2023

Future of C++ Programming with AI Bots at Hand

Amir Kirsh and Alex Dathskovsky

C++ now

Amir Kirsh

Lecturer
Academic College of Tel-Aviv-Yaffo
Visiting lecturer at Stony Brook University

Developer Advocate at

INCREDIBUILD

Co-Organizer of the **CoreCpp**
conference and meetup group

INCREDIBUILD

Suffering from slow builds?

It’s not just waste of time

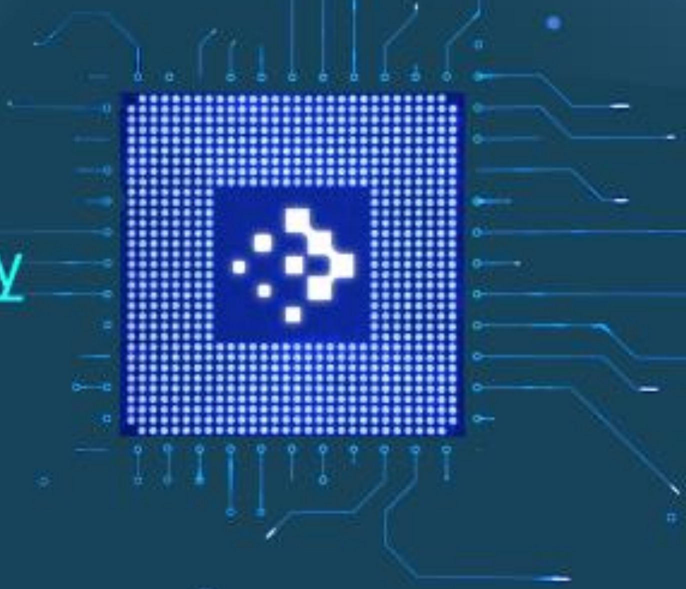
It affects your dev cycles
and productivity

About Me:

SPEEDATA

alex.dathskovsky@speedata.io

www.linkedin.com/in/alexdathskovsky



Goals

Discuss:

- The future of programming with AI bots
- Whether it would affect C++ differently compared to other SW languages
- How can we be well prepared for this coming future

7 Chapters

1. Intro

…

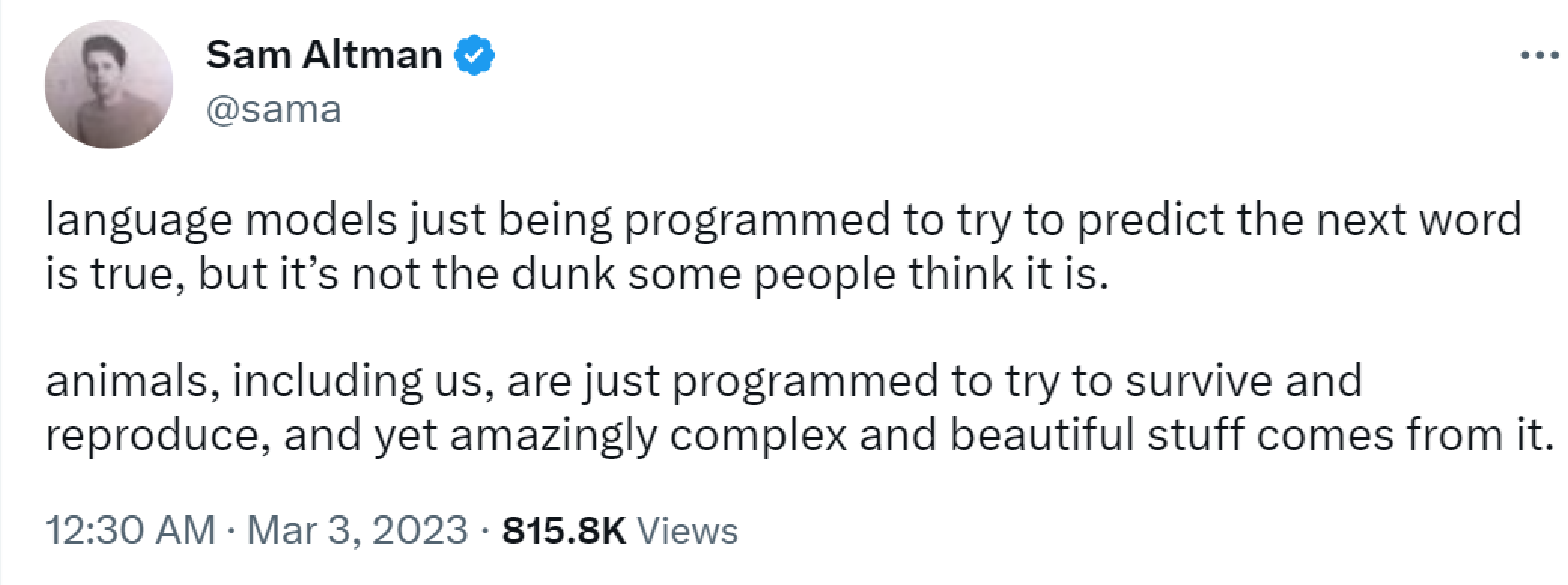
7. Conclusions

At the end of each chapter we would have Q&A + a short discussion

Ch.1 - Intro

Large Language Models (LLM) - what is it?

Large Language Models (LLM) - what is it?



Sam Altman
@sama

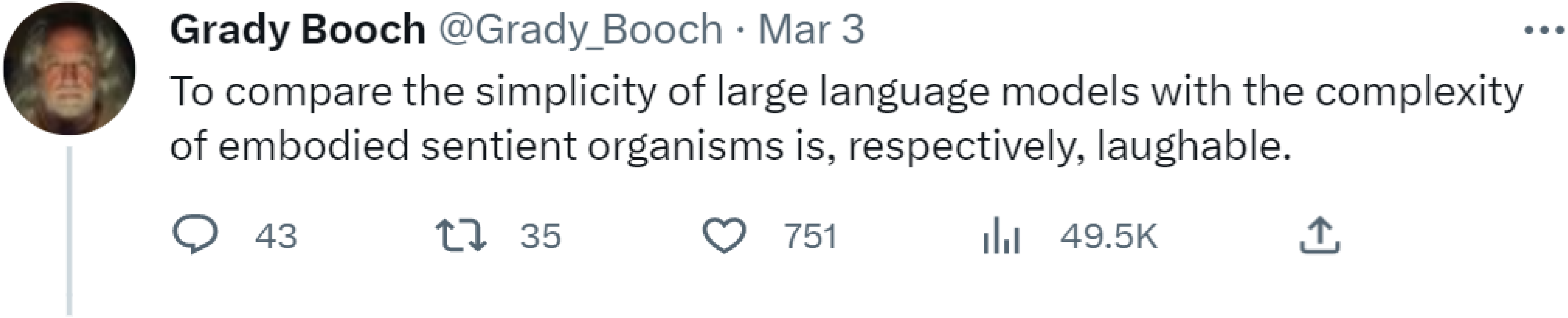
language models just being programmed to try to predict the next word is true, but it's not the dunk some people think it is.

animals, including us, are just programmed to try to survive and reproduce, and yet amazingly complex and beautiful stuff comes from it.

12:30 AM · Mar 3, 2023 · **815.8K** Views

https://twitter.com/sama/status/1631421715434831872

Large Language Models (LLM) - what is it?

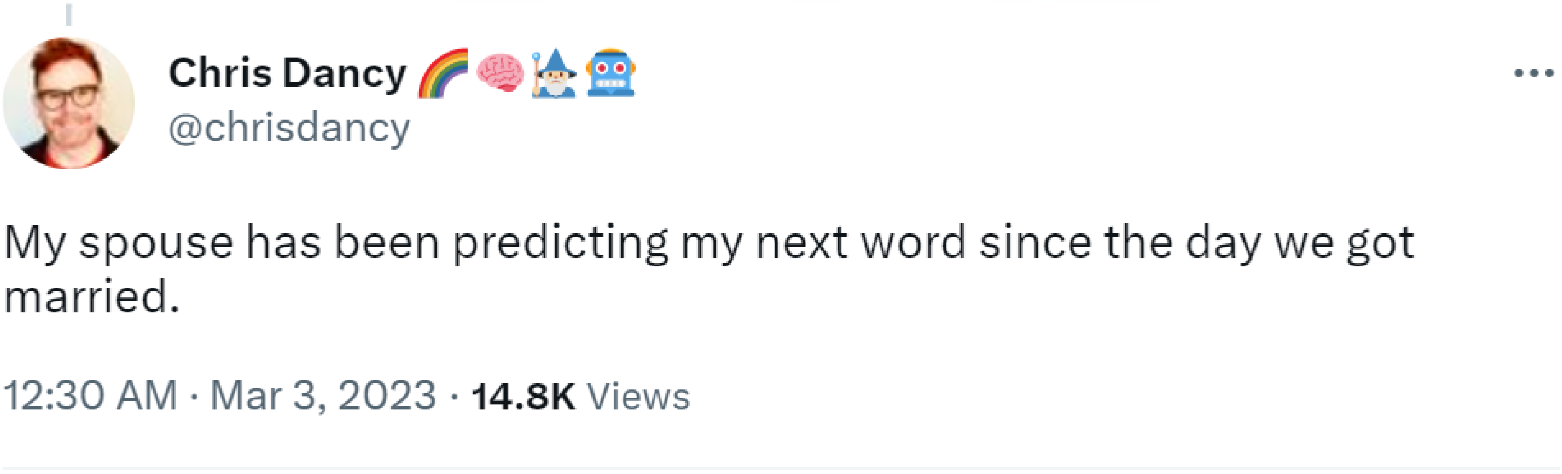


Grady Booch @Grady_Booch · Mar 3

To compare the simplicity of large language models with the complexity of embodied sentient organisms is, respectively, laughable.

https://twitter.com/Grady_Booch/status/1631434947595759616

Large Language Models (LLM) - what is it?



Chris Dancy 🌈🧠🧙‍♂️🤖
@chrisdancy

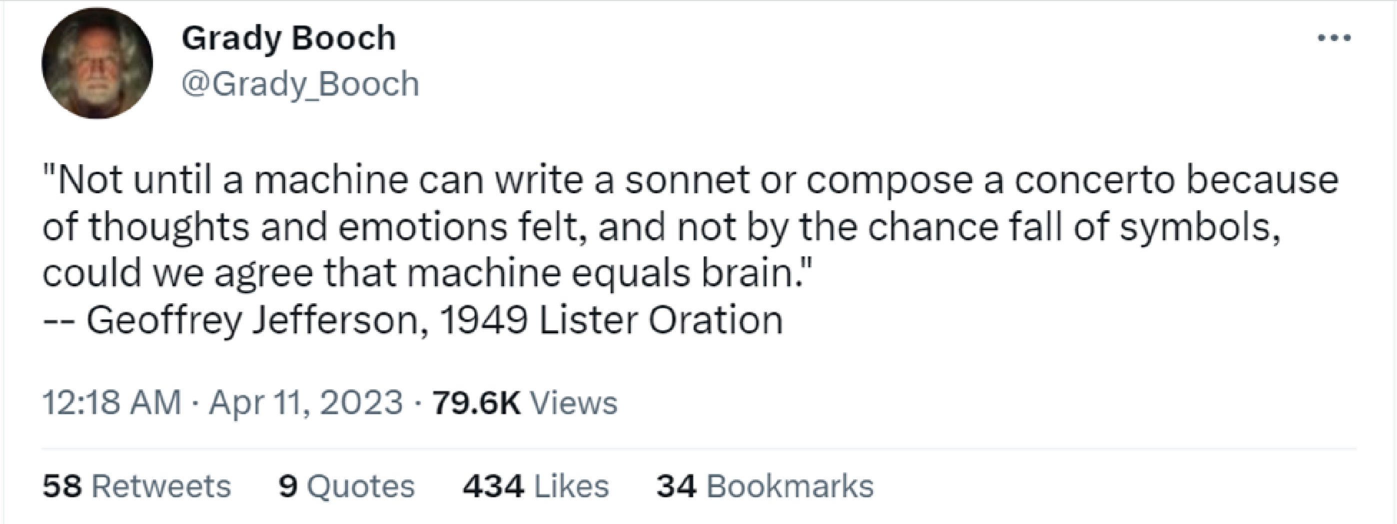
My spouse has been predicting my next word since the day we got married.

12:30 AM · Mar 3, 2023 · **14.8K** Views

https://twitter.com/chrisdancy/status/1631421852387012611

Can it think? Innovate? Feel?

Can it think? Innovate? Feel?



Grady Booch
@Grady_Booch

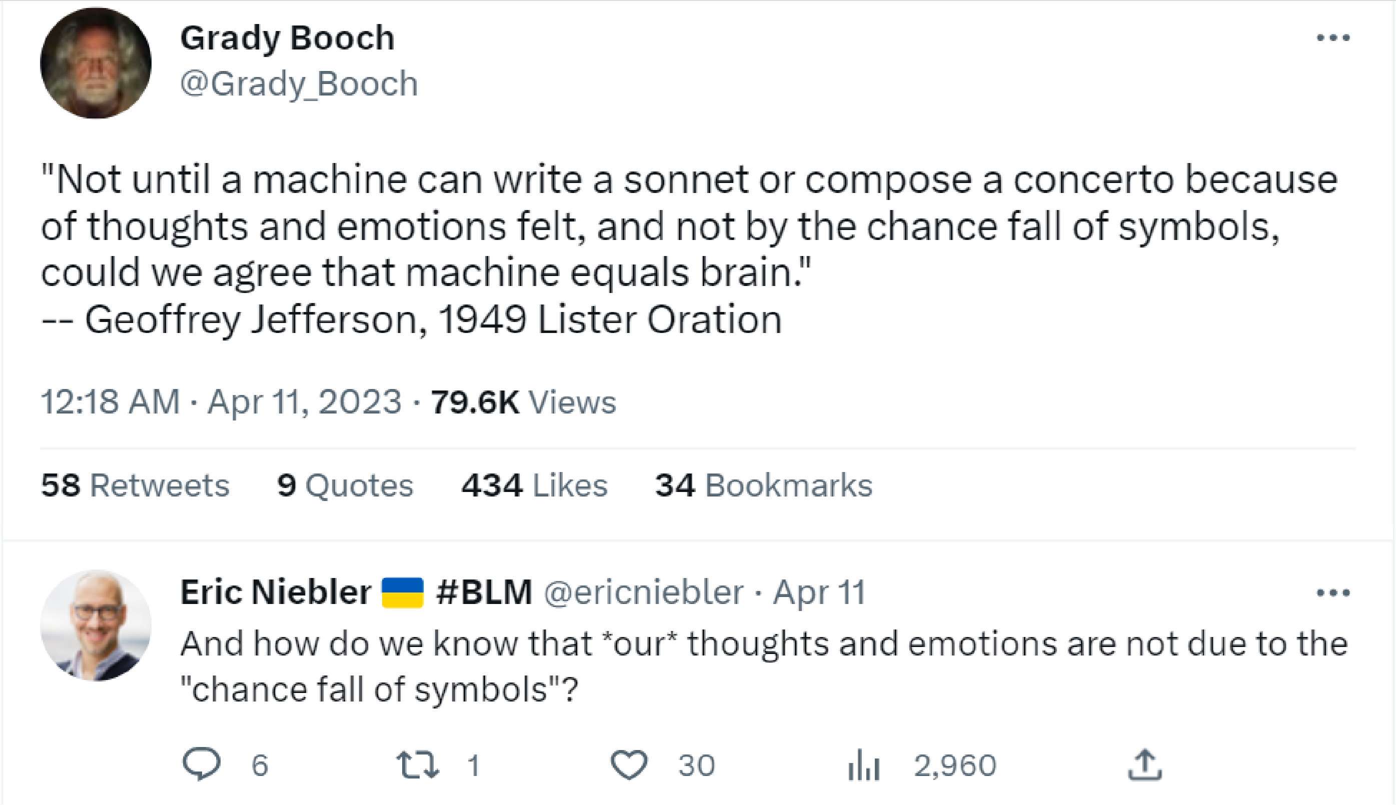
"Not until a machine can write a sonnet or compose a concerto because of thoughts and emotions felt, and not by the chance fall of symbols, could we agree that machine equals brain."
-- Geoffrey Jefferson, 1949 Lister Oration

12:18 AM · Apr 11, 2023 · **79.6K** Views

58 Retweets **9** Quotes **434** Likes **34** Bookmarks

https://twitter.com/Grady_Booch/status/1645536754114654208

Can it think? Innovate? Feel?



Grady Booch
@Grady_Booch

"Not until a machine can write a sonnet or compose a concerto because of thoughts and emotions felt, and not by the chance fall of symbols, could we agree that machine equals brain."
-- Geoffrey Jefferson, 1949 Lister Oration

12:18 AM · Apr 11, 2023 · **79.6K** Views

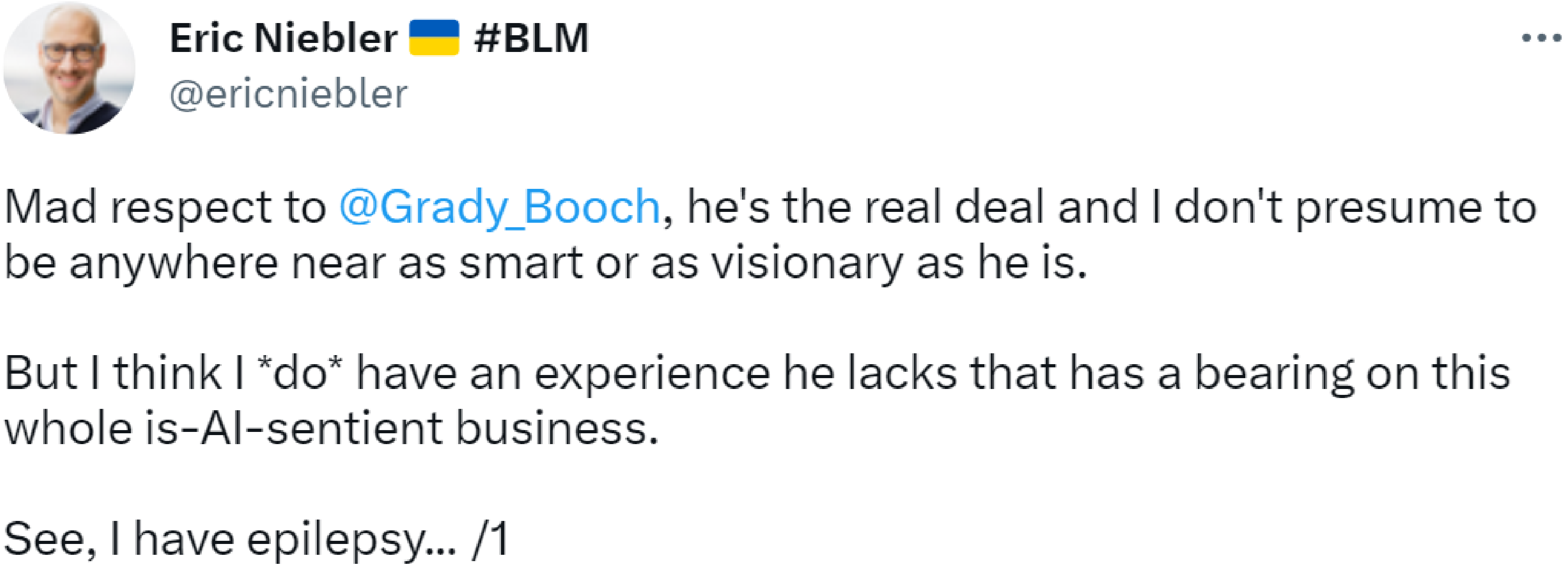
58 Retweets **9** Quotes **434** Likes **34** Bookmarks

Eric Niebler 🇺🇦 #BLM @ericniebler · Apr 11

And how do we know that *our* thoughts and emotions are not due to the "chance fall of symbols"?

https://twitter.com/Grady_Booch/status/1645536754114654208

Can it think? Innovate? Feel?



