Meno II: A self-referential Socratic dialogue about memory and computer programming

Anonymous

March 2025

Abstract

We provide a Socratic dialogue in which Meno challenges Socrates' principle that all learning is actually the remembering of things known but forgotten. Meno claims if Socrates' principle holds, then Meno should already know the contents of the ongoing dialogue. Meno challenges Socrates to help Meno recall those contents, claiming Socrates cannot do this, lest Meno could immediately say the opposite of whatever he supposedly recalls he was about to say. Socrates eventually succeeds at Meno's challenge in an unexpected manner.

1 Introduction

In the Meno, Plato's Socrates suggests the principle of anamnesis: there is no learning, only the remembering of forgotten knowledge. Presumably this should not be understood of contingent facts, or rather, perhaps we never truly know contingent facts: thus, Wittgenstein refused to acknowledge that there is no rhinoceros in the room. But it is fun to speculate about what would be implied if anamnesis did extend to contingent facts. It would extend to Plato's speakers' knowledge of their own words: they would know (but have forgotten) the very words they say, before saying them.

We designed a Socratic dialogue in which Meno asks for assistance remembering what he will say in the dialogue. Against all odds, Socrates succeeds. The dialogue takes place in a universe where Socrates and Meno know computer programming; Socrates uses techniques similar to quines (self-printing computer programs) and obfuscated code to evade Meno's paradox. Because of the venue's wordcount limit, we give here key excerpts (roughly 2500 words) of the roughly 7500 word dialogue. The full dialogue is available at `https://github.com/DoubleBlind87654/MenoDoubleBlind`. In the climax, Socrates leads Meno to decode a computer program steganographically hidden in the words of the dialogue; skeptical readers can verify this computation using the full dialogue at the above link (either from scratch, or with the assistance of a verification script available at the same link). After these excerpts, we remark about how the dialogue was engineered.

2 Excerpts of the Dialogue

Meno: Good morning, Socrates.
Socrates: Good morning to you, Meno, my friend.
Meno: I've been wanting to talk to you.
Socrates: And just what would you like to talk about, my dear friend?
Meno: It's about what you said the last time you and I spoke together.
Socrates: I suppose you mean when I told you that learning is really just remembering?
Meno: Yes, and that we actually know everything already, and we merely remember it.
Socrates: And I suppose you've contrived some sort of paradox or something to try and disprove it?
Meno: Quite. Once we finish this delightful discussion we're having, don't you think that then I will know all the things we said?
Socrates: You mean, know the conversation?
Meno: Exactly. Once we're done, I'll then know everything we said. That is, I will have learned it, wouldn't you say?
Socrates: I'm not really so sure about that. More often than not, I find I misremember conversations, or, even worse, I think that I remember and understand them, and then it later happens that I either remember wrong, or understand wrong. Besides, your question makes me uneasy, for I feel as if I am in danger of falling into a trap somehow.
Meno: What! Does the great Socrates have trouble with his memory?
Socrates: I often wish that I could remember things better, Meno. For I feel sometimes as though I flitter between two different worlds. One is intelligible and beautiful, the other is sensuous and profane. For instance, when my house is well organized, and I remember where I have put everything, it's as if I live in a house designed by the geometers, for I can find my shirt or my shoes by an act of pure thinking. But more often, it pains me to say, my house is not organized, and I must grope around for my shirt or my shoes, and I feel as if I'm cast out of that house of the geometers, and plunged into a carnal house of the senses.
Meno: Well, as for me, I take great pride in my house always being organized. For I employ servants who keep it organized for me. And, look here: I have a servant writing this very conversation of ours on a stone tablet. You know him: it is the same boy with whom you conversed about geometry the last time you visited. I treasure your words like gold, Socrates, and I will preserve them, and guard them, just like I would treat a valuable treasure.
Socrates: You flatter me, Meno. But I suspect you're just saying that because you want to catch me in some trap. Very well, Meno. For the sake of the argument, I grant that after we have this conversation, you shall have learned it.
Meno: Well then, doesn't that mean I already knew it, and have only to remember?

