Think Python 2e, Chapter 1 Notes

Geoffrey Matthews

September 11, 2022

What is a program?

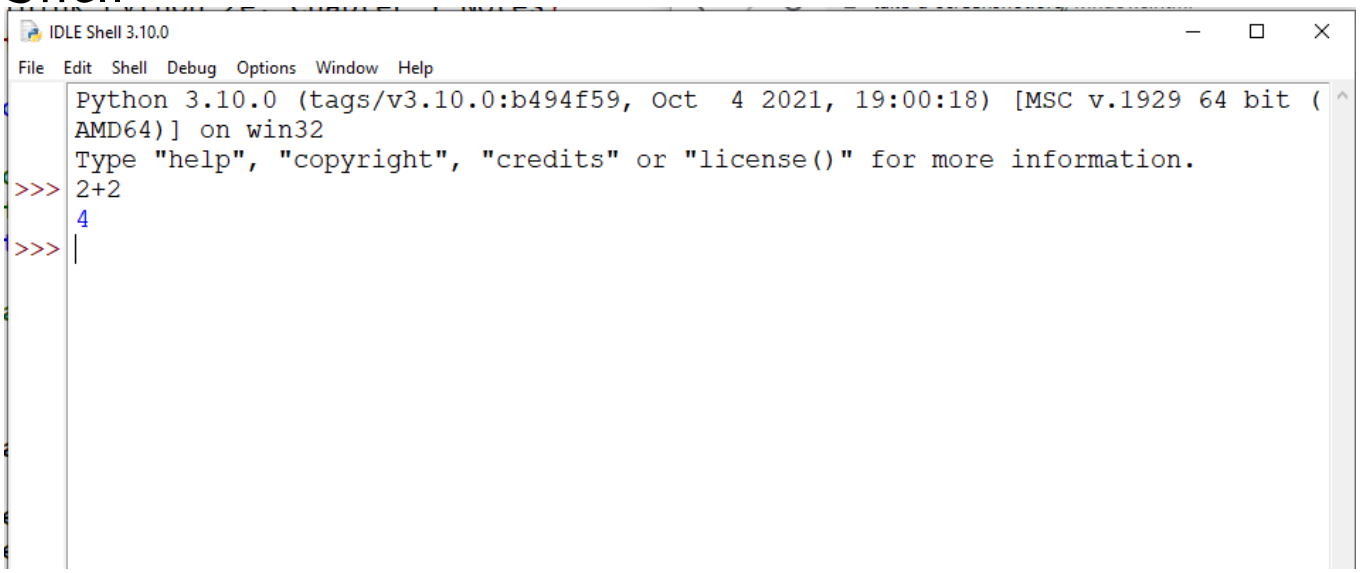
- A sequence of instructions that specifies how to perform a computation.
- Examples:
  - solve a system of equations
  - search and replace text in a document
  - play a video
  - sharpen an image
  - run a simulation

Five kinds of instructions

- Input
- Output
- Math
- Conditional execution
- Repetition

Idle

- Shell



- Editor

Shell

- Also called **interpreter** or **REPL**
  Read-Eval-Print-Loop
- Great place to learn about Python
- Not so great for larger programs.
  - Enter them in the **Editor**
  - Run them: input and output will be in the shell
- First program: `print("Hello world!")`
  - Run this in the shell
  - Enter this in the editor, then run module

Arithmetic

```
2 + 2

5 * 5

3 / 4

3 // 4

3 % 4

2 ** 3
```

Arithmetic: use parentheses!

```

5 - 4 - 3 - 2 - 1

2**3**4

2 + 2/3 + 3

2 * 2/3 * 3
```

Arithmetic: use parentheses!

```

5 - 4 - 3 - 2 - 1

2**3**4

2 + 2/3 + 3

2 * 2/3 * 3
```

```

((((5 - 4) - 3) - 2) - 1)

2**(3**4)

(2 + 2)/(3 + 3)

(2 * 2)/(3 * 3)
```

Types

```
type(22)

type(3.14159)

type('hello')

type("hello")

type('22')
```

Types

```
type(22)

type(3.14159)

type('hello')

type("hello")

type('22')
```

Cats have four legs.

"Cats" has four letters.

Formal languages and natural languages

- **Natural languages:** English, Spanish, Chinese.
  - Not designed by people.
  - Evolved over thousands of years to serve many purposes.
  - Ambiguous.

    He called her a hacker, and she insulted him.

Formal languages and natural languages

- **Formal languages:** math, chemistry, programming
  - Designed by people for specific purposes.

Programming languages are formal languages that have been designed to express computations.

Syntax and Semantics

Good semantics, bad syntax:

> Me want go store, you takem, kay?

Good syntax, bad semantics:

> Colorless green ideas sleep furiously.

Syntax: tokens and structure

Good structure, bad tokens:

> This is @ well-structured Engli$h sentence with invalid t*kens in it.

Bad structure, good tokens:

> This sentence all valid tokens has, but invalid structure with.

Tokens in programming languages.

```
1 print(333+1756.45)
```

$$\underbrace{\texttt{print}}_{token}\ \underbrace{(}_{token}\ \underbrace{333}_{token}\ \underbrace{+}_{token}\ \underbrace{1756.45}_{token}\ \underbrace{)}_{token}$$

Parsing

- The process of finding the tokens and checking for correct structure is called **parsing**.
- The first task of a computer when processing your programs is to parse them.
- Any errors in tokens or structure are reported back to you as syntax errors.
- Nothing can be done with a program with syntax errors.

Parsing

- The process of finding the tokens and checking for correct structure is called **parsing**.
- The first task of a computer when processing your programs is to parse them.
- Any errors in tokens or structure are reported back to you as syntax errors.
- Nothing can be done with a program with syntax errors.

Any homework program submitted with syntax errors will get zero credit!

Vocabulary

high-level language: A programming language like Python that is designed to be easy for humans to read and write.

low-level language: A programming language that is designed to be easy for a computer to run; also called "machine language" or "assembly language".

portability: A property of a program that can run on more than one kind of computer. interpreter:] A program that reads another program and executes it

prompt: Characters displayed by the interpreter to indicate that it is ready to take input from the user.

program: A set of instructions that specifies a computation.

print statement: An instruction that causes the Python interpreter to display a value on the screen.

Vocabulary

operator: A special symbol that represents a simple computation like addition, multiplication, or string concatenation.

value: One of the basic units of data, like a number or string, that a program manipulates.

type: A category of values. The types we have seen so far are integers (type int), floating-point numbers (type float), and strings (type str).

integer: A type that represents whole numbers.

floating-point: A type that represents numbers with fractional parts.

string: A type that represents sequences of characters.

Vocabulary

natural language: Any one of the languages that people speak that evolved naturally.

formal language: Any one of the languages that people have designed for specific purposes, such as representing mathematical ideas or computer programs; all programming languages are formal languages.

token: One of the basic elements of the syntactic structure of a program, analogous to a word in a natural language.

syntax: The rules that govern the structure of a program.

parse: To examine a program and analyze the syntactic structure.