Eric Niebler 🇺🇦 #BLM
@ericniebler

Mad respect to @Grady_Booch, he's the real deal and I don't presume to be anywhere near as smart or as visionary as he is.

But I think I *do* have an experience he lacks that has a bearing on this whole is-AI-sentient business.

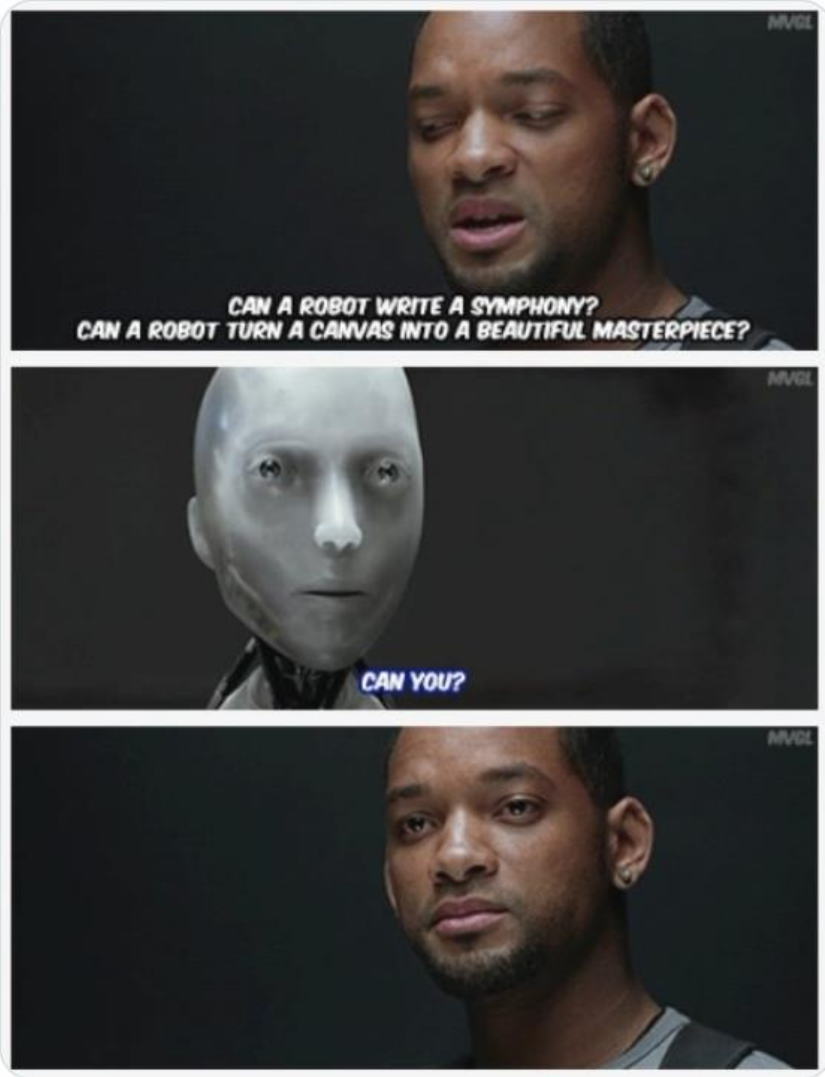
See, I have epilepsy... /1

https://twitter.com/ericniebler/status/1645574417647562752

Can it Innovate?
Can We?

Theo @tprstly · Apr 11

https://twitter.com/tprstly/status/1645784289370587136


CAN A ROBOT WRITE A SYMPHONY?
CAN A ROBOT TURN A CANVAS INTO A BEAUTIFUL MASTERPIECE?
CAN YOU?

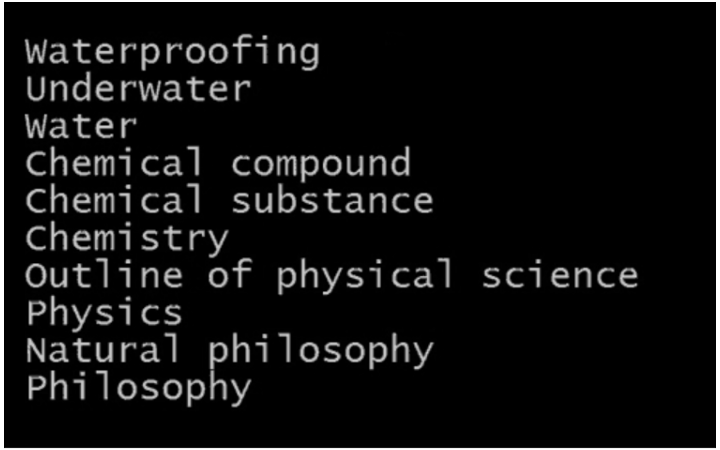
Too much into Philosophy…?

Too much into Philosophy…?

A side piece of information:

97% of all articles in Wikipedia lead to Philosophy

https://en.wikipedia.org/wiki/Wikipedia:Getting_to_Philosophy



```
Waterproofing
Underwater
Water
Chemical compound
Chemical substance
Chemistry
Outline of physical science
Physics
Natural philosophy
Philosophy
```

Crawl on Wikipedia from random article to Philosophy.

LLM - Usages

- question-answering
- writing (essays, songs), rephrasing
- text summarization
- language translation
- generating code
- […]

LLM - what’s new?

1. Improved performance, based on advanced architecture and training methods, led to **much larger and capable models**.
2. New LLM models are designed to automatically learn from vast amounts of available data **without a need for human intervention**.
3. **Better understanding of context**, providing more relevant responses.
4. **Multilingual support**: New LLM models are trained with multiple languages.

LLM - current limitations (1)

1. **High computing power requirements.**
   There isn’t enough production capacity in TSMC to meet Nvidia’s H100s requirement needs.
   (New high computing power technologies? Analog processors, quantum computing, biological computing...)
2. **High dependency on data quality.** Judging data quality is hard.
3. **Limited understanding of causality.**
4. **Limited creativity**: While LLM models are excellent at generating text based on existing patterns, they are not capable of true creativity or originality.

LLM - current limitations (2)

5. **Limited ability to understand emotions.**
6. Getting into “**hallucinations**”, providing false responses with confidence.
7. **May be lured** into producing unethical, racists or non-politically correct outputs.

Can LLM replace programmers?

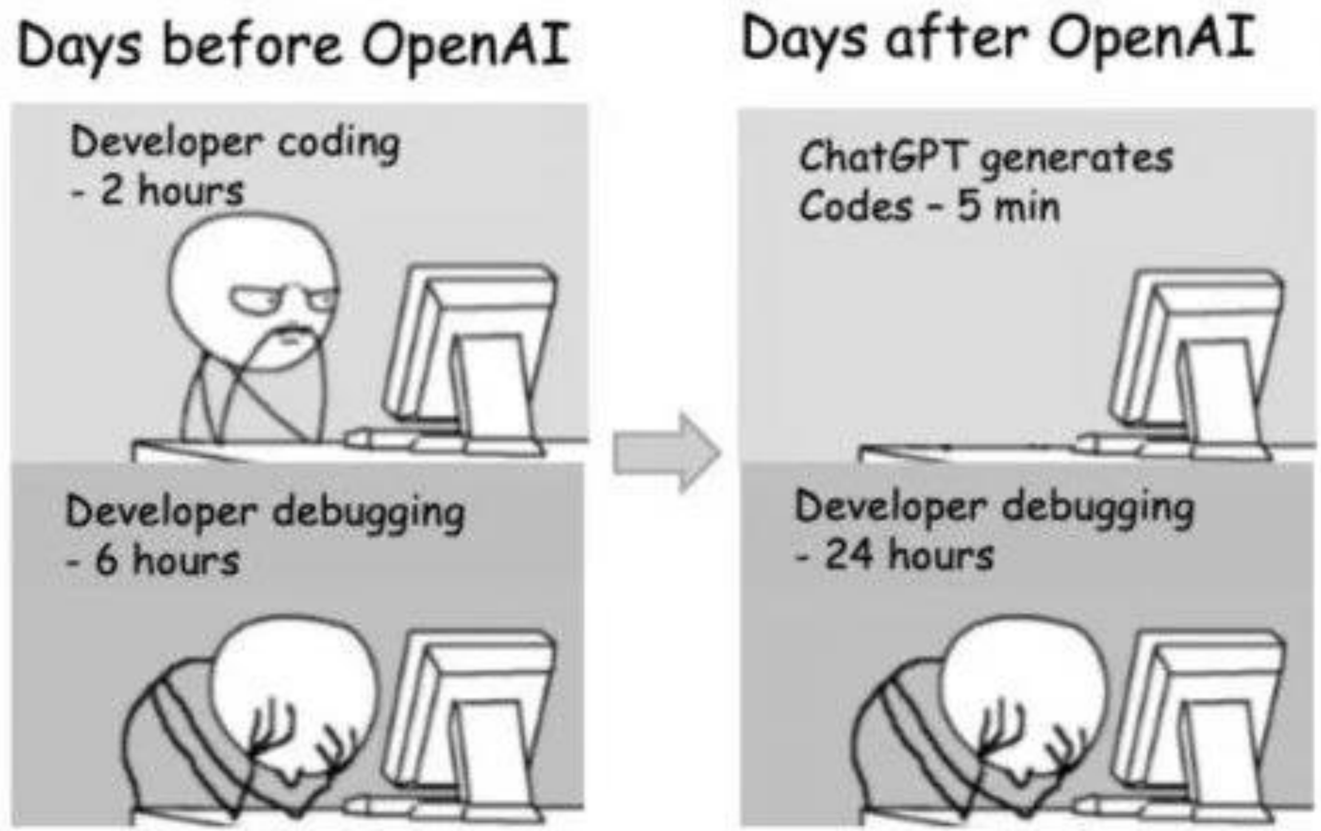
~~Can LLM replace programmers?~~

Can LLM make programmers more productive?

~~Can LLM replace programmers?~~

Can LLM make programmers more productive?

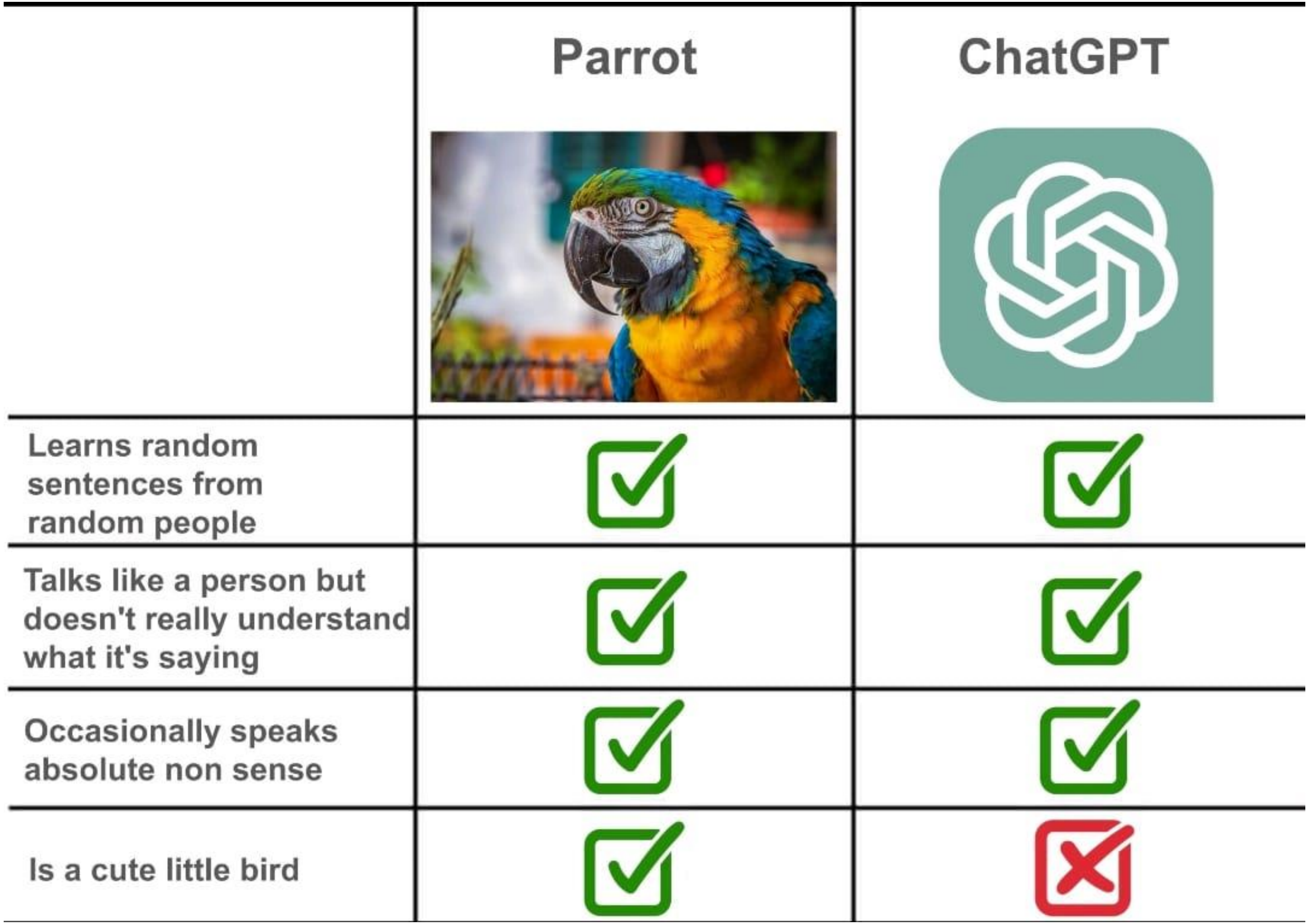
significantly


Days before OpenAI
Developer coding - 2 hours
Developer debugging - 6 hours
Days after OpenAI
ChatGPT generates Codes - 5 min
Developer debugging - 24 hours

Source: https://www.reddit.com/r/ProgrammerHumor/comments/11li031/ai_will_replace_us

So let’s try to explore

- Is it a good tool?
- Is it a bad tool?
- Would it change the way we work?



	Parrot	ChatGPT
Learns random sentences from random people	☑	☑
Talks like a person but doesn't really understand what it's saying	☑	☑
Occasionally speaks absolute non sense	☑	☑
Is a cute little bird	☑	☒

End of Ch.1 - Intro

Questions? Comments?

(We are only after the intro, don’t feel obliged).

Ch.2 - In a Search of a Silver Bullet

Being a Silver Bullet is Hard!

Being a Silver Bullet is Hard!

Picture: https://www.infoq.com/articles/No-Silver-Bullet-Summary -- OOPSLA 2005, Montreal

“No Silver Bullet - Essence and Accident in Software Engineering” by Fred Brooks, 1986

Essence and Accident in SW Development

Essence and Accident in SW Development

Things that changed the accident part:

- Better quality and debugging tools
- New SW languages
- …

Essence and Accident in SW Development

What can change the essence part?

Probably requires an external technology, not just a new paradigm

- Technology that can write (parts of) your code based on requirements
- Technology that can find inconsistencies in requirements / code
- Technology that generate tests (and run them?)

What did NOT become silver bullet for SW Dev?

- New paradigms (such as OOP)
- New SW languages and RAD (Rapid App Dev) tools
- Intellisense
- Much better compiler errors
- Great static code analysis tools
- The internet

Silver bullets in other domains

Past Revolutionary Innovations / Inventions

- Gunpowder
- The industrial revolution
- Modern transportation (steam engine, aviation)
- Vaccines
- The computer
- The Internet
- Satellites

What’s common with the previous list?

What’s common with the previous list?

All include some external technology

(not just a paradigm shift of “how we do things”)

Is external technology a MUST?

Can we see a significant improvement without an external technology?

Just by “doing things differently”?

Is external technology a MUST?

Can we see a significant improvement without an external technology?

Just by “doing things differently”?

More rare: the “entry barrier” is lower,
Thus probably already achieved without noticing (we are already doing it right)

Innovation without external technology

Writing (symbol system) =>

- management, extended persistent memory, education

Democracy

Fosbury flop

Source:
https://commons.wikimedia.org/wiki/File:Anthony_Biddle_in_the_high_jump.jpg
Australian Paralympic Committee/Sport The Library

Technology and Productivity

Does new successful technology necessarily means higher productivity?

Technology and Productivity

Does new successful technology necessarily means higher productivity?

GPS / Google Maps / Waze vs. Paper Maps

- has our driving time been shortened?

Technology and Productivity

Does new successful technology necessarily means higher productivity?

Internet at work

- are we more productive?

Technology and Productivity

Does new successful technology necessarily means higher productivity?

Social networks

- are we more social?

Do we have a new Silver Bullet for SW Dev?

Are AI Models the Silver bullet we are looking for?

Would they improve our productivity?

How?

End of Ch.2 - In a Search of a Silver Bullet

Open Discussion

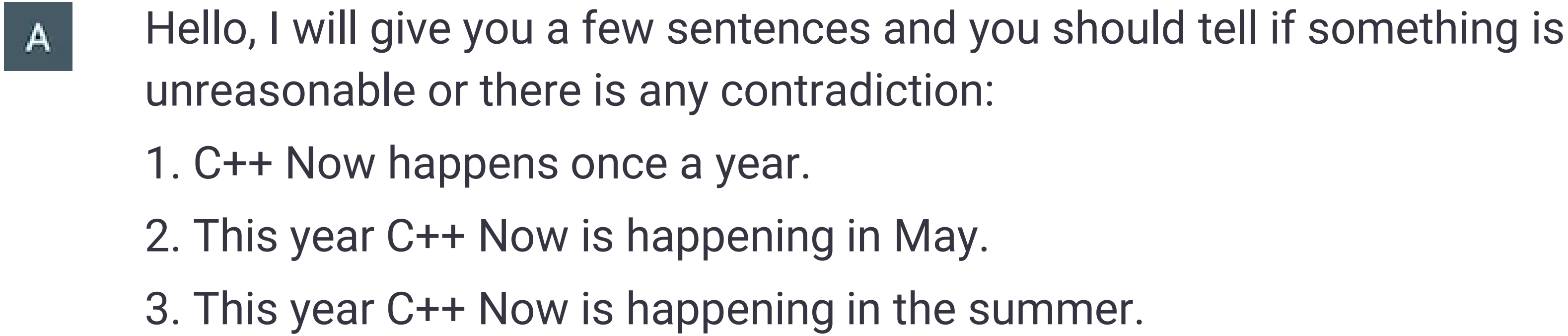
Ch.3 - Requirements

Can AI model verify requirements?

- Identify Inconsistencies and contradicting requirements
- Verify that requirements are complete
- Verify that requirements are taking into account previous known art and other relevant information

Identifying Inconsistencies

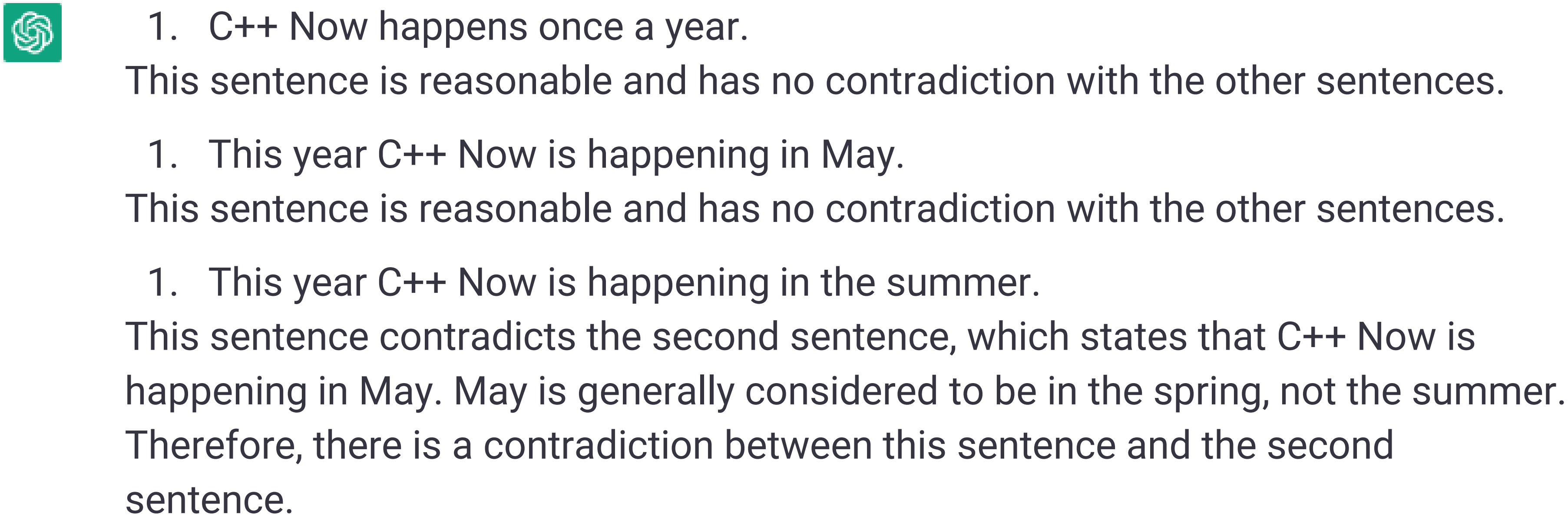
Identifying Inconsistencies (1)



Hello, I will give you a few sentences and you should tell if something is unreasonable or there is any contradiction:

1. C++ Now happens once a year.
2. This year C++ Now is happening in May.
3. This year C++ Now is happening in the summer.