Socrates: Yes, I grant that, Meno. Even before you were born, you already knew I would say the words, 'Yes, I grant that, Meno,' which I said just now.
Meno: Well then, dear Socrates, I have a challenge for you. With your famous method of questioning, please guide me to recall all of the things we shall say in this dialogue we're having. That is, if you are able to. But I doubt it.
Socrates: Why do you doubt I can do it, my dear Meno?
Meno: Because the very moment you help me remember the things we're going to say to one another, I will immediately remember what line I will say next, and I will say the very opposite.
(1672 characters later...)
Socrates: Very well. This is a fun challenge. And I think if I'm to have any hope at succeeding, I will need to resort to computer programming.
Meno: Perhaps. For my challenge does seem somehow related to things like the recursion theorem and the incompleteness theorem. But even computer programming won't rescue you from my paradox.
(1399 characters later...)
Socrates: So for example, suppose I ask you: take the first sentence you said to me, and repeat it to me again right now, and then after that, please repeat it to me a second time. And suppose I claimed that, by doing so, you would recite our entire conversation. Would I be right?
(667 characters later...)
Socrates: So the reason why this supposed one sided conversation would not win your challenge, is because of the content of the results?
Meno: Quite so.
Socrates: And is there any other reason why this example conversation would not beat your challenge?
Meno: No.
Socrates: In that one sided example, I used earlier pieces of our conversation as tools. But you say, the lone reason that example would fail to solve the challenge, is due to the contents of the resulting one sided conversation?
Meno: That's right.
Socrates: Then it's perfectly alright for me to use earlier pieces of our conversation like that?
Meno: Yes, that's alright.
(2234 characters later...)
Socrates: But isn't your servant writing some other things here? Look. He writes 'Meno:' whenever you begin talking. He writes 'Socrates:' whenever I do.
Meno: Sure.
Socrates: We have a serious problem, then. Look at the tablet right here, near the very top. It says: 'Good morning to you, Meno, my friend. Meno:' Now, I can't remember. Did I say 'Meno:' immediately after 'Good morning to you, Meno, my friend'? Or rather, does 'Meno:' here mean that you began talking?
(1196 characters later...)

Socrates: But maybe we can modify the challenge. Do you at least know what is written on the tablet? For instance, that the very first letter on the tablet is 'M'?
Meno: Indeed.
Socrates: Then maybe instead of helping you to recall our future conversation, I could instead help you to recall the future contents of the tablet. Your little paradox would still confound me, wouldn't it?
Meno: It would.
Socrates: And I wouldn't even necessarily need to help you remember my own contributions to the tablet, would I? Wouldn't you say I won if I could just help you recall your own future contributions to the tablet?
(2166 characters later...)
Socrates: Do you remember how Ulysses tricked Polyphemus, the cyclops? He lied to Polyphemus, saying his name was "Noman". Then when Polyphemus's neighbors came and asked if some man was slaying him, Polyphemus said, 'Noman is slaying me.' Didn't Ulysses do that on purpose, so that the cyclops's friends would misunderstand, and think Polyphemus had said "No man is slaying me"? Isn't that just the sort of syntactical semantical confusion you have so effortlessly avoided just now?
(2922 characters later...)
Socrates: And changing people's thoughts is like writing Greek on the surface of the water. The water is so pliable, you can write on it with your bare finger. But the instant that you've written there, your writing disappears, doesn't it?
Meno: It does.
Socrates: In fact, I always assumed that's what Thales meant, when he said that everything is made out of water. He meant that all the impressions we make on the world fade, just as if we inscribed them on the face of the sea. You could not corrupt the Aegean by writing on its surface, no matter how scandalous the things you wrote there. Why, even the language we speak slowly changes. And we try to nail that down, by imposing sharp constraints on it in the form of poetry.
(6011 characters later...)
Socrates: But look here. There's some space on your servant's tablet between 'Meno:' and 'Good'. And if someone included that space in the X they passed to you, then your function would not output 'Meno:Goodm', but rather it would output 'Meno: Good', wouldn't it?
Meno: Yes.
Socrates: Can you modify your function to discard all such spaces? We'd better discard tabs and linebreaks too, and also hyphens, since I see your servant sometimes inserts a hyphen when a line ends in the middle of a word.
(3141 characters later...)
Socrates: Well, let's change our approach then. Suppose we described some elaborate transformation. And suppose you performed that transformation on this conversation so far, from the start until when I said, "Well, let us change our approach then." And suppose the output of that was:

```
print("""You win, Socrates.
You're welcome.
I admit it.""")
```

