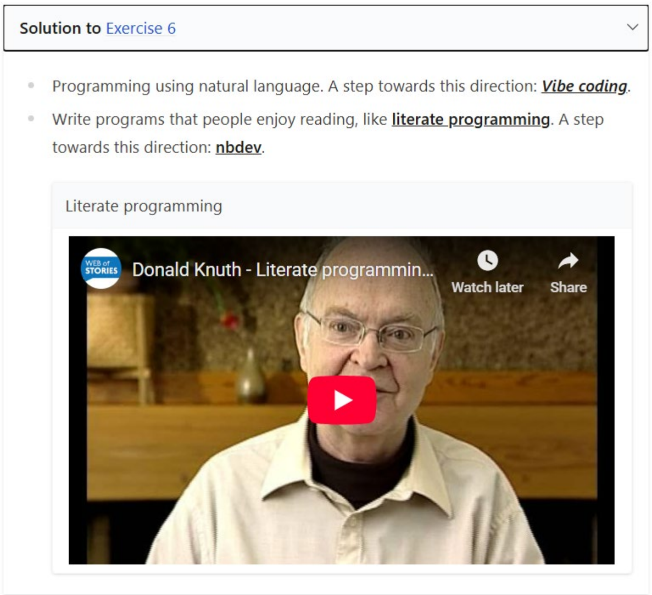
Rewind for the missed parts in last week (Lecture 1):

(1) Let's view the following videos (from Lecture 1 Notebook):

The Art of Writing Software
https://www.youtube.com/watch?v=QdVFvsCWXrA
skip to 2:11 why not machine/assembly language

Types of programming languages
https://www.youtube.com/watch?v=aYjGXzktatA
(1:18 - 3:05)

Donald Knuth – Literate Programming
https://www.youtube.com/watch?v=bTkXg2LZIMQ&t=8s
(The link is from https://ccha23.github.io/cs1302i25b/introduction-to-computer-programming#ex-future-programming)



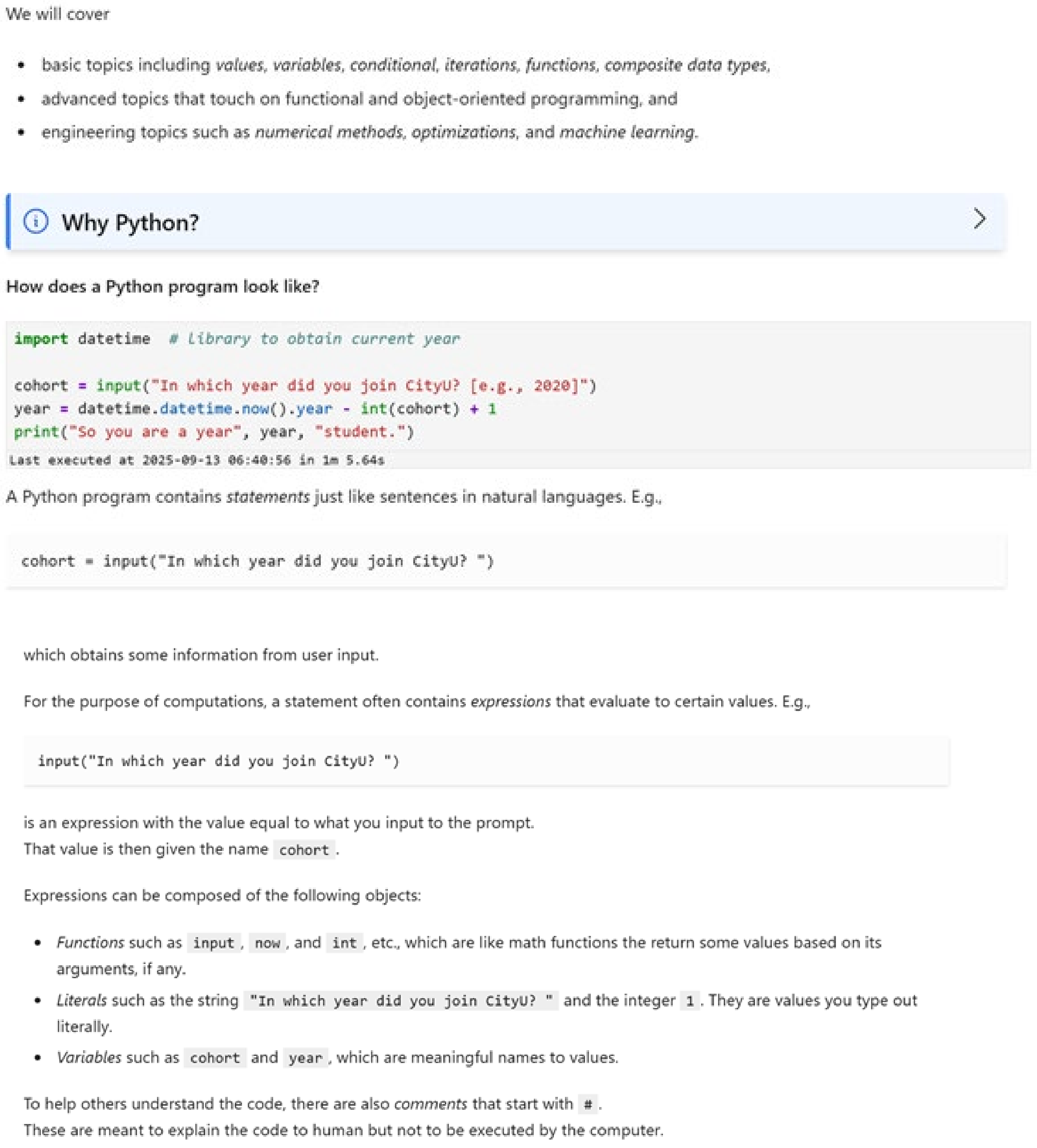
Solution to Exercise 6

- Programming using natural language. A step towards this direction: ***Vibe coding***.
- Write programs that people enjoy reading, like **literate programming**. A step towards this direction: **nbdev**.

Literate programming

(2) *What we will cover* (from Lecture 1 Notebook):

Recall what we tried last week:
print(2+3)
print(int(input()) + int(input()))



We will cover

- basic topics including *values, variables, conditional, iterations, functions, composite data types,*
- advanced topics that touch on functional and object-oriented programming, and
- engineering topics such as *numerical methods, optimizations*, and *machine learning*.

ⓘ Why Python? ›

How does a Python program look like?

```
import datetime  # Library to obtain current year

cohort = input("In which year did you join CityU? [e.g., 2020]")
year = datetime.datetime.now().year - int(cohort) + 1
print("So you are a year", year, "student.")
```

```
Last executed at 2025-09-13 06:40:56 in 1m 5.64s
```

A Python program contains *statements* just like sentences in natural languages. E.g.,

```
cohort = input("In which year did you join CityU? ")
```

which obtains some information from user input.

For the purpose of computations, a statement often contains *expressions* that evaluate to certain values. E.g.,

```
input("In which year did you join CityU? ")
```

is an expression with the value equal to what you input to the prompt.
That value is then given the name `cohort`.

Expressions can be composed of the following objects:

- *Functions* such as `input`, `now`, and `int`, etc., which are like math functions the return some values based on its arguments, if any.
- *Literals* such as the string `"In which year did you join CityU? "` and the integer `1`. They are values you type out literally.
- *Variables* such as `cohort` and `year`, which are meaningful names to values.

To help others understand the code, there are also *comments* that start with `#`.
These are meant to explain the code to human but not to be executed by the computer.