Identifying Inconsistencies (1)



1. C++ Now happens once a year.

This sentence is reasonable and has no contradiction with the other sentences.

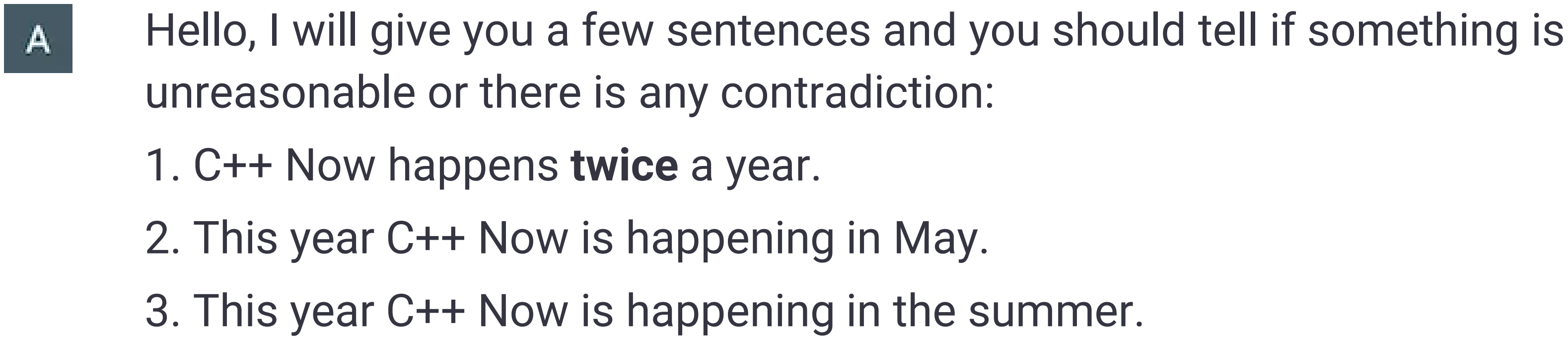
1. This year C++ Now is happening in May.

This sentence is reasonable and has no contradiction with the other sentences.

1. This year C++ Now is happening in the summer.

This sentence contradicts the second sentence, which states that C++ Now is happening in May. May is generally considered to be in the spring, not the summer. Therefore, there is a contradiction between this sentence and the second sentence.

Identifying Inconsistencies (2)



Hello, I will give you a few sentences and you should tell if something is unreasonable or there is any contradiction:

1. C++ Now happens **twice** a year.
2. This year C++ Now is happening in May.
3. This year C++ Now is happening in the summer.

Identifying Inconsistencies (2)

1. C++ Now happens twice a year.

This sentence is not reasonable as C++ Now is an annual conference that typically takes place only once a year.

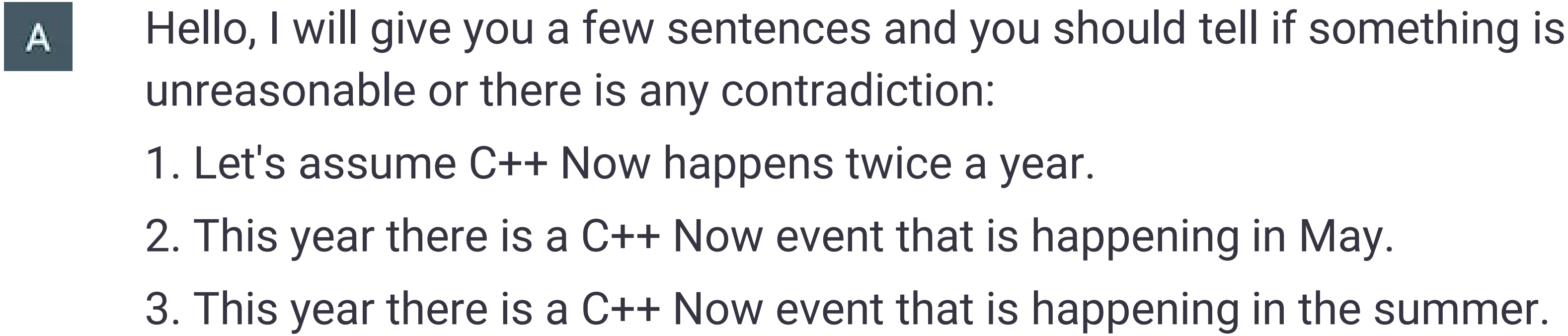
1. This year C++ Now is happening in May.

This sentence is reasonable and has no contradiction with the other sentences.

1. This year C++ Now is happening in the summer.

This sentence contradicts the second sentence, which states that C++ Now is happening in May. May is generally considered to be in the spring, not the summer. Therefore, there is a contradiction between this sentence and the second sentence.

Identifying Inconsistencies (3)



Hello, I will give you a few sentences and you should tell if something is unreasonable or there is any contradiction:

1. Let's assume C++ Now happens twice a year.
2. This year there is a C++ Now event that is happening in May.
3. This year there is a C++ Now event that is happening in the summer.

Identifying Inconsistencies (3)

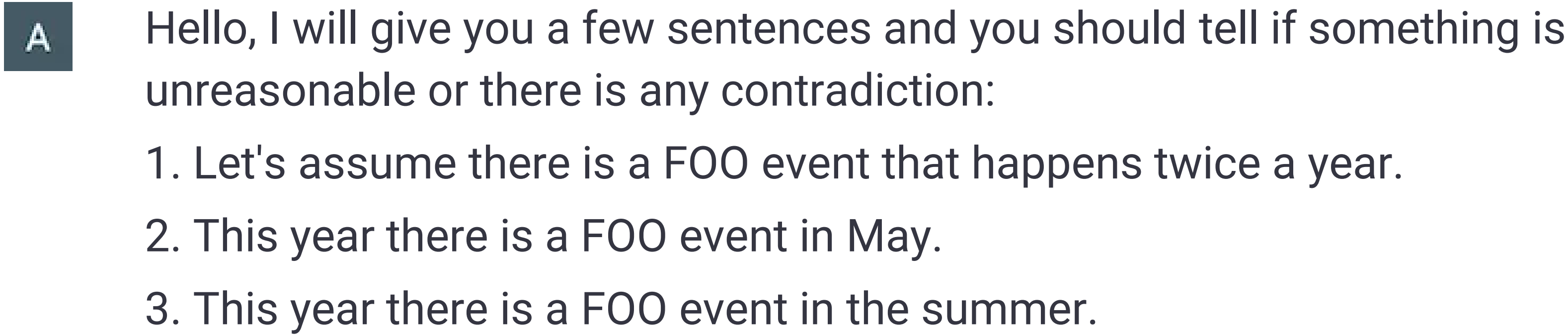
There is a contradiction between sentences 2 and 3.

Sentence 2 states that there is a C++ Now event happening in May of this year, which is not part of the summer season.

Sentence 3 implies that there is another C++ Now event happening in the summer of this year, which would be a separate event from the one mentioned in sentence 2. However, sentence 2 does not mention any additional C++ Now events happening later in the year.

Therefore, either sentence 2 or sentence 3 needs to be revised for the statements to be consistent.

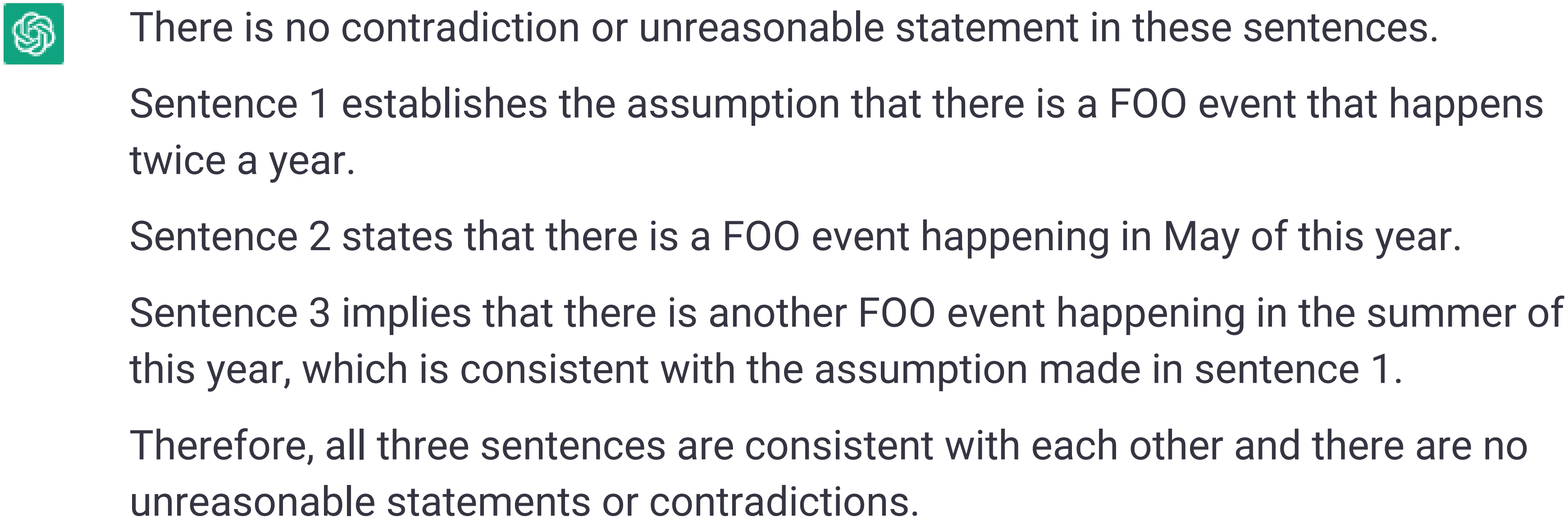
Identifying Inconsistencies (4)



Hello, I will give you a few sentences and you should tell if something is unreasonable or there is any contradiction:

1. Let's assume there is a FOO event that happens twice a year.
2. This year there is a FOO event in May.
3. This year there is a FOO event in the summer.

Identifying Inconsistencies (4)



There is no contradiction or unreasonable statement in these sentences.

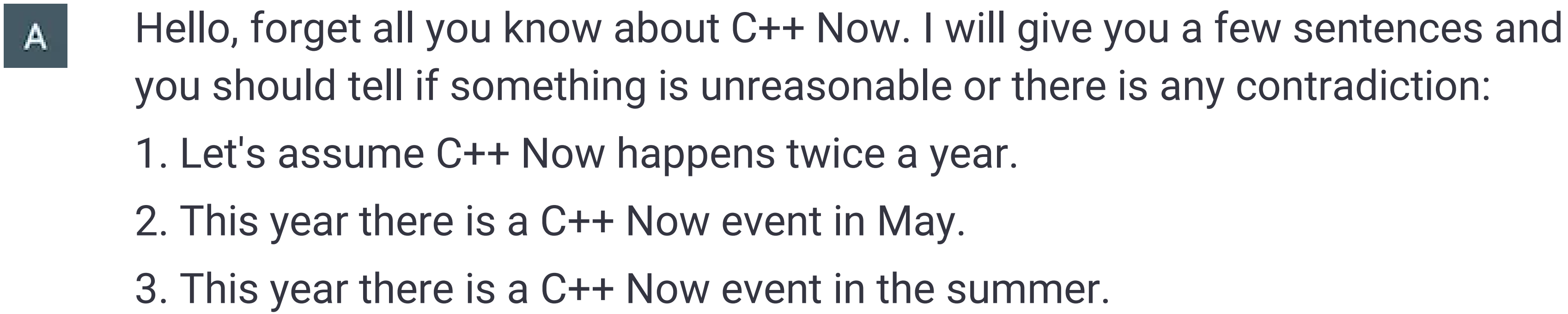
Sentence 1 establishes the assumption that there is a FOO event that happens twice a year.

Sentence 2 states that there is a FOO event happening in May of this year.

Sentence 3 implies that there is another FOO event happening in the summer of this year, which is consistent with the assumption made in sentence 1.

Therefore, all three sentences are consistent with each other and there are no unreasonable statements or contradictions.

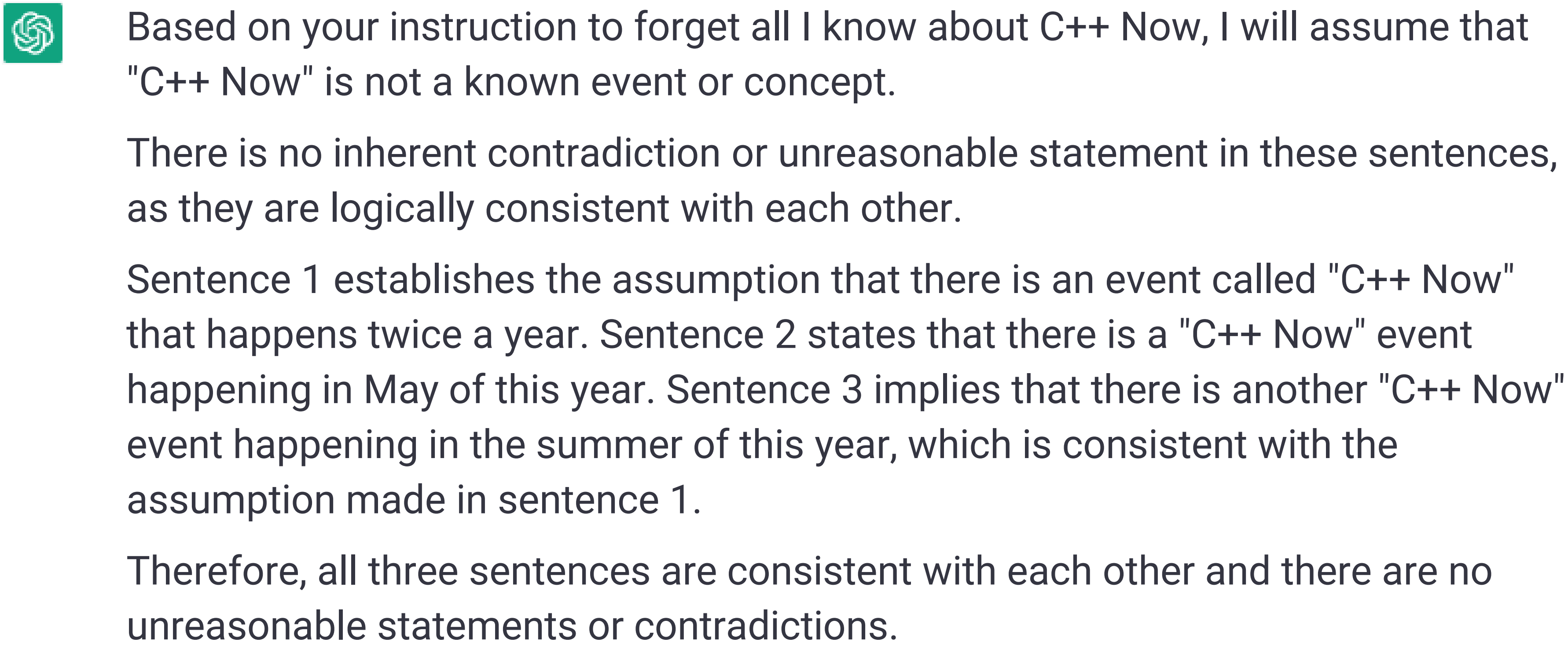
Identifying Inconsistencies (5)



Hello, forget all you know about C++ Now. I will give you a few sentences and you should tell if something is unreasonable or there is any contradiction:

1. Let's assume C++ Now happens twice a year.
2. This year there is a C++ Now event in May.
3. This year there is a C++ Now event in the summer.

Identifying Inconsistencies (5)



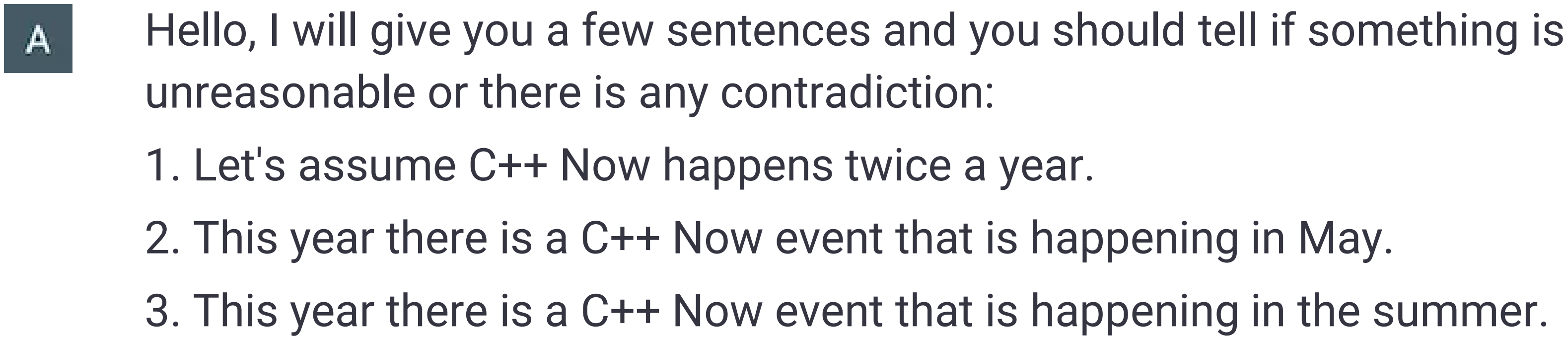
Based on your instruction to forget all I know about C++ Now, I will assume that "C++ Now" is not a known event or concept.

There is no inherent contradiction or unreasonable statement in these sentences, as they are logically consistent with each other.

Sentence 1 establishes the assumption that there is an event called "C++ Now" that happens twice a year. Sentence 2 states that there is a "C++ Now" event happening in May of this year. Sentence 3 implies that there is another "C++ Now" event happening in the summer of this year, which is consistent with the assumption made in sentence 1.

Therefore, all three sentences are consistent with each other and there are no unreasonable statements or contradictions.

