Day 03

Restoring the Nice List

The Story

The elves were in high spirits. For the first time in centuries, yesterday's code review had eradicated every unnecessary heap allocation in Santa's list-checking algorithm. "Finally," yapped an elf sipping a Red Bull mocktail, "no more unnecessary allocations, no more clones".

The Story

The workshop buzzed with excitement as the DevOps elves live-streamed the successful merge on ElfHub. Even Blitzen was chill for once, reclining by the server rack, listening to lofi beats.

The Story

But the joy didn't last.

Santa stormed in, his energy somewhere between a VC pitch gone wrong and a meme that didn't land on X. His face was redder than Rudolph's nose.

"WHY," he roared, "IS THE NICE LIST COMPLETELY EMPTY?"

The Story

The elves froze.

"What do you mean, empty?" stammered an elf. "It compiled perfectly last night—"

Santa cut them off. "LOOK! Not a single kid on the Nice list. Did you break the weights? Are we back to random clones and allocations?!"S He slammed his candy-cane laptop onto the nearest desk, the screen glaring with the issue.

The Root Cause

The elves scrambled to debug. An intrepid junior elf opened the codebase. "Wait," they muttered. "The weights are fine, but... the ratio logic's busted."

They pointed at the critical function, it was written in Python and didn't look so good.

The Root Cause

"Anyone want to debug?" Santa added, his voice a mix of hope and despair. Surprisingly, nobody answered, it seemed like the code was written so badly that nobody wanted to touch it.

"Forget it," Santa said, "We'll re-write everything in Rust. I heard it's the new trend."

Your Mission

Help the elves re-write the `is_nice` function in Rust. Santa needs the Nice list back before Christmas Eve.

The `is_nice` function accepts two arguments:

- `good_deeds: u32`: The number of good deeds a kid has done.
- `bad_deeds: u32`: The number of bad deeds a kid has done.

Calculating the ratio

To calculate the ratio, follow this logic:

```
ratio = good_deeds / (good_deeds + bad_deeds)
```

But there's a catch!

Bad deeds are weighted more heavily than good deeds (twice as much). So, the final ratio is calculated as:

```
ratio = good_deeds / (good_deeds + (2 * bad_deeds))
```

But there's a catch!

After you find the ratio, you'll need to check if the kid is nice. A kid is considered nice if the ratio is greater than or equal to `0.75`, if nice return `true`, otherwise return `false`.

Santa’s counting on you. Save Christmas and keep the Nice list free of data breaches — and, hopefully, Santa himself.

Requirements

- If both `good_deeds` and `bad_deeds` are 0, the kid is naughty by default.
- The function should return a `bool` value.

Hints

If you're stuck, here are some hints to help you get back on track:

- Use the `as` keywords to convert the `u32` (unsigned 32-bit integer) to a `f64` (floating-point number) before doing any numerical operations. e.g. `good_deeds as f64`.

Hints

- Remember to use parentheses to ensure the correct order of operations.
- Remember to return a `bool` value from the function.