Programming, or Computational Problem Solving:
What, Why, When, Where

• Which (Python)/How:
  • Python primitive elements
  • Combining primitive elements/Expressions
  • Python assignment
  • Example problem

Talk is cheap; show me the code.

A computer program implements an algorithm on a computer.
A computer program is (therefore) a set of instructions written in a language the computer can understand.

A program is a set of instructions written in a language the computer can understand.

Which language is that?

Natural language?
Flip switches?
Programming language
How to Program:

Programming languages like Python are
- Formal
- Precise
- Unambiguous
- Readable

Why Python?
- Accessible to entry level programmers and also for experts – like chess or tennis
- Python is widely used in many fields
- Interactive, high-level, syntax-lite language - Exploring is easy: concentrate on problem-solving not the language itself – harness the power of the computer
- Lots of built in functionality and support libraries ("batteries included")
- General purpose, multiple paradigm language and syntax support straightforward transition to C++, Java
- Popular, well supported, good documentation and development environments.

HOW: For Python (or any computer language)

- What are the primitive elements?
- How can we combine the primitive elements?
- How can we extend the language?

Python primitive elements are called

Objects
  value
  type
  id

For example, the number 4 is a Python object.

```python
>>> 4
4

>>> type(4)
<class 'int'>

>>> id(4)
4297261248
```
Python types are associated with operations.

Integer operations, for example:

```
+, -, *, /  
also: //, %, **, pow
```

Operations are also objects:

```python
>>> pow
<built-in function pow>

>>> type(pow)
<class 'builtin_function_or_method'>

>>> id(pow)
4298373168
```

Objects can be combined in expressions

E.g., `>>> 44 + 55`

Expressions are evaluated and return a value

```python
>>> 44 + 55
99
```

Expressions are evaluated and return a value

```python
>>> 4 + 5
9
```
a) 10 + 4
b) 10 - 4
c) 10 + -4
d) 10 + 4.0
e) --7
f) +7
g) 10 + 4 / 2
h) (10 + 4) / 2

Python Primitive Elements

Variables/Assignment

Evaluate the expression of the rhs of =

Store the result in a memory location

Make the variable name on the lhs of = refer to that memory location

Python Primitive Elements

Variables/Assignment

Evaluate the expression of the rhs of =

Store the result in a memory location

Make the variable name on the lhs of = refer to that memory location

Python Primitive Elements

Variables/Assignment

Evaluate the expression of the rhs of =

Store the result in a memory location

Make the variable name on the lhs of = refer to that memory location

Python Primitive Elements

Variables/Assignment

Evaluate the expression of the rhs of =

Store the result in a memory location

Make the variable name on the lhs of = refer to that memory location

Python Primitive Elements

Variables/Assignment

Evaluate the expression of the rhs of =

Store the result in a memory location

Make the variable name on the lhs of = refer to that memory location

Python Primitive Elements

Variables/Assignment

Evaluate the expression of the rhs of =

Store the result in a memory location

Make the variable name on the lhs of = refer to that memory location

Python Primitive Elements

Variables/Assignment

Evaluate the expression of the rhs of =

Store the result in a memory location

Make the variable name on the lhs of = refer to that memory location
Python Primitive Elements

Variables/Assignment

```python
>>> example = 99
>>> example
99
>>> try-this = 4
SyntaxError: invalid syntax
```
Solving a problem using Python:

- you have $50 and are buying some movies that cost $15 each. Write an expression that shows how much money you have left after buying \( n \) movies. Evaluate the expression when \( n = 2 \) and \( n = 3 \).

```python
start_cash = 50
movie_cost = 15
n = 2
end_cash = start_cash - (movie_cost * n)
```