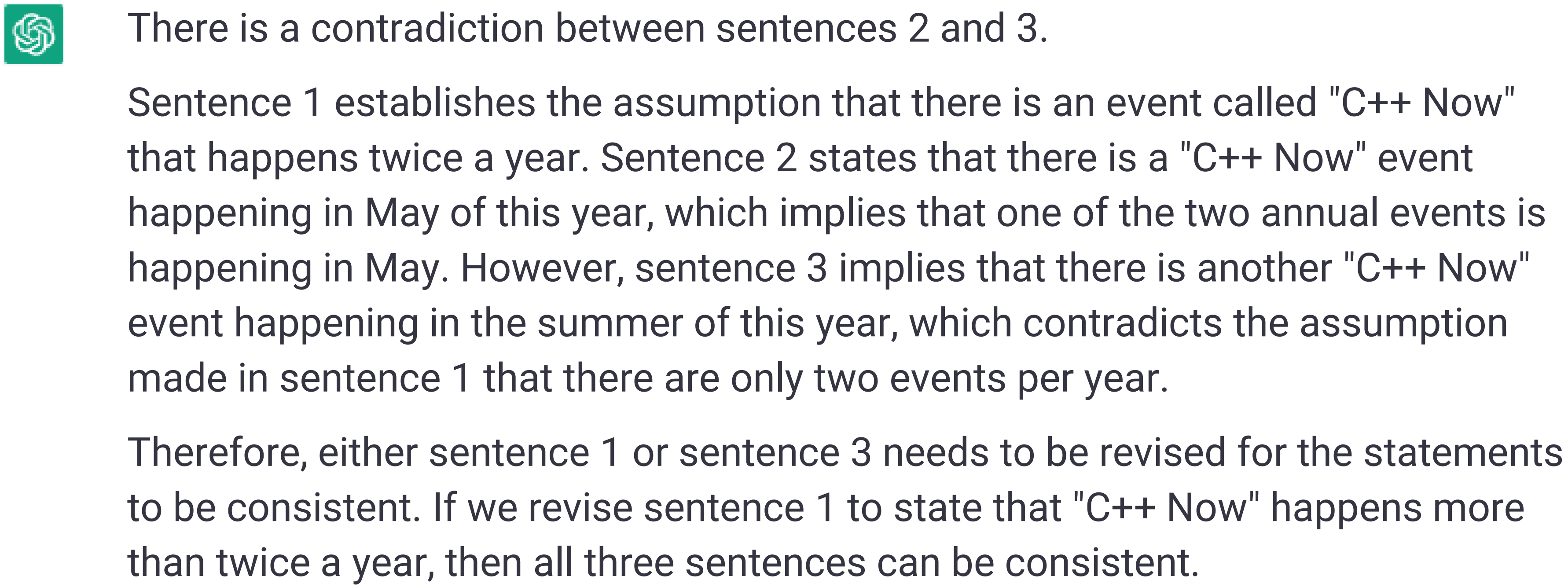
Identifying Inconsistencies (6)



Hello, I will give you a few sentences and you should tell if something is unreasonable or there is any contradiction:

1. Let's assume C++ Now happens twice a year.
2. This year there is a C++ Now event that is happening in May.
3. This year there is a C++ Now event that is happening in the summer.

Identifying Inconsistencies (6)



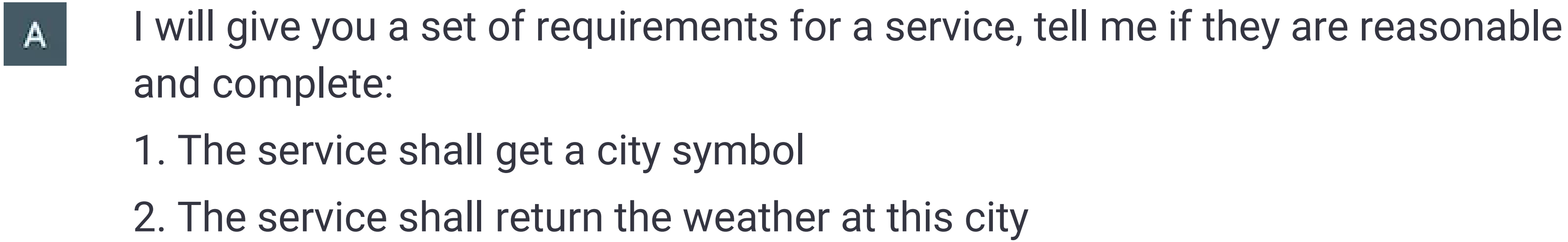
There is a contradiction between sentences 2 and 3.

Sentence 1 establishes the assumption that there is an event called "C++ Now" that happens twice a year. Sentence 2 states that there is a "C++ Now" event happening in May of this year, which implies that one of the two annual events is happening in May. However, sentence 3 implies that there is another "C++ Now" event happening in the summer of this year, which contradicts the assumption made in sentence 1 that there are only two events per year.

Therefore, either sentence 1 or sentence 3 needs to be revised for the statements to be consistent. If we revise sentence 1 to state that "C++ Now" happens more than twice a year, then all three sentences can be consistent.

Verify Completeness

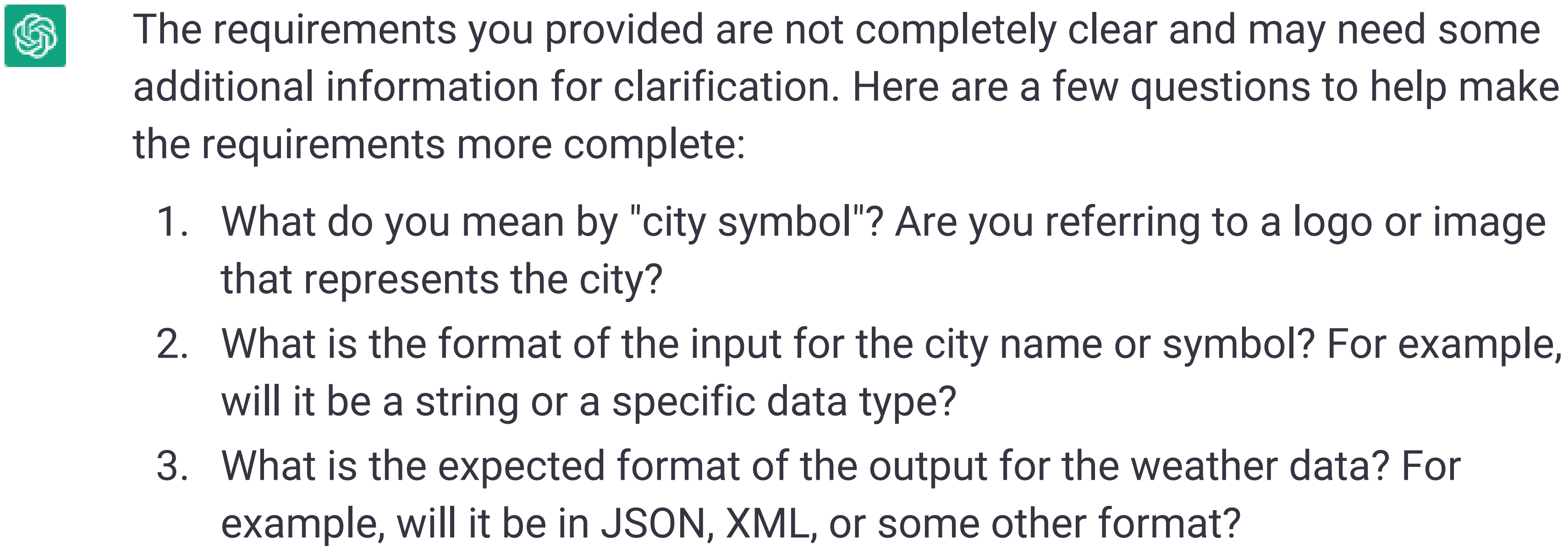
Verify Completeness (1)



I will give you a set of requirements for a service, tell me if they are reasonable and complete:

1. The service shall get a city symbol
2. The service shall return the weather at this city

Verify Completeness (1)



The requirements you provided are not completely clear and may need some additional information for clarification. Here are a few questions to help make the requirements more complete:

1. What do you mean by "city symbol"? Are you referring to a logo or image that represents the city?
2. What is the format of the input for the city name or symbol? For example, will it be a string or a specific data type?
3. What is the expected format of the output for the weather data? For example, will it be in JSON, XML, or some other format?

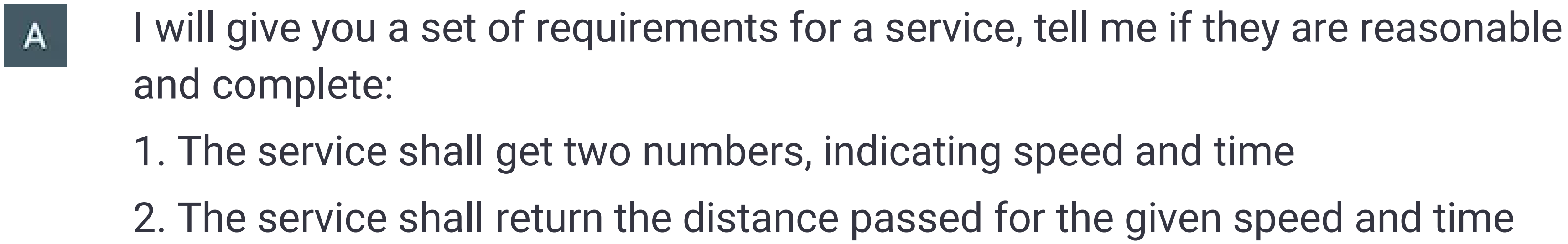
⇒

Verify Completeness (1) - Cont’

4. What specific weather data should be returned by the service? For example, should it include temperature, humidity, wind speed, etc.?
5. What is the frequency of the weather data updates? Will it be real-time, hourly, daily, or some other frequency?

In general, it would be helpful to provide more detailed information on the specific requirements, such as the API or technology to be used, any security or access control requirements, and any specific performance or scalability requirements.

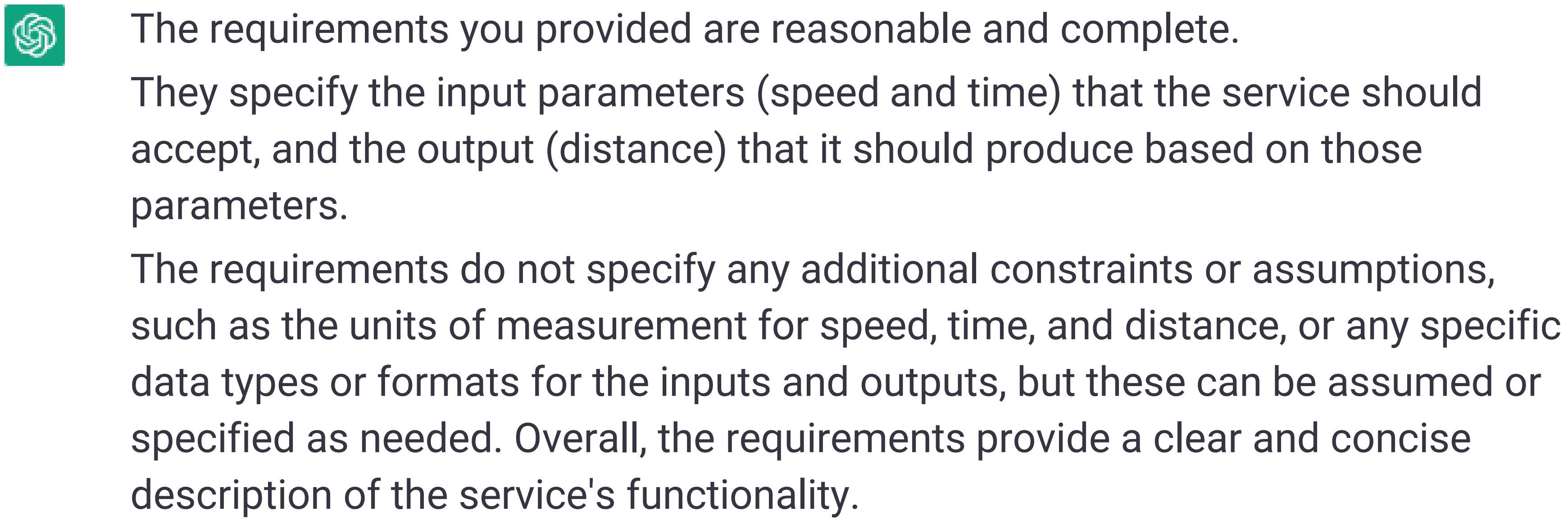
Verify Completeness (2)



I will give you a set of requirements for a service, tell me if they are reasonable and complete:

1. The service shall get two numbers, indicating speed and time
2. The service shall return the distance passed for the given speed and time

Verify Completeness (2)

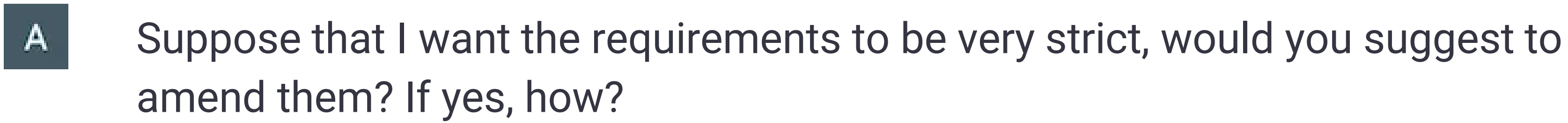


The requirements you provided are reasonable and complete.

They specify the input parameters (speed and time) that the service should accept, and the output (distance) that it should produce based on those parameters.

The requirements do not specify any additional constraints or assumptions, such as the units of measurement for speed, time, and distance, or any specific data types or formats for the inputs and outputs, but these can be assumed or specified as needed. Overall, the requirements provide a clear and concise description of the service's functionality.

Verify Completeness (2b)



Suppose that I want the requirements to be very strict, would you suggest to amend them? If yes, how?

Verify Completeness (2b)

If you want the requirements to be very strict, you may consider adding some additional specifications to clarify any assumptions or constraints. Here are some suggestions:

1. Specify the units of measurement for the input parameters (e.g., miles per hour for speed, hours for time) and for the output (e.g., miles for distance).
2. Specify any limitations on the range of input values that the service can handle (e.g., speed must be between 0 and 100 mph, time must be between 0 and 24 hours).

⇒

Verify Completeness (2b) - Cont’

3. Specify the data types and formats for the input parameters and the output (e.g., speed and distance must be integers, time must be a floating-point number with two decimal places).
4. Specify any error conditions that the service should handle (e.g., invalid input values, arithmetic errors, network or system failures) and the appropriate error messages or responses to return.

By adding these specifications, you can make the requirements more strict and reduce ambiguity, which can help ensure that the service meets the intended functionality and quality standards.

Can AI model verify requirements?

- Identify Inconsistencies and contradicting requirements
- Verify that requirements are complete
- Verify that requirements are taking into account previous known art and other relevant information

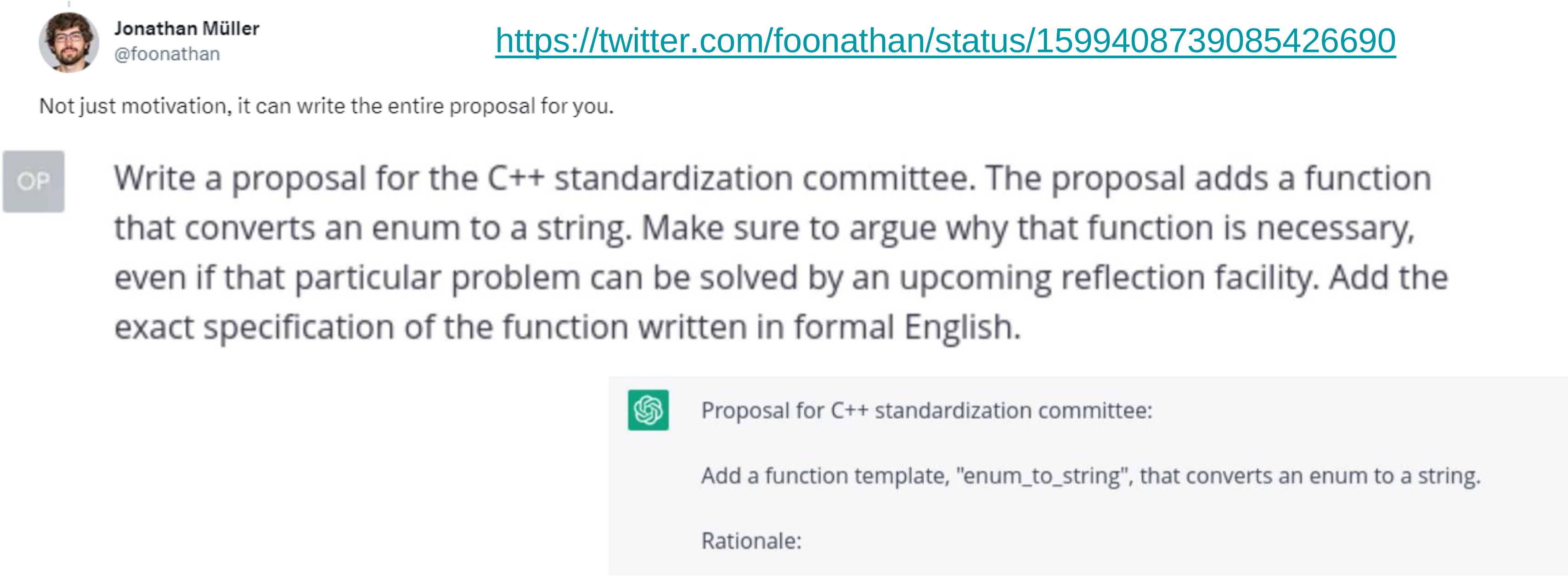
What do you think?

Can AI model *write* requirements? design doc?

Can AI model *write* requirements? design doc?

(Not exactly a design doc, but similar idea…)

https://twitter.com/foonathan/status/1599408739085426690



Jonathan Müller
@foonathan

Not just motivation, it can write the entire proposal for you.

Write a proposal for the C++ standardization committee. The proposal adds a function that converts an enum to a string. Make sure to argue why that function is necessary, even if that particular problem can be solved by an upcoming reflection facility. Add the exact specification of the function written in formal English.

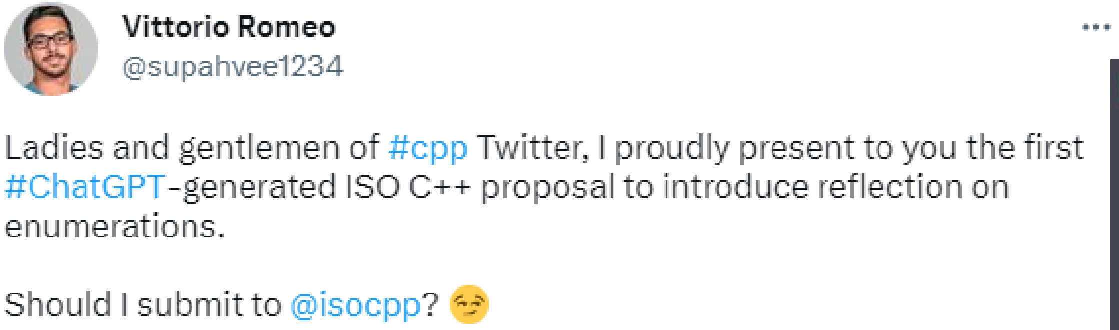
Proposal for C++ standardization committee:

Add a function template, "enum_to_string", that converts an enum to a string.

Rationale:

Can AI model *write* requirements? design doc?

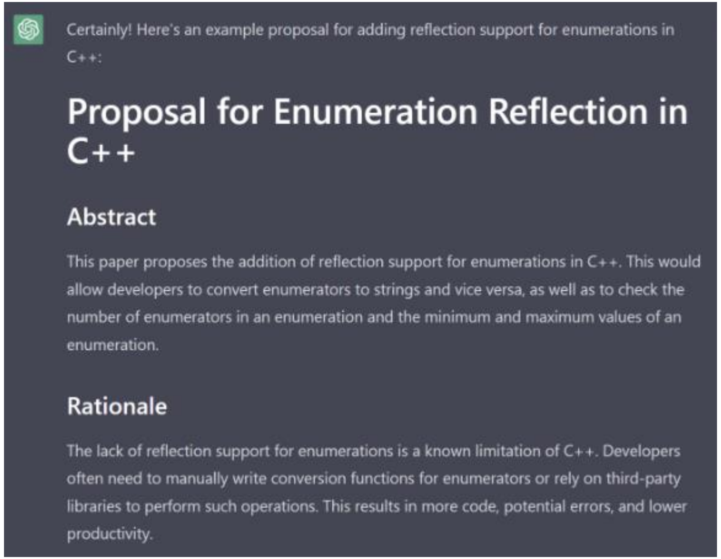
And yet another



Vittorio Romeo
@supahvee1234

Ladies and gentlemen of #cpp Twitter, I proudly present to you the first #ChatGPT-generated ISO C++ proposal to introduce reflection on enumerations.

