CIS 122 Introduction to Programming and Problem Solving  Spring 2017

✓ Programming is Computational Problem Solving

✓ Computational Problem Solving is
  Computational Thinking + Coding

Tools for programming / Programming languages/
Python

Intro to Python
  Objects / type / numbers (integer and float) /
  numeric operations

Variables and assignment
Expressions and evaluating expressions

---

CIS 122 Introduction to Programming and Problem Solving  Spring 2017

TASK ↔ Computational Thinking ↔
  ALGORITHM ↔ Coding ↔
  COMPUTER PROGRAM → Execute →
  SOLUTION

Create/revise/understand

*Programming = Computational Thinking + Coding*
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.

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; supports straightforward transition to C++, Java, other languages
• Popular, well-supported, good documentation and development environments.

HOW: 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

For example, the number 4 is a Python object.

```python
>>> 4
4
```
but hello and 4 5 6 are not objects:

```python
>>> hello
Traceback (most recent call last):
  File "<pyshell#0>", line 1, in <module>
    hello
NameError: name 'hello' is not defined

>>> 4 5 6
SyntaxError: invalid syntax
```

Objects have a type

```python
>>> 4
4

>>> type(4)
<class 'int'>
```
Object type is associated with
range of values
operations

integer values – 0, 1, 2, 3, -99, ...

integer operations, for example:

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

For example,

>>> 44 + 55
99

But not

>>> len(44)

Traceback (most recent call last):
  File "<pyshell#0>", line 1, in <module>
    len(44)
TypeError: object of type 'int' has no len()

>>> len
<built-in function len>
Objects can be combined in expressions
   e.g., >>> 44 + 55

Expressions are evaluated and return a value

>>> 44 + 55
99

>>> 55  # a simple expression
55

Expressions are evaluated and return a value

>>> 44 + 55
99

>>> 2**3
8

>>> pow(2, 3)
8

>>> 4+-*
Expressions are evaluated and return a value

```python
>>> 44 + 55
99

>>> 2**3
8

>>> pow(2, 3)
8

>>> 4+-*
SyntaxError: invalid syntax
```

```python
>>> 9 / 3
3.0

>>> type(3.0)
<class 'float'>

>>> 9 / 4
2.25

>>> 9 // 4
2

>>> 9 % 4
1
### CIS 122 Introduction to Programming and Problem Solving  Spring 2017

1. a) $10 + 4$
   
2. b) $10 - 4$
   
3. c) $10 + -4$
   
4. d) $10 + 4.0$
   
5. e) $--7$
   
6. f) $+-7$
   
7. g) $10 + 4 / 2$
   
8. h) $(10 + 4) / 2$

### CIS 122 Introduction to Programming and Problem Solving  Spring 2017

1. a) $2/3$
   
2. b) $2//3$
   
3. c) $1/3 * 15$
   
4. d) $1//3 * 15$
   
5. e) $(10 + 4) \% 2$
   
6. f) $10 + 4 \% 2$
   
7. g) $9 \% 2$
   
8. h) $2**6$
   
9. i) $-2**2$
   
10. j) $(-2)**2$
   
11. k) $1+3 * 5$
   
12. l) $2^6$  # challenge
Python Primitive Elements

Variables/Assignment

>>> example = 99
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

>>> example = 99

assignment statements are not expressions
assignment statements do not evaluate to a value
Python Primitive Elements

Variables/Assignment

>>> example = 99

assignment statements are not expressions
assignment statements do not evaluate to a value

**variable names are expressions**
do evaluate to a value

>>> example
99

>>> try-this = 4
*SyntaxError: invalid syntax*
Python Primitive Elements
Variables/Assignment

```python
>>> 122class = 125

>>> cis122 = 125 * 3

>>> CIS122

SyntaxError: invalid syntax

>>> cis122 = 125 * 3
>>> CIS122

NameError: name 'CIS122' is not defined
```