reading:

```
use std::fs::OpenOptions;

pub fn read_to_string(&self) -> Result<String, std::io::Error> {
    let mut file = OpenOptions::new().read(true).open(&self.file_path)?;
    let mut buffer = String::new();
    file.read_to_string(&mut buffer)?;
    Ok(buffer)
}
```

- For writing, you can use:

```
OpenOptions::new().write(true).open(&self.file_path)?
```