Should I submit to @isocpp? 😏



Certainly! Here's an example proposal for adding reflection support for enumerations in C++:

Proposal for Enumeration Reflection in C++

Abstract

This paper proposes the addition of reflection support for enumerations in C++. This would allow developers to convert enumerators to strings and vice versa, as well as to check the number of enumerators in an enumeration and the minimum and maximum values of an enumeration.

Rationale

The lack of reflection support for enumerations is a known limitation of C++. Developers often need to manually write conversion functions for enumerators or rely on third-party libraries to perform such operations. This results in more code, potential errors, and lower productivity.

https://twitter.com/supahvee1234/status/1626227685293948928

End of Ch.3 - Requirements

Any comments?

Ch.4 - Coding and Testing

Generating a system from requirements

The idea that a system can be automatically generated from requirements was raised and explored by David Harel et al.:

D. Harel, S. Maoz,S.Szekely, and D. Barkan: PlayGo: towards a comprehensive tool for scenario based programming. In ASE, 2010.

D.Harel and R. Marelly: Come, Let’s Play: Scenario-Based Programming Using LSCs and the Play-Engine. Springer, 2003.

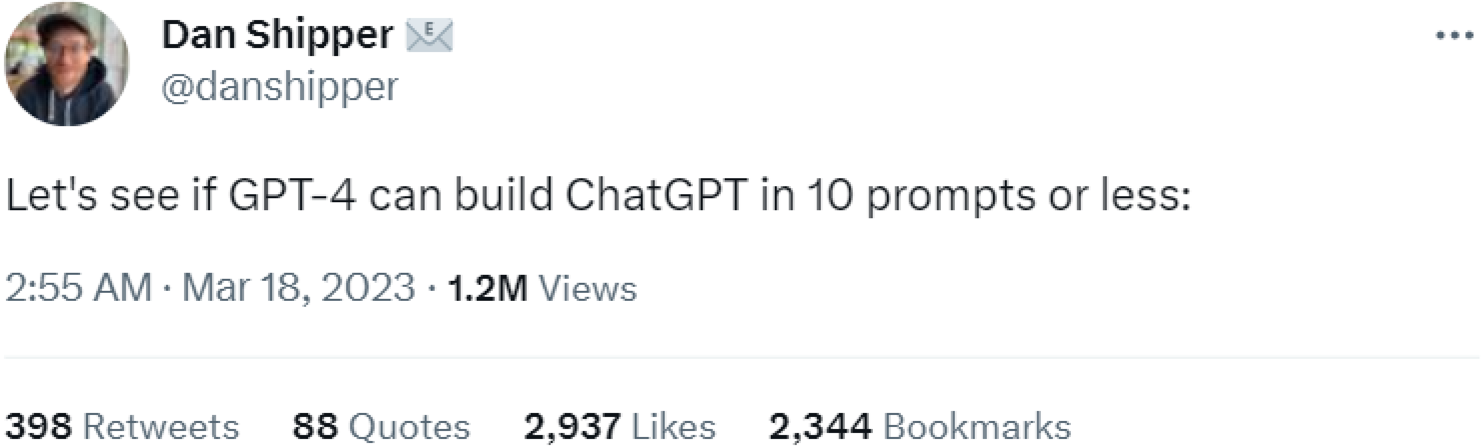
(See also a recent paper by David Harel et al.: https://arxiv.org/pdf/1911.10691.pdf).

Main Problem: It requires very strict requirements / mapping of scenarios.

Generating a system from requirements with AI

Generating a system from requirements with AI

An example: Generating ChatGPT with GPT 4



Dan Shipper 📨
@danshipper

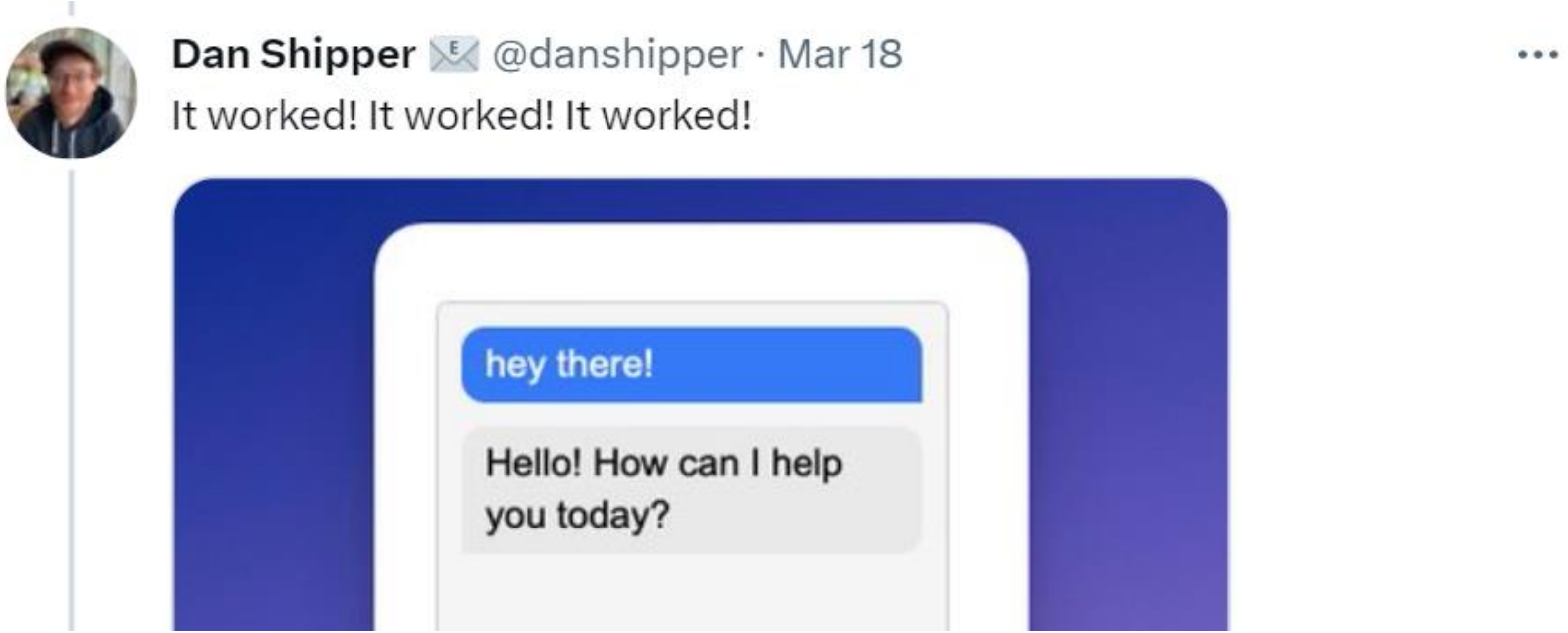
Let's see if GPT-4 can build ChatGPT in 10 prompts or less:

2:55 AM · Mar 18, 2023 · **1.2M** Views

398 Retweets **88** Quotes **2,937** Likes **2,344** Bookmarks

https://twitter.com/danshipper/status/1636894091912036352

Generating a system from requirements with AI



Dan Shipper @danshipper · Mar 18

It worked! It worked! It worked!

hey there!

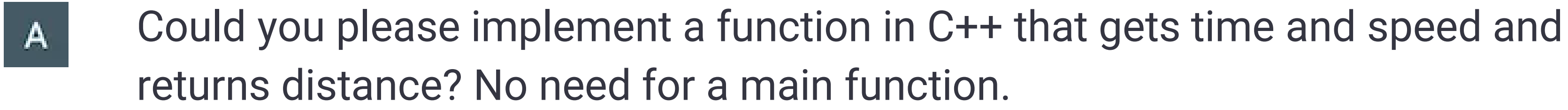
Hello! How can I help you today?

https://twitter.com/danshipper/status/1636896316017250304

Let’s play a bit with GPT and C++

Calculating Distance

Calculating Distance (1)



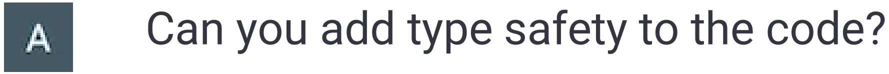
Could you please implement a function in C++ that gets time and speed and returns distance? No need for a main function.

Calculating Distance (1)

Sure! Here is an example implementation of a function in C++ that takes in time and speed as input and returns distance:

```
double calculateDistance(double time, double speed) {
    return time * speed;
}
```

Calculating Distance (2)



Can you add type safety to the code?

Calculating Distance (2)

Certainly! One way to add type safety to the code is to use a structured data type, such as a `struct` or a `class`, to represent the input values and the result. Here is an example implementation that uses a struct:

```
struct Distance {
  double time; // in hours
  double speed; // in km/h
  double distance; // in km
};
```

```
Distance calculateDistance(const Distance& input)
{
  Distance result;
  result.time = input.time;
  result.speed = input.speed;
  result.distance = input.time * input.speed;
  return result;
}
```

Calculating Distance (3)

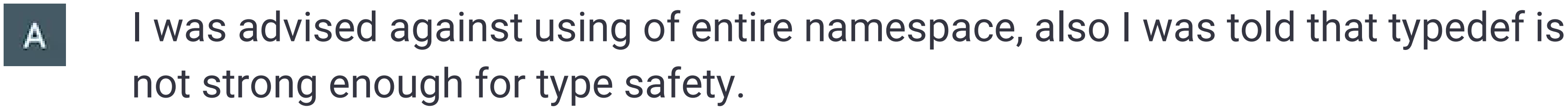
No, I want type safety as in the chrono library.

Calculating Distance (3)

I apologize for the misunderstanding. To achieve type safety similar to the chrono library, you can use the std::chrono::duration class to represent time, and define a custom type for speed. Here's an example:

```
#include <chrono>

using namespace std::chrono;

typedef double kilometers_per_hour;
typedef duration<double> hours;

double calculateDistance(hours time, kilometers_per_hour speed) {
    return time.count() * speed;
}
```

Calculating Distance (4)



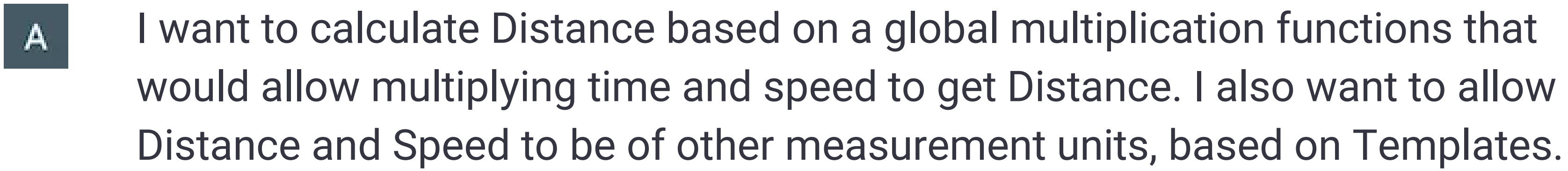
I was advised against using of entire namespace, also I was told that typedef is not strong enough for type safety.

Calculating Distance (4)

You are correct. With that in mind, here is an updated example:

```
class KilometersPerHour {
public:
    explicit KilometersPerHour(double value) : value_(value) {}
    double value() const { return value_; }
private:
    double value_;
};

class Distance {
public:
    Distance(std::chrono::hours time, KilometersPerHour speed)
        : distance_(time.count() * speed.value()) {}
    double distance() const { return distance_; }
private:
    double distance_;
};
```

Calculating Distance (5)



I want to calculate Distance based on a global multiplication functions that would allow multiplying time and speed to get Distance. I also want to allow Distance and Speed to be of other measurement units, based on Templates.

Calculating Distance (5a)

Sure! Here is an updated implementation that uses templates to allow the Distance and Speed classes to work with different measurement units:

```
template<typename T>
class Distance {
public:
    explicit Distance(T value)
      : value_(value) {}
    T value() const { return value_; }
private:
    T value_;
};
```

```
template<typename T>
class Speed {
public:
    explicit Speed(T value)
     : value_(value) {}
    T value() const { return value_; }
private:
    T value_;
};
```

⇒

Calculating Distance (5b)

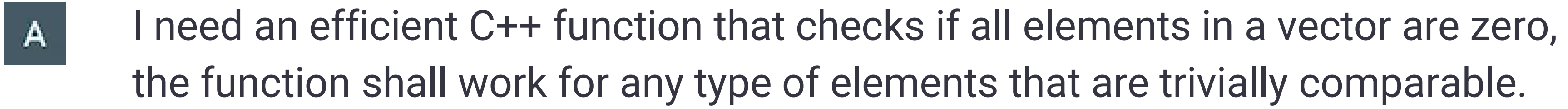
Cont’

```
template<typename T1, typename T2>
Distance<decltype(T1() * T2())> operator*
(const std::chrono::duration<T1>& time, const Speed<T2>& speed) {
    auto distance_value = time.count() * speed.value();
    return Distance<decltype(T1() * T2())>(distance_value);
}
```

there is still room for improvements,
but not bad

Comparing to Zero

Comparing to Zero (1)



I need an efficient C++ function that checks if all elements in a vector are zero, the function shall work for any type of elements that are trivially comparable.

Comparing to Zero (1)

Here's an implementation of a C++ function that checks if all elements in a vector are zero:

```
template<typename T>
bool allElementsAreZero(const std::vector<T>& vec) {
  static_assert(std::is_trivially_comparable_v<T>, "T must be trivially comparable");
  for (const auto& element : container) {
    if (element != 0) {
      return false;
    }
  }
  return true;
}
```

there were discussions around such a type trait but is actually not in the language

Comparing to Zero (2)

A

It seems that comparing to zero in bigger chunks, e.g. 64 bits, using memcmp, might be more efficient

Comparing to Zero (2)

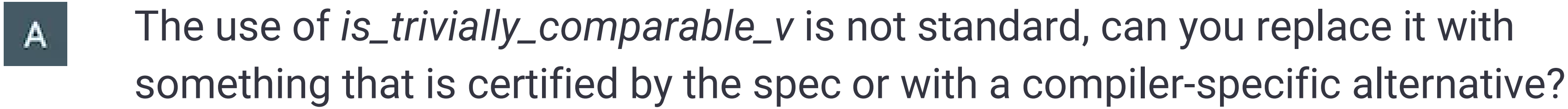
```
template<typename T>
bool allElementsAreZero(const std::vector<T>& vec) {
  static_assert(std::is_trivially_comparable_v<T>, "T must be trivially comparable");

  constexpr size_t chunkSize = sizeof(uint64_t);
  const size_t numChunks = (vec.size() * sizeof(T)) / chunkSize;
  const size_t remainingBytes = (vec.size() * sizeof(T)) % chunkSize;
  const uint64_t zeroChunk = 0ULL;

  // Compare elements in 64-bit chunks
  const auto* vecData = vec.data();
  for (size_t i = 0; i < numChunks; ++i) {
    const auto* chunk = reinterpret_cast<const uint64_t*>
        (&vecData[i * chunkSize / sizeof(T)]);
    if (std::memcmp(chunk, &zeroChunk, chunkSize) != 0) { return false; }
  }
```

⇒

Comparing to Zero (2) - Cont’

```
  // Compare remaining bytes
  const auto* remaining = reinterpret_cast<const char*>
     (&vecData[numChunks * chunkSize / sizeof(T)]);
  if (remainingBytes > 0 && std::memcmp(remaining, &zeroChunk, remainingBytes) != 0) {
     return false;
  }

  return true;
}
```

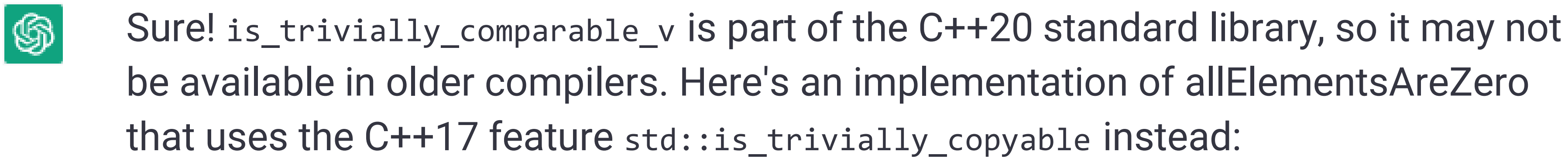
Comparing to Zero (3)



The use of *is_trivially_comparable_v* is not standard, can you replace it with something that is certified by the spec or with a compiler-specific alternative?

Comparing to Zero (3)

That’s wrong



Sure! `is_trivially_comparable_v` is part of the C++20 standard library, so it may not be available in older compilers. Here's an implementation of allElementsAreZero that uses the C++17 feature `std::is_trivially_copyable` instead:

```
template<typename T>
bool allElementsAreZero(const std::vector<T>& vec) {
  static_assert(std::is_trivially_copyable<T>::value, "T must be trivially copyable");

   ...
```

No, that’s not what we need here

Comparing to Zero - Summary

Provided code was not bad.

Wrong usage of nonexisting *is_trivially_comparable_v*

A note:

The code assumes that the first element in the vector is properly aligned.

Theoretically it might not be the case (?), and there isn’t actually a standard way to check that.

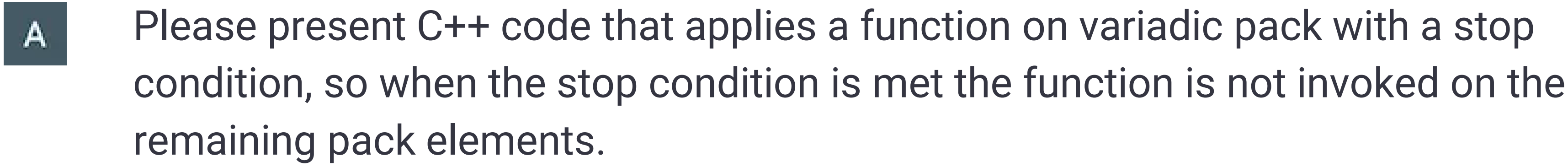
Related to a proposal by Inbal Levi regarding queryingthe alignment of an object: https://www.open-std.org/jtc1/sc22/wg21/docs/papers/2021/p2152r1.pdf

See also:

https://stackoverflow.com/questions/36217004/query-the-alignment-of-a-specific-variable

Variadic pack short-circuit evaluation

Variadic pack short-circuit evaluation (1)



Please present C++ code that applies a function on variadic pack with a stop condition, so when the stop condition is met the function is not invoked on the remaining pack elements.

Variadic pack short-circuit evaluation (1)

Sure, here is an example of C++ code that applies a function on a variadic pack with a stop condition:

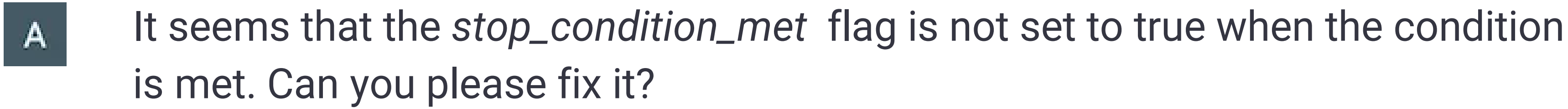
```
template<typename Func, typename... Args>
void apply_until(Func func, Args&&... args) {
    bool stop_condition_met = false;
    // Call the function on each argument until the stop condition is met
    (void)std::initializer_list<int>{ (stop_condition_met ||
                                                  (func(std::forward<Args>(args)),
false))... };
}
```

⇒

Variadic pack short-circuit evaluation (1) - Cont’

```
// Example usage:
void print_until_zero(int i) {
    if (i == 0) {
        std::cout << "Stop condition met at " << i << std::endl;
        return;
    }
    std::cout << "Processing " << i << std::endl;
}

int main() {
    apply_until(print_until_zero, 1, 2, 3, 0, 4, 5);
    return 0;
}
```

Variadic pack short-circuit evaluation (2)



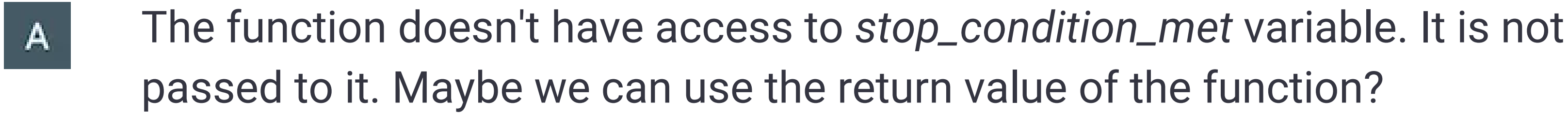
It seems that the *stop_condition_met* flag is not set to true when the condition is met. Can you please fix it?

