Think Python 2e, Chapter 2 Notes

Geoffrey Matthews

September 13, 2022

Assignment statements

```
message = 'Hello world!'
n = 14 + 3
pi = 3.14169265
```

- Variables name places in computer memory.
- An assignment statement places data at the place represented by the variable.
- An assignment statement changes the **state** of the computer.
- We represent the state of a computer at a particular time with a **state diagram:**

message	⟶	'Hello world!'
n	⟶	17
pi	⟶	3.14159265

Variable name syntax

```
>>> 76trombones = ’big parade’
SyntaxError: invalid syntax
>>> more@ = 1000000
SyntaxError: invalid syntax
>>> class = ’Advanced Theoretical Zymurgy’
SyntaxError: invalid syntax
```

Python keywords

```
False       class       finally     is          return
None        continue    for         lambda      try
True        def         from        nonlocal    while
and         del         global      not         with
as          elif        if          or          yield
assert      else        import      pass
break       except      in          raise
```

Expressions *vs.* Statements

Expressions:

```
42
n
n + 25
’hello’
```

Statements:

```
n = 17
print(n)
print(’hello’)
print(14 + 3)
n = n * 2
```

String operations

```
>>> first = ’throat’
>>> second = ’warbler’
>>> first + second
throatwarbler
>>> ’spam’ * 3
spamspamspam
```

Comments

```
# compute the percentage of the hour that has elapsed
percentage = (minute*100) / 60
```

```
percentage = (minute*100) / 60 # percentage of an hour
```

Good and Bad Comments

Good comment:

```
v = 5       # velocity in meters/second.
```

Bad comment:

```
v = 5       # assign 5 to v
```

Debugging

Syntax error: These are found by Python when you try to run the program, or use the check module button.

Runtime error: These errors are not found by Python until the program is running. An example would be dividing by zero or taking the square root of a negative number. Also called **exceptions**. We say that the running program raised an exception.

Semantic error: An error in telling the computer what you mean. For example, if you wanted the speed in kilometers per hour, but your instructions calculated speed in miles per hour, the program would compute an incorrect result and give you the wrong answer with no warning.

The computer *always* does exactly what you say, but only *sometimes* does what you mean.

Vocabulary

variable: A name that refers to a value.

assignment: A statement that assigns a value to a variable.

state diagram: A graphical representation of a set of variables and the values they refer to.

keyword: A reserved word that is used to parse a program; you cannot use keywords like if, def, and while as variable names.

operand: One of the values on which an operator operates.

Vocabulary

expression: A combination of variables, operators, and values that represents a single result.

evaluate: To simplify an expression by performing the operations in order to yield a single value.

statement: A section of code that represents a command or action. So far, the statements we have seen are assignments and print statements.

execute: To run a statement and do what it says.

Vocabulary

interactive mode: A way of using the Python interpreter by typing code at the prompt.

script mode: A way of using the Python interpreter to read code from a script and run it.

script: A program stored in a file.

Vocabulary

comment: Information in a program that is meant for other programmers (or anyone reading the source code) and has no effect on the execution of the program.

syntax error: An error in a program that makes it impossible to parse (and therefore impossible to interpret).

exception: An error that is detected while the program is running.

semantics: The meaning of a program.

semantic error: An error in a program that makes it do something other than what the programmer intended.