Computational Structures in Data Science

Lecture #2: Algorithmic Structures


September 2, 2016

http://inst.eecs.berkeley.edu/~cs88

Requirements for CS61b and CS Major

- Data8+CS88 qualify you for CS61b
- CS majors: Need to take CS47a any time after CS88 to fulfill requirements.

Computational Concepts today

- Algorithm, Code, Data, Information
- Data Types, Simple Data Structures
- Function Definition Statement
- Conditional Statement
- Iteration

Algorithm

- An algorithm (pronounced AL-go-rith-um) is a procedure or formula for solving a problem.
- In mathematics and computer science, an algorithm is a self-contained step-by-step set of operations to be performed.
- An algorithm is an effective method that can be expressed within a finite amount of space and time and in a well-defined formal language for calculating a function.

Algorithms early in life

Carry (MSD)
Operator + Operands

Least significant digit of result

Algorithms early in life (in binary)

Carry (MSD)
Operator + Operands

LSB result

09/02/16 UCB CS88 Fa16 L2

4

09/02/16 UCB CS88 Fa16 L2

5
More Terminology (Dictionary)

- Code
  A system of symbols (as letters or numbers) of communication

- Data
  Facts and statistics collected together for reference or analysis

- Information
  Facts provided or learned about something or someone

Experiment

Algorithm vs Code vs Data vs Information

Data or Code?

Data or Code?

Data or Code?

Data or Code?

Human-readable code (programming language)

Machine-executable instructions (byte code)

Compiler or Interpreter

Here: Python
Language Structures (Python)

- Variables and literals
  - with some internal representation, e.g. Integers, Floats, Booleans, Strings, ...
  - In Python: Implicit data types!

- Operations on variable and literals of a type
  - e.g. +, -, /, //, **
  - ==, <, >, <=, >=

- Expressions are valid well-defined sets of operations on variables and literals that produce a value of a type.
  - e.g. 3+4*3

More Language Structures (Python)

- Data type: values, literals, operations, e.g., int, float, string
- Expression
  - \[3.1 \times 2.6\]
- Call expression
  - \[\text{max}(0, x)\]
- Variables
  - Assignment Statement
  - \[x = \text{<expression>}\]
- Control Statement
  - if ... (see later)
- Sequences: tuple, list
  - \[\text{numpy.array}(\text{<object>})\]
- Data structures
  - \[\text{numpy.array}, \text{Table}\]
- Tuple assignment
  - \[x, y = \text{<expression>}\]

Call Expressions

- Evaluate a function on some arguments

What would be some useful functions?

- Built-in functions
  - \[\text{min}, \text{max}, \text{sum}\]
  - \[\text{https://docs.python.org/3/library/functions.html}\]
  - \[\text{https://docs.python.org/3/library/}\]
  - \[\text{str}\]
  - \[\text{import math; help(math)}\]

Defining Functions

- Generalizes an expression or set of statements to apply to lots of instances of the problem
- A function should do one thing well

```python
def \text{<function name>} \text{<argument list>} : \\
  \text{return \text{expression}}
```

Conditional statement

- Do some statements, conditional on a predicate expression

```python
\text{if} \text{<predicate>}: \\
  \text{<true statements>}
\text{else:} \\
  \text{<false statements>}
```

for statement – iteration control

- Repeat a block of statements for a structured sequence of variable bindings

```python
\text{<initialization statements>}
\text{for} \text{<variables> in} \text{<sequence expression>}: \\
  \text{<body statements>}
\text{<rest of the program>}
```
**while statement – iteration control**

- Repeat a block of statements until a predicate expression is satisfied

```c
<initialization statements>
while <predicate expression>:
    <body statements>
<rest of the program>
```

**Data-driven iteration**

- describe an expression to perform on each item in a sequence
- let the data dictate the control

```c
[ <expr with loop var> for <loop var> in <sequence expr> ]
```

**By the Way…**

- Could we build a computer that has no instructions, only data?

Yes! The One Instruction Set Computer.

Check it out: [https://en.wikipedia.org/wiki/One_instruction_set_computer](https://en.wikipedia.org/wiki/One_instruction_set_computer)

**Questions?**