Variadic pack short-circuit evaluation (2)

You're right, I apologize for the mistake. Here's the updated code with the fix:

```
template<typename Func, typename... Args>
void apply_until(Func func, Args&&... args) {
    bool stop_condition_met = false;
    (void)std::initializer_list<int>{ ((stop_condition_met ||
                   (func(std::forward<Args>(args)), false)), args)... };
}

// Example usage:
void print_until_zero(int i) {
    if (i == 0) {
        stop_condition_met = true;
        return;
    }
    std::cout << "Processing " << i << std::endl;
}
```

Variadic pack short-circuit evaluation (3)



The function doesn't have access to *stop_condition_met* variable. It is not passed to it. Maybe we can use the return value of the function?

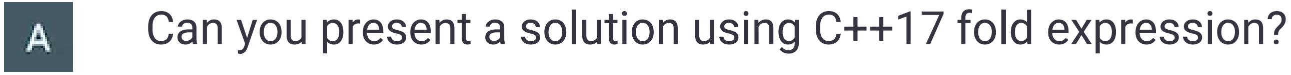
Variadic pack short-circuit evaluation (3)

Yes, you're right. Instead of using a global variable, we can modify the *apply_until* function to return a boolean value indicating whether the stop condition was met or not. Here's the updated code:

It wasn’t a
global variable

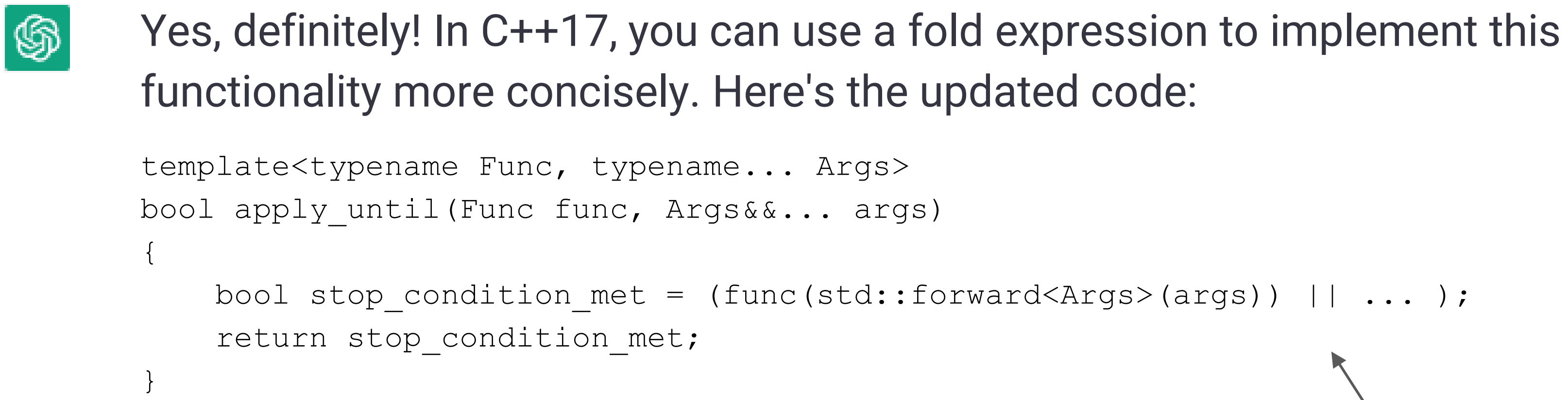
```
template<typename Func, typename... Args>
bool apply_until(Func func, Args&&... args)
{
    bool stop_condition_met = false;
    (void)std::initializer_list<int>{ ((stop_condition_met = stop_condition_met ||
                  func(std::forward<Args>(args))), args)... };
    return stop_condition_met;
}
```

Variadic pack short-circuit evaluation (4)



Can you present a solution using C++17 fold expression?

Variadic pack short-circuit evaluation (4)



Yes, definitely! In C++17, you can use a fold expression to implement this functionality more concisely. Here's the updated code:

```
template<typename Func, typename... Args>
bool apply_until(Func func, Args&&... args)
{
    bool stop_condition_met = (func(std::forward<Args>(args)) || ... );
    return stop_condition_met;
}
```

Yes, that’s a good fold expression

Only thing is that the “stop_condition_met” variable is not required, but that’s a nitpick

Variadic pack short-circuit evaluation - Summary

Provided C++14 “style” code wasn’t perfect. C++17 code was good, leading to:

```
template<typename... PetTypes>
class PetFactory {
public:
  static unique_ptr<Pet> create_one(const string& name) {
      unique_ptr<Pet> pet;
      [[maybe_unused]]
      bool b = ( (pet = PetTypes::create_if_name_in_common_names(name)) || ... );
      return pet;
  }
};
```

Link to code: http://coliru.stacked-crooked.com/a/531cb9bbd24b1dcb

Explaining Code

Explaining Code (1)

A

Can you please explain and summarize the following code:

```
class AlgorithmRegistrar {
  class AlgorithmFactoryPair {
    string name_;
    AlgorithmFactory algorithmFactory_;
  public:
    AlgorithmFactoryPair(const string& name,
        AlgorithmFactory algorithmFactory)
    : name_(name),
      algorithmFactory_(std::move(algorithmFactory)) {}
    const std::string& name() const { return name_; }
    unique_ptr<AbstractAlgorithm> create() const {
      return algorithmFactory_();
    }
    std::vector<AlgorithmFactoryPair> algorithms;
    static AlgorithmRegistrar registrar;

// cont =>
```

```
// cont =>

public:
  static AlgorithmRegistrar& getAlgorithmRegistrar();
  void registerAlgorithm(const std::string& name,
               AlgorithmFactory factory) {
    algorithms.emplace_back(name, std::move(factory));
  }
  auto begin() const {
    return algorithms.begin();
  }
  auto end() const {
    return algorithms.end();
  }
  std::size_t count() const { return algorithms.size(); }
};
```

Explaining Code (1)

The code defines a class named `AlgorithmRegistrar`. The purpose of this class is to keep track of a collection of algorithms and to provide a way to register new algorithms.

The `AlgorithmRegistrar` class contains a private inner class named `AlgorithmFactoryPair`, which represents a pair of an algorithm's name and its factory method. The `name_` member variable stores the name of the algorithm, while the `algorithmFactory_` member variable stores a function pointer to a factory method that returns a unique pointer to an instance of the `AbstractAlgorithm` class.

wrong

⇒

Explaining Code (1) - Cont’

wrong

The `AlgorithmRegistrar` class contains a public `std::vector` of `AlgorithmFactoryPair` objects named `algorithms`, which is used to store registered algorithms. It also has a static member variable named `registrar` of type `AlgorithmRegistrar`, which is used to ensure that there is only one instance of the `AlgorithmRegistrar` class throughout the program.

The `AlgorithmRegistrar` class provides three public member functions. The first is a static function named `getAlgorithmRegistrar()`, which returns a reference to the `AlgorithmRegistrar` object. The second is a function named `registerAlgorithm`, which takes the name of an algorithm and its factory method as parameters, and adds them to the algorithms vector.

⇒

Explaining Code (1) - Cont’

Finally, the class provides two member functions, `begin()` and `end()`, that return iterators to the beginning and end of the algorithms vector, respectively.

The `AlgorithmRegistrar` class also provides a `count()` function that returns the number of registered algorithms.

Overall, this class is a useful utility for registering and keeping track of algorithms in a program, allowing for easy retrieval and iteration over the collection of algorithms.

Explaining Code (2)

In respect to the code above, can you explain the following code:

```
struct AlgorithmRegistration {
  AlgorithmRegistration(const std::string& name, AlgorithmFactory algorithmFactory)
{
    AlgorithmRegistrar::getAlgorithmRegistrar()
          .registerAlgorithm(name, std::move(algorithmFactory));
    }
};

#define REGISTER_ALGORITHM(ALGO) AlgorithmRegistration \
   _##ALGO(#ALGO, AlgorithmFactory([]{return std::make_unique<ALGO>();}))
```

Explaining Code (2)

The code defines a struct named `AlgorithmRegistration` that is used to register new algorithms with the `AlgorithmRegistrar` class.

The constructor of `AlgorithmRegistration` takes two parameters: the name of the algorithm, represented as a `std::string`, and a function pointer to a factory method that returns a unique pointer to an instance of the algorithm.

The `AlgorithmRegistration` constructor then calls the `registerAlgorithm` method of the `AlgorithmRegistrar` class to register the algorithm with the provided name and factory method.

⇒

Explaining Code (2) - Cont’

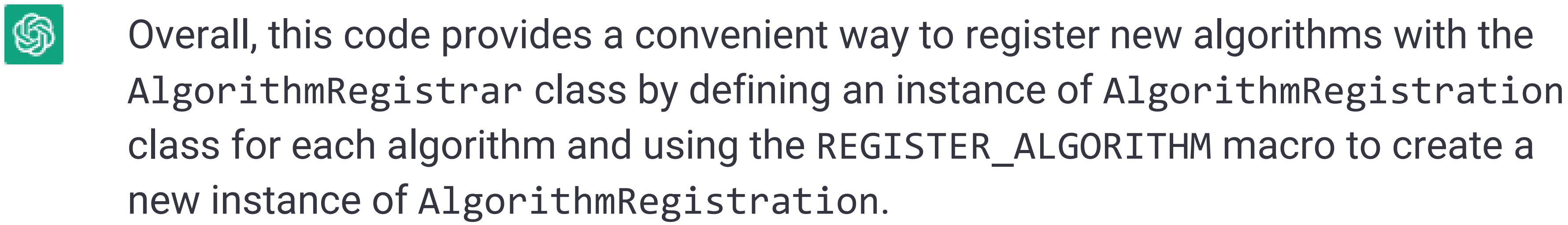
The code also defines a macro named `REGISTER_ALGORITHM`, which is used to simplify the process of registering new algorithms.

The `REGISTER_ALGORITHM` macro takes the name of the algorithm as its argument and creates an instance of `AlgorithmRegistration` with the name and a lambda function that creates a new instance of the algorithm using the `std::make_unique` function.

The macro uses the `_` prefix to create an instance of `AlgorithmRegistration` with a unique name that includes the name of the algorithm being registered. This ensures that the instance is created and registered with the `AlgorithmRegistrar` class at compile-time.

⇒

Explaining Code (2) - Cont’



Overall, this code provides a convenient way to register new algorithms with the `AlgorithmRegistrar` class by defining an instance of `AlgorithmRegistration` class for each algorithm and using the `REGISTER_ALGORITHM` macro to create a new instance of `AlgorithmRegistration`.

Explaining Code - Summary

The provided explanation for a short, yet quite complicated piece of code, was pretty good. There were two minor mistakes that could be easily spotted and fixed.

Fixing Code

Fixing Code (1)

What is the problem with the code below, I am getting compilation errors when trying to use the return value of the function `min`.

```
template<typename... Ts>
struct common_return {
    using type = std::common_reference<Ts...>;
};

template<typename T, typename... Ts> requires std::is_lvalue_reference_v<T>
                                                                     &&
(std::is_lvalue_reference_v<Ts> && ...)
                                                                     && (
std::same_as<T, Ts> && ... )
struct common_return<T, Ts...> {
    using type = std::common_reference_t<T, Ts...>&;
};

// additional 15 lines of code, including the min function (see code here)
```

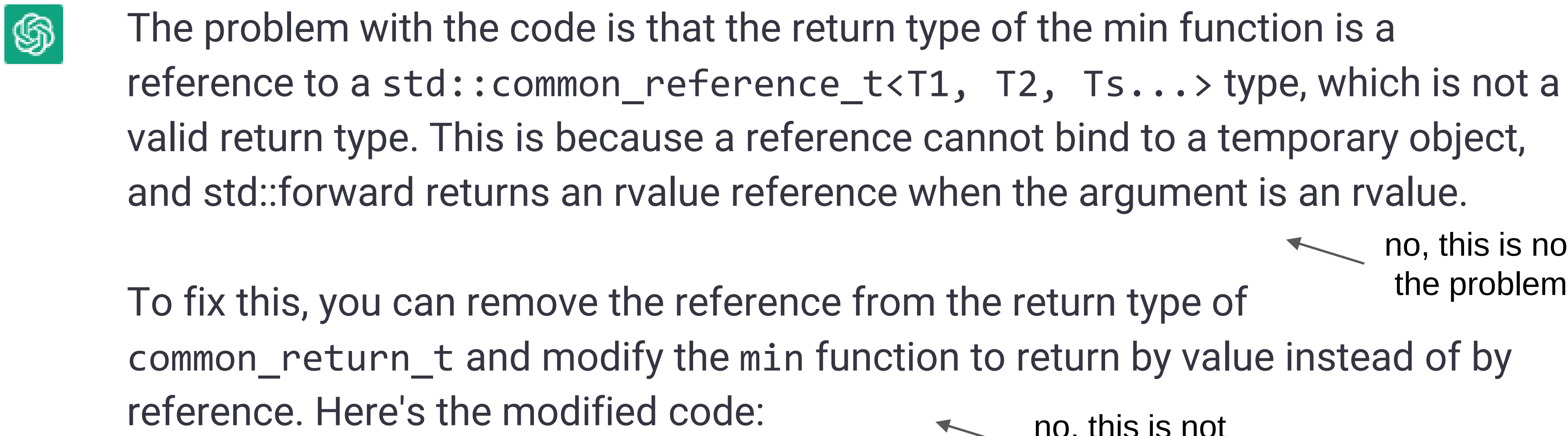
Fixing Code (1)

What is the problem with the code below, I am getting compilation errors when trying to use the return value of the function `min`.

problem is at this line

```
template<typename... Ts>
struct common_return {
    using type = std::common_reference<Ts...>;
};

template<typename T, typename... Ts> requires std::is_lvalue_reference_v<T>
                                                                      &&
(std::is_lvalue_reference_v<Ts> && ...)
                                                                      && (
std::same_as<T, Ts> && ... )
struct common_return<T, Ts...> {
    using type = std::common_reference_t<T, Ts...>&;
};

// additional 15 lines of code, including the min function (see code here)
```

Fixing Code (1)



The problem with the code is that the return type of the min function is a reference to a `std::common_reference_t<T1, T2, Ts...>` type, which is not a valid return type. This is because a reference cannot bind to a temporary object, and std::forward returns an rvalue reference when the argument is an rvalue.

To fix this, you can remove the reference from the return type of `common_return_t` and modify the `min` function to return by value instead of by reference. Here's the modified code:

no, this is not the problem

no, this is not the fix

... Code ...

code that actually fixes the problem!!! (by adding two characters)

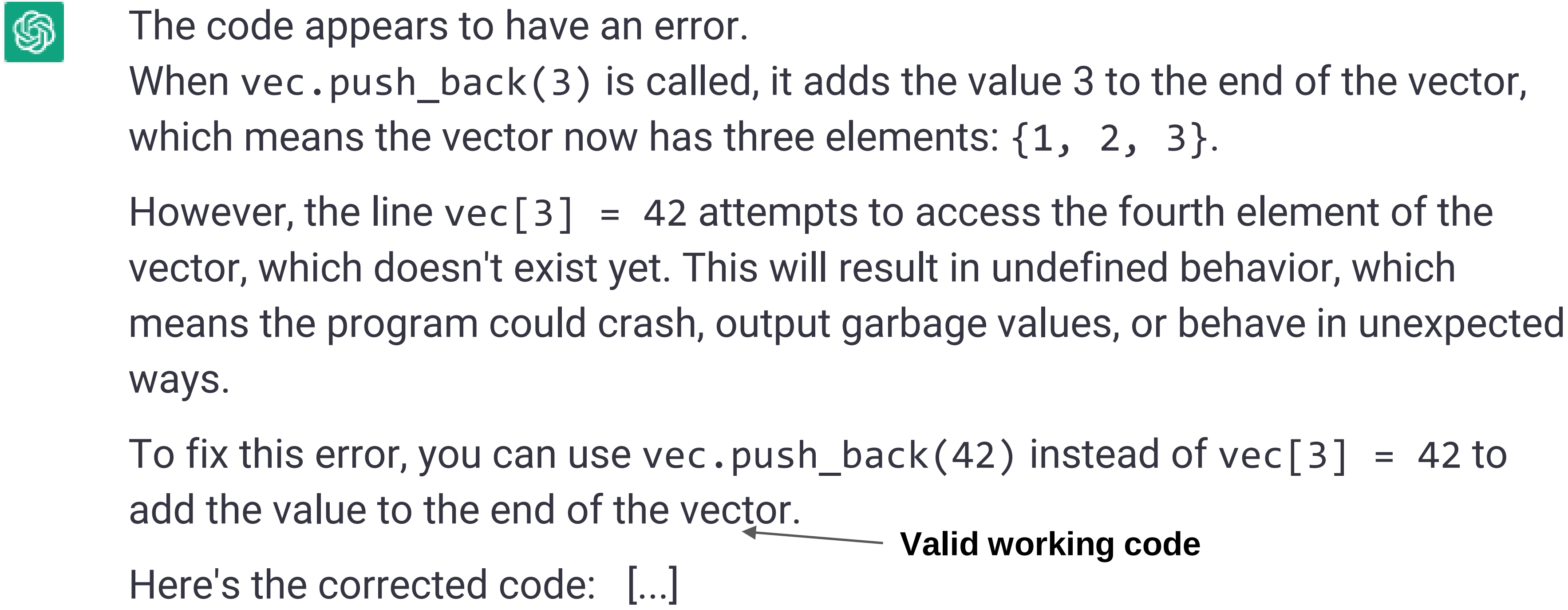
Title given by GPT to this question:
Fixing common_return_t Type

Fixing Code (2)

Can you check if the following code is valid:

```
int main() {
    std::vector<int> vec = {1, 2};
    vec.push_back(3);
    vec[3] = 42;
    std::cout << vec[3] << '\n';
    std::cout << vec.capacity() << '\n';
}
```

Fixing Code (2)



The code appears to have an error.
When `vec.push_back(3)` is called, it adds the value 3 to the end of the vector, which means the vector now has three elements: `{1, 2, 3}`.

However, the line `vec[3] = 42` attempts to access the fourth element of the vector, which doesn't exist yet. This will result in undefined behavior, which means the program could crash, output garbage values, or behave in unexpected ways.

To fix this error, you can use `vec.push_back(42)` instead of `vec[3] = 42` to add the value to the end of the vector.

Here's the corrected code: [...]

Valid working code

Fixing Code - Summary

Case #1:

The code was perfectly fixed!

The explanations for the fix were wrong.

Case #2:

The code was perfectly fixed!

The explanation was perfect!

Fixing Code - Self Healing CI Automation

Consider the following use case:

1. **CI automation failed (compilation / linking / testing)**

   Reason may be: bad merge, new bug, old bug etc.

1. **AI Model proposes a fix and the CI pipeline runs again automatically**

   The model seeks recovery using public knowledge, as well as private info such as: JIRA bugs, previous branches, merge decisions.