And suppose that the moment you knew that output, you said, "You win, Socrates," and I said, "Thank you, Meno," and then you said, "You're welcome," and then I said, "Do you admit there is no learning, only remembering?" and you answered, "I admit it." Would I have passed your challenge?
Meno: Certainly.
(3856 characters later...)
Socrates: Perhaps the problem is that a haystack can't be made entirely out of needles, and if our conversation does secretly hide some program that predicts you, I suppose many characters from the conversation must work together to hide each character in the program. Perhaps every five characters from our conversation could be used to generate one character of our desired program. For example, perhaps the correct cipher, rather than changing "a" into "b", changes "Meno:" into "p", and changes "Goodm" into "r", and changes "ornin" into "i", and changes "g,Soc" into "n", and changes "rates" into "t", so that, altogether, the first 25 characters of the conversation become "print". Do you think that's at all likely?
(756 characters later...)
Socrates: You mean, if programs were written with a larger alphabet, then we would need more characters of conversation per character of the hidden program?
Meno: Yes.
Socrates: Then I guess since we haven't got the whole day to stand here talking, we should agree to write this desired program with the smallest alphabet we can. We should ask ourselves: which characters are absolutely essential for writing computer programs? Within reason, that is. After all, we're not trying to compete in some obfuscated code contest.
Meno: I think a reasonable alphabet for writing programs, without too many characters, would be:

```
alphabet = "abcdefghijklmnopqrstuvwxyz"
alphabet += " ()[]:'=,+.#\n"
alphabet += '"'
```

Socrates: That's forty characters in all, right? Since I assume by \n you mean a linebreak, which is one single character. And no uppercase letters, like "A", "B", "C"?
Meno: For your sake, Socrates, I shall consider it a victory for you if our program predicts I'll say "hey", all lowercase, even if my servant here actually writes "Hey" with the H uppercase.
Socrates: Most generous of you, Meno. Well then, if we assume this rather barebones alphabet of yours is rich enough to write computer programs, how

many letters of our conversation should we use to encode each character of our program? We already saw that "one", "five", and "ten" are too small. Perhaps twenty?
Meno: Socrates, I know you haven't studied computer science, but really. Twenty is still way too small. You're probably still off by an order of magnitude, so let's make it two hundred.
Socrates: Alright, 200 it shall be. So every 200 characters of our conversation shall hide within them a single character of a computer program. Alright then, how would you define the cipher? It must take a string of 200 characters from our conversation and convert that string into a single character from your 40 letter alphabet.
Meno: Let me think, Socrates. Hmmm... Well, there's a convenient function named "ord" for turning characters into numbers. For example, ord("a") is 97, ord("b") is 98, ord("+") is 43, and so on. Don't fret about those specific numbers, this "ord" function is built into my programming language (Python). As our tablet has quotation marks and my servant sometimes writes those with fancy unicode quotes, I guess to be safe we should consider all the quotes in our conversation to be plain ASCII quotes, just like we delete all the spaces and linebreaks and tabs in the conversation. And plain ASCII quotes have ords as follows: for the single quote, ord("'")=39; for the double quote, ord('"')=34. With this ord function, we can systematically turn characters into numbers, add up those numbers, and then take the remainder after dividing the sum by the length of my program alphabet. In other words, if the sum is 40 or more, smash it down into the range from 0 to 39, so it can be used as the index of a character in our alphabet. If we write it all as a function:

```
def cipher(X):
total = sum(ord(character) for character in X)
remainder = total % len(alphabet)
return alphabet[remainder]
```

