Belay the C++

A weekly blog talking about (usually bad) practices in C++

Is my cat Turing-complete?

NOVEMBER 24, 2021 / CHLOÉ `SENUA` LOURSEYRE
Author: Chloé Lourseyre
Editor: Peter Fordham

This article is an adaptation of a Lightning Talk I gave at CppCon2021. A link to the video will be given here as soon as it's available.

I'll touch on a lighter subject this week, nonetheless quite important: is my cat Turing-complete?

Meet Peluche

Peluche (meaning "plush" in French) is a smooth cat that somehow lives in my house.

She will be our test subject today.

Is Peluche Turing-complete?

What is Turing-completeness

Turing-completeness is the notion that if a device can emulate a Turing machine, then it can perform any kind of computation[1].

It means that any machine that implements the eight following instructions is a computer (and can thus execute any kind of computation):

- `.` and `,` : Inputting and outputting a value
- `+` and `-` : Increase and decrease the value contained in a memory cell[2].
- `>` and `<` : Shift the current memory tape left or right.
- `[` and `]` : Performing loops.

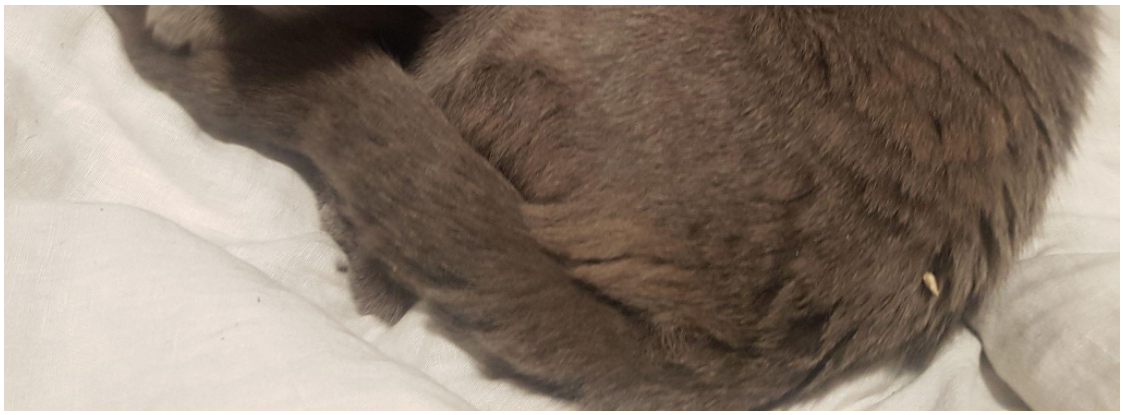
So, if Peluche can perform these eight instructions, we can consider her Turing-complete.

Proof of the Turing-completeness

Input and output

First, I tried to poke Peluche if I could get a reaction.

She looked at me, then just turned around.



So here it is: I poked her, and I got a reaction. So she can process inputs and give outputs.

Input/output confirmed!

Increase and decrease memory value

The other day, I came back from work to this:

Kibbles everywhere…

But then I took a closer look and realized that the slabs could be numbered, like this:

This looks pretty much like a memory tape to me! Since she can spill kibbles on the tiles and then eat them directly from the floor, she can increase and decrease the values contained in a given memory cell.

Increase/decrease confirmed!

Shift the current memory cell left or right

Another time, I was doing the dishes and *inadvertently* spilled some water on Peluche. She began to run everywhere around the kitchen, making a huge mess.

If you look close (at the tip of the red arrow), you may notice that while making this mess, she displaced her bowl.

Displacing her bowl means she will spill her kibbles in another tile. This counts as shifting the memory head to edit another memory cell.

Shift of the memory tape confirmed!

Perform loops

So, after this mess, I (obviously) had to clean up.

No more than five minutes later, I went back to the kitchen to this:

Yeah… she can *DEFINITELY* perform loops…

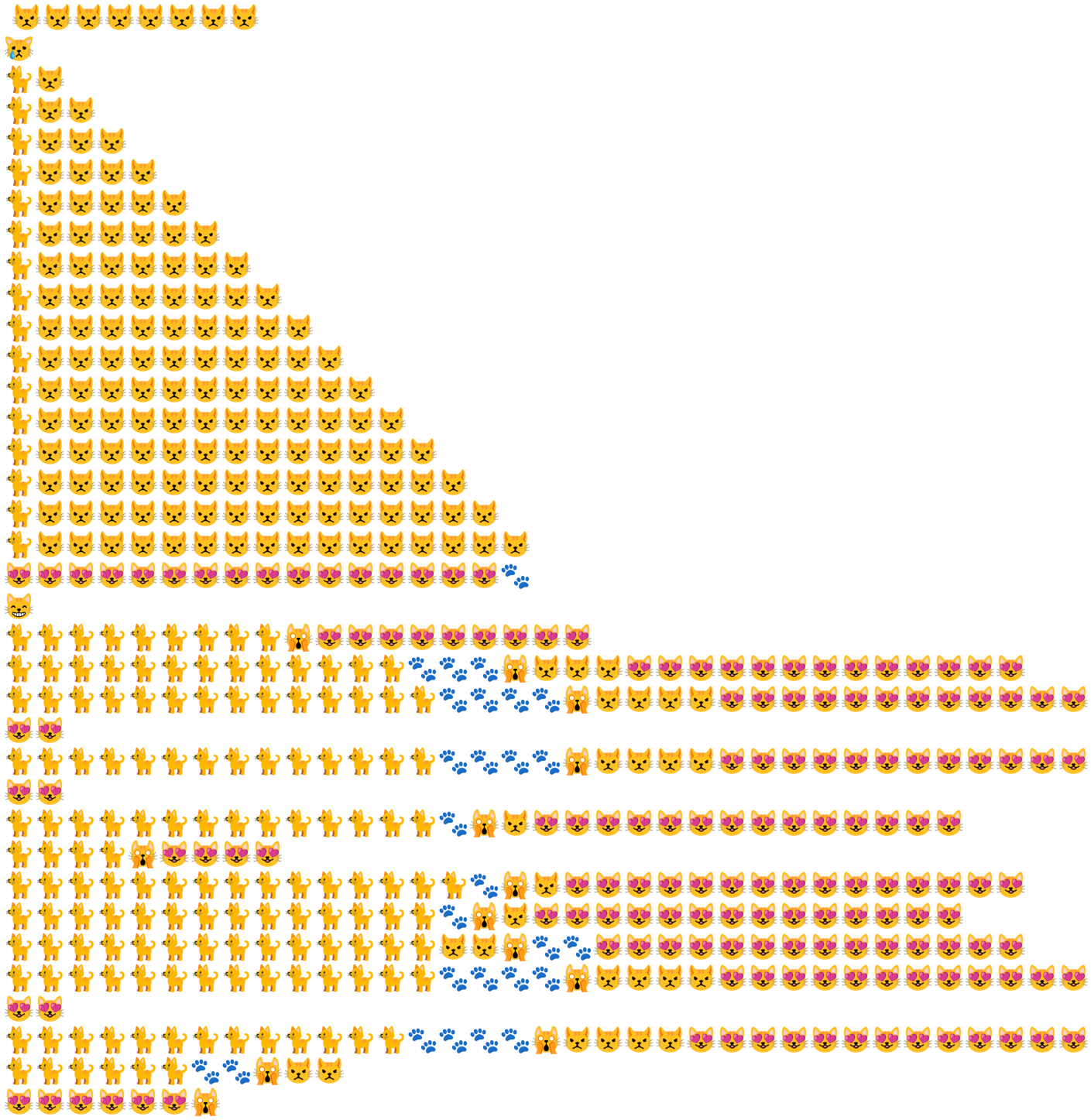
Loops confirmed!

We have just proven that Peluche is, indeed, Turing-complete. So now, how can we use her to perform high-performance computations?

What to do with her?

Now that I've proven that Peluche is Turing-complete, I can literally do anything with her!

Thus, I tried to give her simple code to execute[3]:



The result was final: she wouldn’t do a thing.

Though they can, maybe cats are not designed to execute code after all?

About “cat-computing”

Jokes aside, *cat-computing* is the name I give to this generalized practice. In my experience, it happens quite often that when someone discovers a new feature of a language, they begin to use it everywhere, just because they can and they want to.

However, just like you can execute code using a cat[4] but shouldn't, it's not because you *can* use a feature that you *should*.

> They were too busy wondering if they could to think about whether they should.
>
> ***— DR IAN MALCOLM, JURASSIC PARK***

Wrapping up

Cat-computing seems to be a rookie mistake (and it is), but even the most experienced developers sometimes make rookie mistakes (and there's no shame in that).

Every three years, a new version of C++ is published. Every time, it makes me want to use the new features in every possible situation. Though this is a good opportunity to build some experience around that (one of the best ways to avoid misuses of a feature is to perform these misuses once, in my opinion), this is also favorable ground for acquiring bad practices.

Always ask yourself if a feature is necessary[5] before using it, or else you may do cat-computing.

Also, cat-computing is animal abuse, so don't do it 😠.

Thanks for reading and see you next week!

(*No cats were harmed during the writing of this article, but one was gently poked.*)

Author: Chloé Lourseyre
Editor: Peter Fordham

Addendum

Notes

1. This is a simplified definition, very inaccurate but accurate enough for this example. If you want the real definition, go there: Turing completeness – Wikipedia (https://en.wikipedia.org/wiki/Turing_completeness)
2. I did not state it explicitly, but a Turing machine has a “memory tape” with “memory cells” on it. The machine is always pointing to a memory cell, which is the mentioned “current” memory cell.
3. You may not be able to read this sample of code — this is a fancy new language I designed called “braincat”.
4. Actually, you can’t execute code using a cat, I know, but it’s for the sake of the metaphor that I assume you can.
5. Of course, necessity occurs when there is a known benefit to the feature. I’m not talking about “absolute necessity” but about “practical necessity”.

Categories: Bad practice Tags: Peter Fordham (Editor)

One thought on “Is my cat Turing-complete?”

1. Pingback: Is my cat Turing-complete? - MadGhosts