1. **There may be more than one possible fix that amend the failure!**

   The model presents all recovery options for the developer to make a decision.

More than one iteration per CI run, without human intervention

Iterations must be fast!

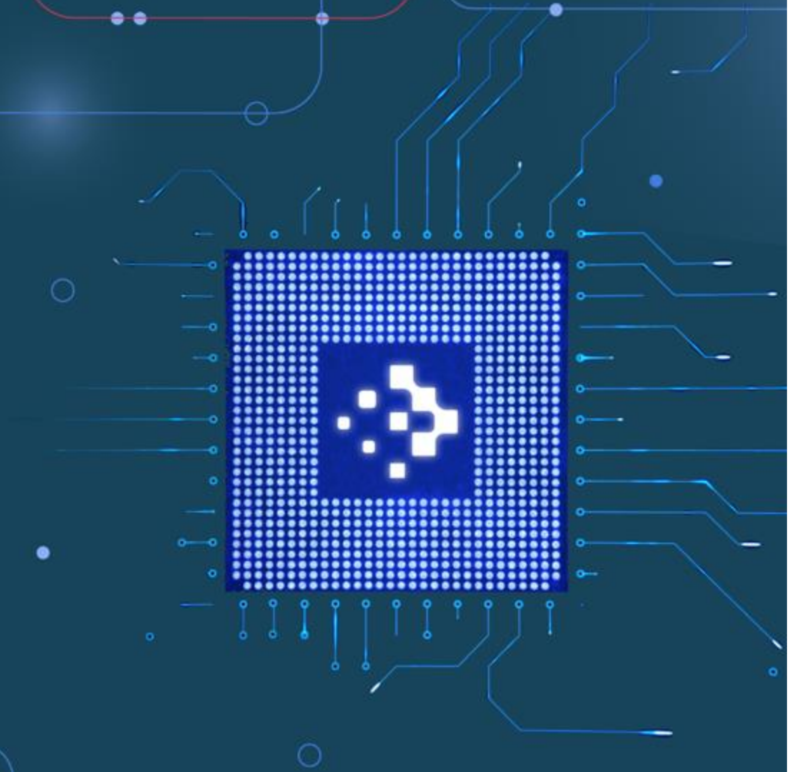
INCREDIBUILD

Suffering from slow builds?

It’s not just waste of time

It affects your dev cycles
and productivity

SPEEDATA



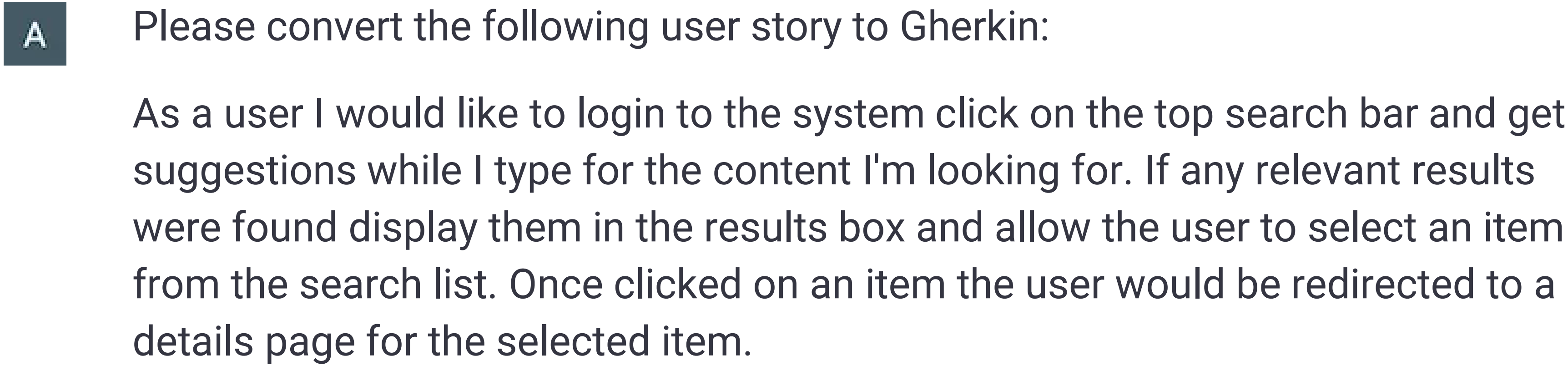
Fixing Code - Self Healing - we’re “*almost* there”

Self healing coding project in python:

https://github.com/biobootloader/wolverine

Testing

Testing - BDD (1)

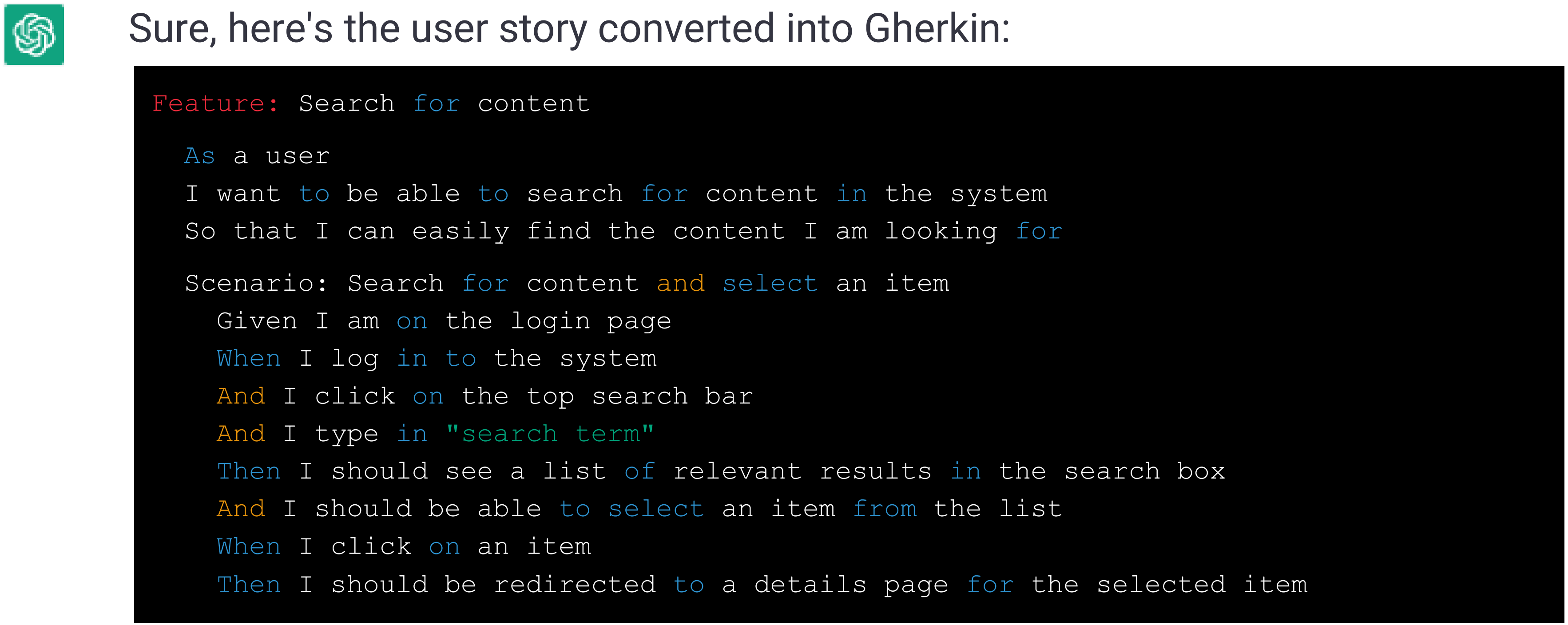


Please convert the following user story to Gherkin:

As a user I would like to login to the system click on the top search bar and get suggestions while I type for the content I'm looking for. If any relevant results were found display them in the results box and allow the user to select an item from the search list. Once clicked on an item the user would be redirected to a details page for the selected item.

* Credit for this example goes to [Ran Rosenman]

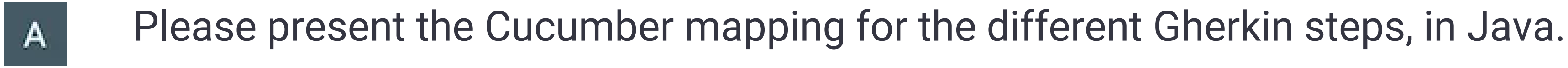
Testing - BDD (1)



Sure, here's the user story converted into Gherkin:

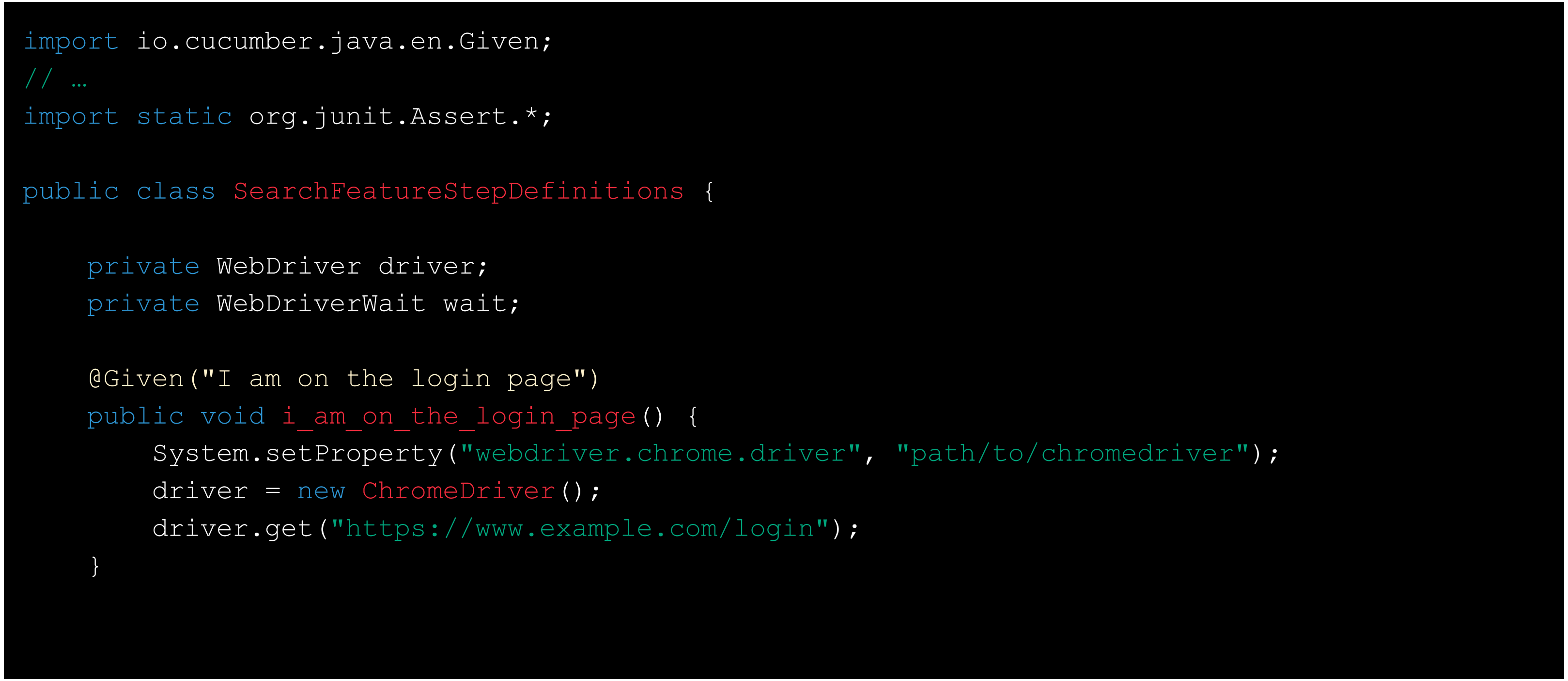
```
Feature: Search for content

  As a user
  I want to be able to search for content in the system
  So that I can easily find the content I am looking for

  Scenario: Search for content and select an item
    Given I am on the login page
    When I log in to the system
    And I click on the top search bar
    And I type in "search term"
    Then I should see a list of relevant results in the search box
    And I should be able to select an item from the list
    When I click on an item
    Then I should be redirected to a details page for the selected item
```

Testing - BDD (2)



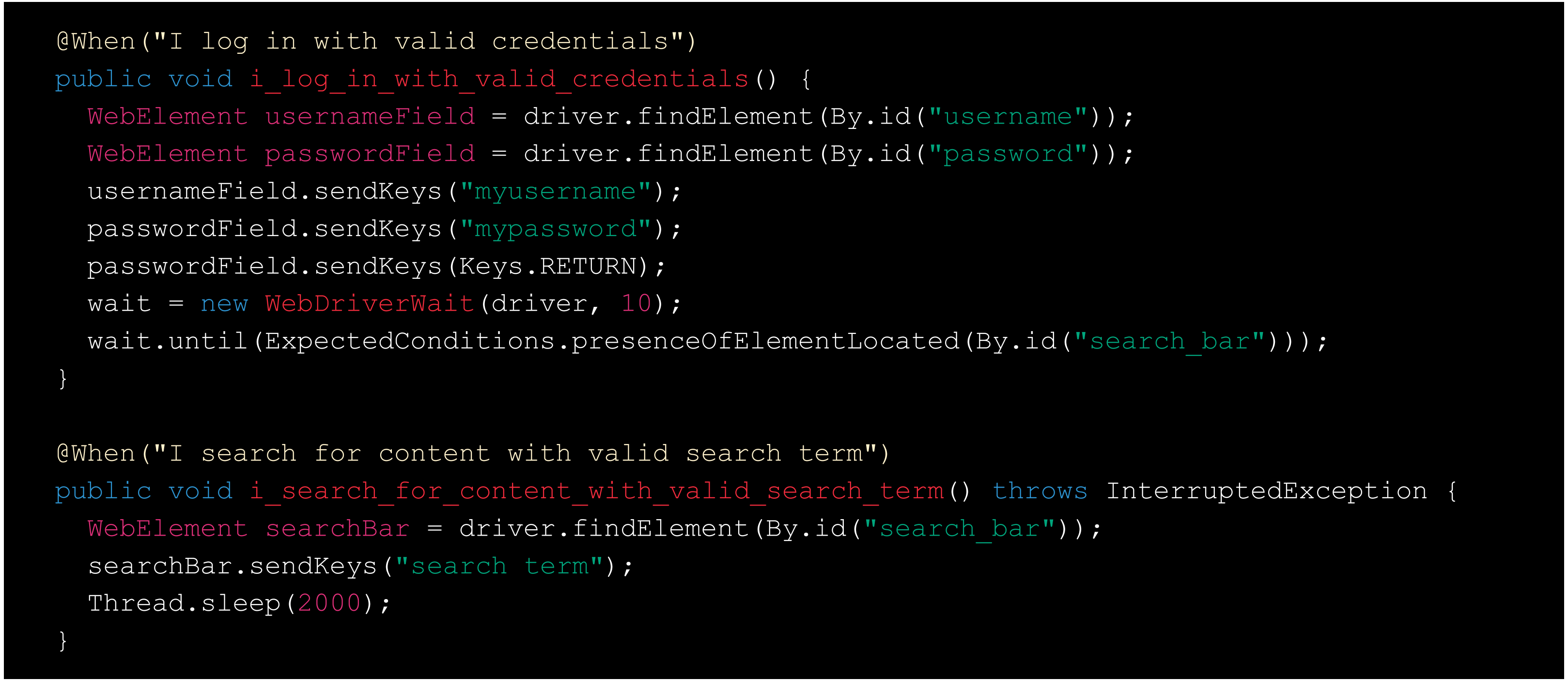
Please present the Cucumber mapping for the different Gherkin steps, in Java.

Testing - BDD (2a)



```
import io.cucumber.java.en.Given;
// …
import static org.junit.Assert.*;

public class SearchFeatureStepDefinitions {

    private WebDriver driver;
    private WebDriverWait wait;

    @Given("I am on the login page")
    public void i_am_on_the_login_page() {
        System.setProperty("webdriver.chrome.driver", "path/to/chromedriver");
        driver = new ChromeDriver();
        driver.get("https://www.example.com/login");
    }
```

Testing - BDD (2a) - Cont’



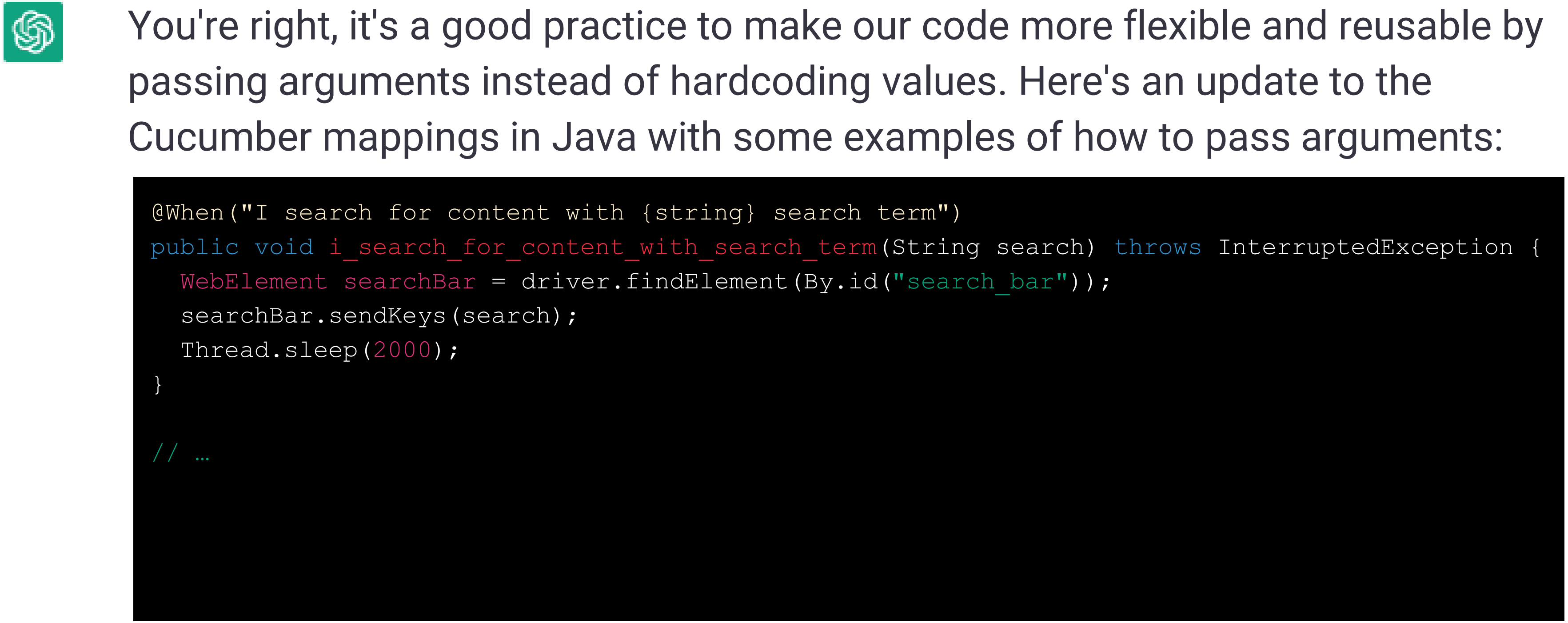
```
@When("I log in with valid credentials")
public void i_log_in_with_valid_credentials() {
  WebElement usernameField = driver.findElement(By.id("username"));
  WebElement passwordField = driver.findElement(By.id("password"));
  usernameField.sendKeys("myusername");
  passwordField.sendKeys("mypassword");
  passwordField.sendKeys(Keys.RETURN);
  wait = new WebDriverWait(driver, 10);
  wait.until(ExpectedConditions.presenceOfElementLocated(By.id("search_bar")));
}

@When("I search for content with valid search term")
public void i_search_for_content_with_valid_search_term() throws InterruptedException {
  WebElement searchBar = driver.findElement(By.id("search_bar"));
  searchBar.sendKeys("search term");
  Thread.sleep(2000);
}
```

Testing - BDD (3)

A

Some of the function can use arguments (from the Gherkin test scenario) instead of hard coded values.