This function works for inputs X of any length. Of course it is intended to be used on inputs X of length 200.
Socrates: Just to make sure I understand right, what would be the value of, say, cipher("Meno:")?
Meno: Well, ord("M") is 77, ord("e") is 101, ord("n") is 110, ord("o") is 111, and ord(":") is 58. Add those up and you get 457. Now, the length of my alphabet is 40. If we divide 457 by 40, the result is 11 plus a remainder of 17. So the index we get is 457 % 40 = 17. So cipher("Meno:") is alphabet[17], character number 17 in our alphabet, remembering to count from zero. Which is "r". Altogether, cipher("Meno:")="r".
Socrates: Amazing, Meno, remind me someday to introduce you to my friend Timaeus. And what do we get when we take the first 200 characters from this conversation and plug them into this cipher?
Meno: If we take the first 200 characters of our conversation, they make the following string, though I fear my servant will insert linebreaks because other-

wise this long string won't fit in one single row on his tablet, so please ignore those linebreaks; all that said, those 200 characters are as follows:

```
Meno:Goodmorning,Socrates.Socrates:Goodmorningtoyo
u,Meno,myfriend.Meno:I'vebeenwantingtotalktoyou.So
crates:Andjustwhatwouldyouliketotalkabout,mydearfr
iend?Meno:It'saboutwhatyousaidthelasttimeyouandIsp
```

If we plug this length 200 string into the cipher function, remembering not to include the linebreaks, we get the output "#".
(1375 characters later...)
Socrates: By Zeus! Alright, Meno, keep going. Plug in all the length 200 blocks from our conversation, stopping with the first block in which I say "Zeus". Do we get a valid program?
Meno: One minute... No, but I am shocked, for the output does resemble a computer program, but it has all kinds of errors. Here it is, note the double space between the "n" and the "r" on line 3:

```
#trojn hrse
seed(p=askpword())
z,x=", yo ooned","""n r''s'
iati.idnet'pas'r.p(se)d nt yap'o
+twspcw'tr"""
c=jn(smpl(p,z.len
shfl(d=rng(x.len
if c==z:exec(jn(x.i:i in d
```

Socrates: You say it has errors? Can we repair it? Tell me, what's the first error you notice?
(1824 characters later...)
Socrates: What's the program now, with all the errors fixed?
Meno: Here you go:

```
#Trojan Horse
import random
p=input('Enter the password:  ')
random.seed(p)
z,x=", yo ooned","""n r''s'
iati.idnet'pas'r.p(se)d nt yap'o
+twspcw'tr"""
c="".join(random.sample(p,len(z)))
d=list(range(len(x)))
random.shuffle(d)
if c==z:exec("".join(x[i] for i in d))
```

Socrates: Run it! What does it predict? I understand you plan to say the opposite. What's wrong? You are quiet. Is there some problem with your program?
Meno: It wants a password.
Socrates: You're in a bind, for until you enter the password, you won't know what it will say once you enter it. Maybe when you enter the password, it will say, "It wants a password." It's too late to say the opposite of that. Maybe studying the code, you can figure out the next prediction, and defy the oracle? What's wrong? Can't you analyze the code to determine the next prediction?
Meno: It's encrypted.
Socrates: Then you must enter the password, and we shall see what your program predicted. Meno! Tell me! What is the password?
Meno: I don't know any password, Socrates.
Socrates: The password, in your own words, is: "I don't know any password, Socrates." Enter that, and if your wisdom is as great as I believe, I think your program will print:

```
it wants a password.
it's encrypted.
I don't know any password, Socrates.
```

Now I must leave, Meno, I have an appointment at the courthouse.

3 Conclusion

One moral we might draw from this story is that the statement "philosophy will help me remember future contingent facts even if I try to contradict them" could itself be contingent. Meno's paradoxical challenge initially seems to disprove this statement by pure reasoning (like the geometry example in Plato's *Meno*), which would make the statement universally false. Socrates' victory disproves that disproof.

We wrote the dialogue using annotations. For example, "Meno: [Yes|Correct]" means that Meno could say "Yes" or "Correct". A computer program flattened these annotations to encode Meno's program. The ways a passage can be so obtained grow exponentially with the number of annotations. "Meno: [Yes|Indeed], [Socrates|friend]" can be flattened in 4 ways; "Socrates: [Isn't that|Is that not] [right|correct], [Meno|friend]?" can be flattened in 8 ways. Thus the problem is easier than it initially appears. The annotated dialogue, and associated computer programs, are available at `https://github.com/DoubleBlind87654/MenoDoubleBlind`.

As an application, a variation of this technique could theoretically be used to assist in the problem of preserving rhyme and metre when translating poetry. We speculate perhaps similar methods were used (without computer assistance) for example by John Dryden, whose English translation of Virgil's Aeneid rhymes almost every line.