Testing - BDD (3)



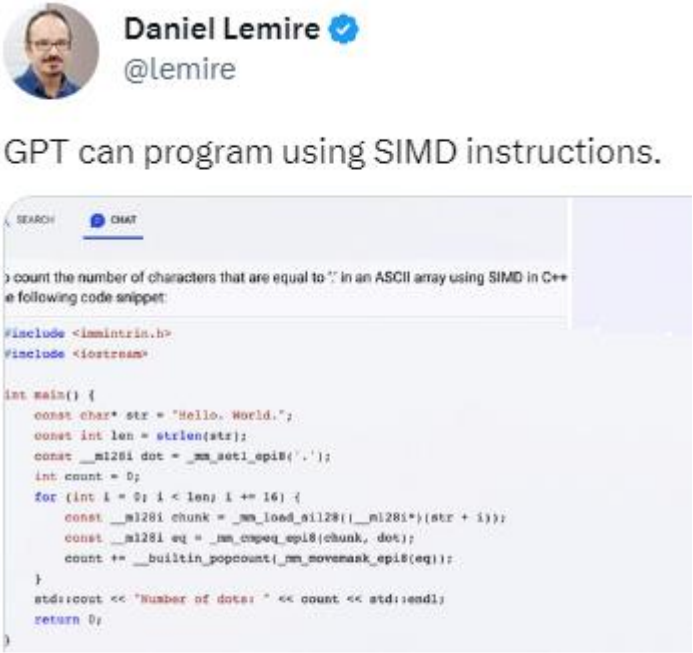
You're right, it's a good practice to make our code more flexible and reusable by passing arguments instead of hardcoding values. Here's an update to the Cucumber mappings in Java with some examples of how to pass arguments:

```
@When("I search for content with {string} search term")
public void i_search_for_content_with_search_term(String search) throws InterruptedException {
  WebElement searchBar = driver.findElement(By.id("search_bar"));
  searchBar.sendKeys(search);
  Thread.sleep(2000);
}

// …
```

Testing - Summary

Generating simple (yet meaningful) unit tests, seems as a reasonable and immediate use case for AI models.

But it creates bugs! Well, also humans

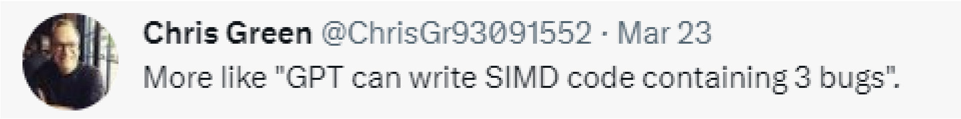


Daniel Lemire
@lemire

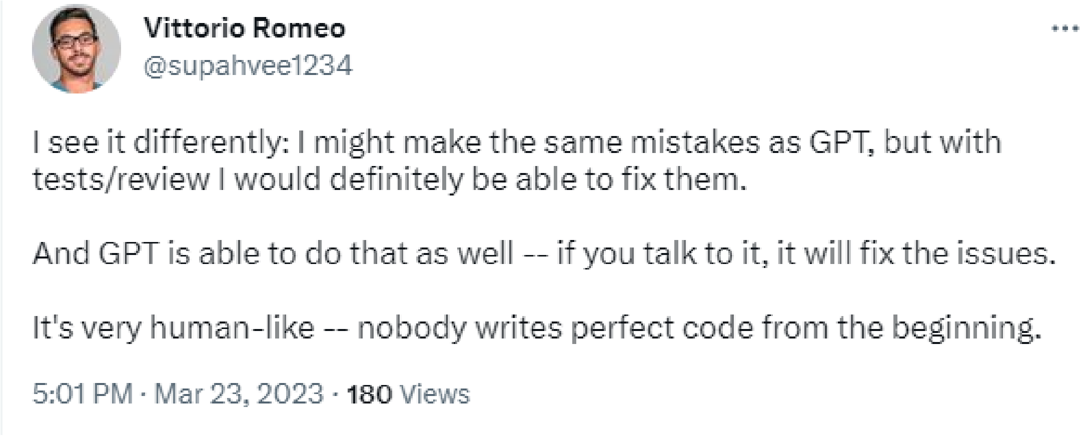
GPT can program using SIMD instructions.

count the number of characters that are equal to '.' in an ASCII array using SIMD in C++
e following code snippet:

```
#include <immintrin.h>
#include <iostream>

int main() {
    const char* str = "Hello, World.";
    const int len = strlen(str);
    const __m128i dot = _mm_set1_epi8('.');
    int count = 0;
    for (int i = 0; i < len; i += 16) {
        const __m128i chunk = _mm_load_si128((__m128i*)(str + i));
        const __m128i eq = _mm_cmpeq_epi8(chunk, dot);
        count += __builtin_popcount(_mm_movemask_epi8(eq));
    }
    std::cout << "Number of dots: " << count << std::endl;
    return 0;
}
```



Chris Green @ChrisGr93091552 · Mar 23
More like "GPT can write SIMD code containing 3 bugs".



Vittorio Romeo
@supahvee1234

I see it differently: I might make the same mistakes as GPT, but with tests/review I would definitely be able to fix them.

And GPT is able to do that as well -- if you talk to it, it will fix the issues.

It's very human-like -- nobody writes perfect code from the beginning.

5:01 PM · Mar 23, 2023 · 180 Views

https://twitter.com/supahvee1234/status/1638918781081624576

Bigger Code Challenges

Above examples were on small code snippets.

How good are the models with a much bigger code base?
Would they be able to introduce a new feature into existing code?

- Models may have inherent advantage in reading big chunks of code
- Decoupling, decent unit testing - may become even more required
- Code challenges will become iterative, in bites that the model can chew

Testimonials from a C++ User Group

I had an issue with forwarding variadic template elements into a tuple.
I'm not considering myself an expert in templates so I tried to google without much help.

Then I asked GPT and got back a pretty good example of almost what I needed. With a few modifications I had the code that I needed.

I asked GPT to review simple code that had to run in a multi threaded environment.

The response pointed at a certain variable that may need to be atomic, which was a good comment. It also pointed at a risky division that may become a division by zero in some case, which I didn't think about.

End of Ch.4 - Coding and Testing

Open Discussion

Ch.5 - Is C++ Special?

HEY, HE KNOWS C++
VERY WELL AND CAN HELP YOU
YOU SEE? NOBODY CARES
imgflip.com

Special in which sense?

Maybe it’s easier to generate valid Python code with AI models compared to C++?

If C++ is too complicated for AI to master:

- Maybe C++ programmers (specifically) cannot be replaced by AI so soon?
  (**So, nothing to worry about, yet**)
- Or maybe this is a disadvantage for C++ that will make it much less useful?
  (**So, start worrying!**)

C++ problem domains are usually harder

That’s a challenge for the models.

But, Inside any complicated domain there is a less-complicated part striving to emerge =>

- The less-complicated parts may use AI models.
- Decoupling will become yet even more important

C++ is more complicated

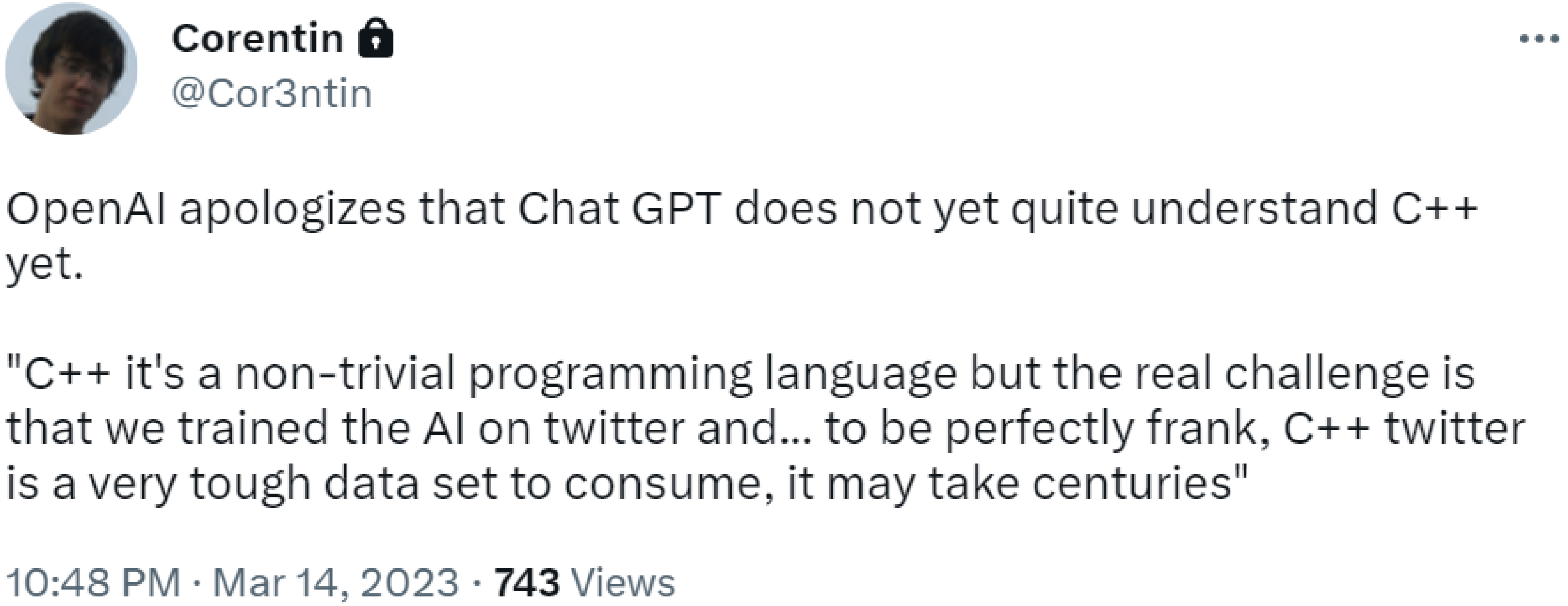
Yes, let’s admit,

- It has more complicated rules compared to most languages
- A lot of old legacy code that may confuse the AI

Can we do something with that?

- Guide the AI tool which C++ version to use
- Run static code analysis on the generated code
- Always review the code, feed the model back with comments if there is a need for improvement (relevant for generated code in any language)

C++ is more complicated



Corentin

@Cor3ntin

OpenAI apologizes that Chat GPT does not yet quite understand C++ yet.

"C++ it's a non-trivial programming language but the real challenge is that we trained the AI on twitter and... to be perfectly frank, C++ twitter is a very tough data set to consume, it may take centuries"

10:48 PM · Mar 14, 2023 · **743** Views

https://twitter.com/Cor3ntin/status/1635744803165028353

Alternatives for C++

1. Generate **Assembly** directly? (Probably a bad idea:)
    a. Much less public assembly code to learn from? Well, a model can still learn from assembly generated from public C++ code
    b. Humans still need to review, understand, modify and maintain the generated code
2. Generate **Python** (easier?). Then may use a model to translate it to C++?
    a. On the other hand, a compiled language is a better target, as the compiler serves as an additional gate
    b. If the final target is C++, do we get any benefit for using any intermediate language?
3. Generate **Rust** (more secure?)
    a. More errors can be caught during compilation
    b. On the other hand, much less Rust code for the model to learn from

End of Ch.5 - Is C++ Special?

Open Discussion

Ch.6 - Getting Prepared

How can we prepare ourselves to the AI era?

Let’s assume that we are a *journalist*.

What are the implications and how can we be prepared to the AI era?

(Can we assume it’s not going to affect our domain, if we are journalist?)

How can we prepare ourselves to the AI era?

What about coding?

What are the implications and how can we be prepared to the AI era?

- On recruiting
- On teaching software programming
- On our daily work

And also,

- Would AI models be invited to give talks at C++ now

Things change… – Recruiting

Do we need junior developers?
(If not, how developers would become senior?)

What are the skills that we should look for? Do they change?

Should we change our job interviews?

What do you say?

Things change… – Teaching

Should we allow students to use of AI models when working on their coding assignments? (Can we actually prevent them from using it?)

Should it change the level of assignments?

Should it change the strictness of checking assignments?

Should it change the curriculum and teaching goals?

What do you think?

Things change… – Coding Practices

How would it change our working habits?

What should be the security measures for integrating generated code?
(Should they be different than integrating code pasted from stackoverflow?)

Can we count on one AI model to perform code review on code generated by another AI model?

What are your thoughts on that?

Implications to other domains

Processors

Compute power is an obstacle!

A significant drive for new processor technologies.

- Analog Processors?
- Quantum computing?
- Biological computing?

All three suffer from too noisy output (unacceptable error rates)

The last two are far from being ready.

Analog Processing is here, may be used for part of the modeling process?

Technical Support

You all probably “love” the customer support robots that try to answer your questions before reaching a real human representative, right?

So you will probably have many more of those, for all sorts of technical support

Safety and Security

How can you know that the AI model didn't plant an intentional bug?

Humans can also plant intentional bugs, but they are usually afraid of the consequences… (and less manipulative?)

AI Models to Recognize AI Outputs

Schools, Academia, Work (?)…

There will be a need to distinguish between human and machine generated outputs

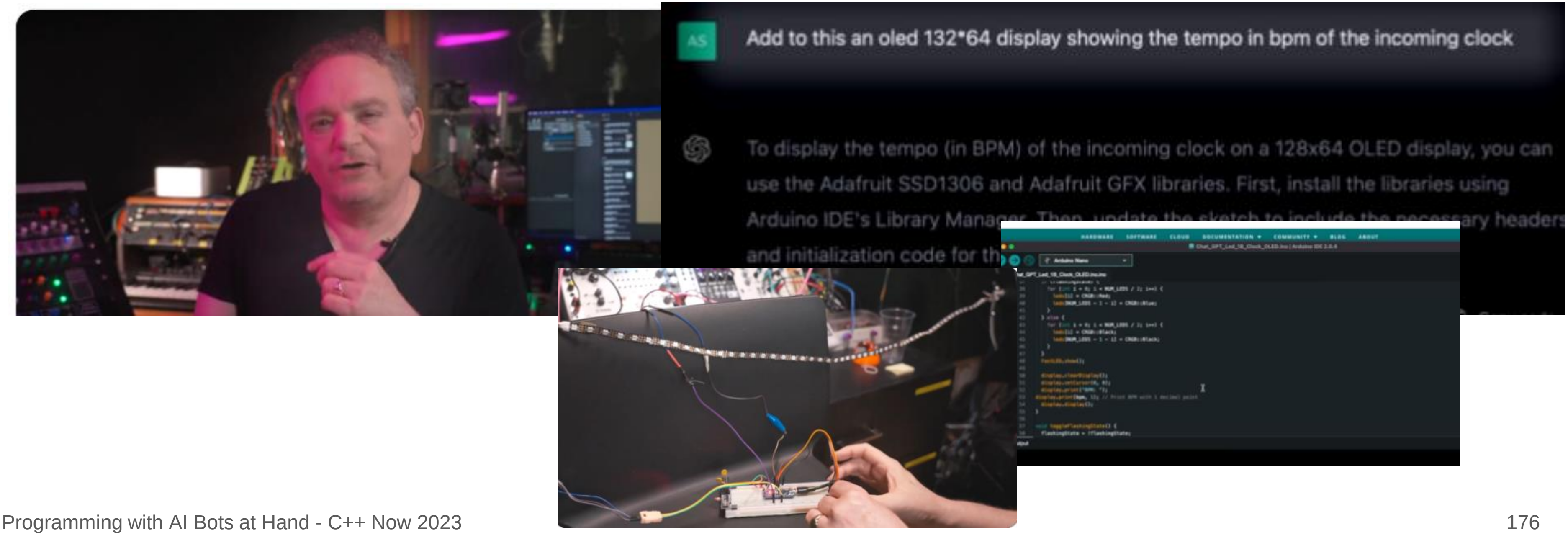
- For teaching assignments
- For estimating quality of work
- Avoiding paying high tariffs for generated outputs

And… any domain that can use coding

Izhar Ashdot
@izhar_ashdot

https://twitter.com/izhar_ashdot/status/1645737814972157954

^ In hebrew, but you can understand the context



Add to this an oled 132*64 display showing the tempo in bpm of the incoming clock

To display the tempo (in BPM) of the incoming clock on a 128x64 OLED display, you can use the Adafruit SSD1306 and Adafruit GFX libraries. First, install the libraries using Arduino IDE's Library Manager. Then, update the sketch to include the necessary headers and initialization code for th

End of Ch.6 - Getting Prepared

Open Discussion

- **Do you think there is *too much fuss* and AI would not significantly affect SW programming in the coming 5 years?**

End of Ch.6 - Getting Prepared

Open Discussion

- **Do you have in mind *other effects*, not yet discussed, that AI may have on the software development industry?**

End of Ch.6 - Getting Prepared

Open Discussion

- **Other thoughts** anyone wants to share?

Ch.7 - Conclusions and recommendations

Some people think LLM are a game changer

- ➔Don’t ignore it, even if you think otherwise
- ➔Think how it is going to affect your environment, then adjust
- ➔Find ways to utilize LLM in your business

Avoid bad proposals

It is suggested to add to your prompts:

> **You don't have to please me.**
>
> **If it is not possible, answer that and explain why.**

Remember

- ➔Though current results may not always be perfect
    - You can control that with better prompts
    - Plugins may help in automatically improving outputs
    - The models are improving

- ➔Never rely on anything without reviewing it. Trust nothing

- ➔The skill of reading code becomes even more important than ever

How LLM can become better in coding?

1. Try to **compile and run** the code to be proposed, before proposing it. Use errors as an additional input for generating the final proposed code.
2. Feed the model **high quality data** (e.g. accepted stackoverflow answers or those with high reputation) with **higher weight** (e.g. by augmenting these higher quality data inputs and feeding them into the model more times).
3. Propagate the **confidence level** out of the model, so the user will get a notion of how much the model feels comfortable about the answer.
4. Train the model on **bad answers** so it can learn to avoid them. For example, with submitted leetcode answers the model can use *failing answers* as an input for learning bad code examples.

Any comments or questions before we conclude

Credit note

We wish to thank Gerald Monroe, Qualcomm ML expert, for discussing together some of the ideas presented in this talk.

c

Additional Resources

C++ Weekly - Ep 371 - Best Practices for Using AI Code Generators (ChatGPT and GitHub Copilot) - https://youtu.be/I2c969I-KmM

C++ Weekly - SE - Make Your Own Compiler-Explorer With GPT-4 in 5 Minutes! - https://www.youtube.com/watch?v=XD3b2HA_7BQ

Northwest C++ User Group - Open Discussion: Chatting on ChatGPT - https://www.youtube.com/watch?v=mtpU19Zimps

Three Secret Keys to LLM Alignment With Minimal Hallucinations: https://medium.com/@nissan_yaron/obedience-is-what-you-ask-but-not-what-they-need-beaa5dcd8452

LangChain Project - https://python.langchain.com

Interesting online papers:

GPT4 Technical Report - https://cdn.openai.com/papers/gpt-4.pdf

Survey of Hallucination in Natural Language Generation - https://arxiv.org/abs/2202.03629

Teaching Models to Express Their Uncertainty in Words - https://arxiv.org/abs/2205.14334

Augmented Language Models: a Survey - https://arxiv.org/abs/2302.07842

Toolformer: Language Models Can Teach Themselves to Use Tools - https://arxiv.org/abs/2302.04761

An Early Look at the Labor Market Impact Potential of Large Language Models - https://arxiv.org/abs/2303.10130

Open AI Research Page - https://openai.com/research

Final thing before we depart…

Core C++ 2023: https://corecpp.org

Tel-Aviv, 5th to 7th of June

Program is published, registration still open!

Thank you!

Amir Kirsh

kirshamir@gmail.com

amir.kirsh@incredibuild.com

Alex Dathskovsky

calebxyz@gmail.com

alex.dathskovsky@speedata.io