Lecture 28: Implementing Interpreters

Reason 1: Vocational Skill
Job listings at monster.com in Virginia (27 March 2007, postings in last 3 months):
- **Python**: 27, $40-200K
- **Java**: 770, $35-200K
- **SQL**: 1138, $60-400K
- **Scheme**: 55, $100-999K

Reason 2: Expanding Minds
Languages change the way we think.

The more languages you know, the more different ways you have of thinking about (and solving) problems.

“Jamais Jamais Jamais” from *Harmonice Musices Odhecaton A*. Printed by Ottaviano Dei Petrucci in 1501 (first music with movable type)
Python

• A universal programming language
  – Everything you can compute in Scheme you can compute in Python, and vice versa

• Imperative Language
  – Designed to support a programming where most of the work is done using **assignment statements**: \( x = e \)

• Object-Oriented Language
  – Every data thing is an object
  – Built in support for classes, inheritance

Reason 3: Deepening Understanding

By seeing how the same concepts we encountered in Scheme are implemented by a different language, you will understand those concepts better (especially classes/objects, assignment, data abstraction).

Reason 4: Building Confidence

By learning Python (mostly) on your own, the next time you encounter a problem that is best solved using a language you don’t know, you will be confident you can learn it (rather than trying to use the wrong tool to solve the problem).

Reason 5: Fun

Programming in Python is fun.

Especially because:
• It is an elegant and simple language
• Most programs mean what you think they mean
• It is dynamic and interactive
• It can be used to build web applications (PS8, PS9)
• It is named after *Monty Python’s Flying Circus*
• It was designed by someone named Guido.

Learning New Languages

• Syntax: Where the `, ;`, `,`, etc. all go
  – If you can understand a BNF grammar, this is easy

• Semantics: What does it mean
  – Learning the evaluation rules
  – Harder, but most programming languages have very similar evaluation rules

• Style
  – What are the idioms and customs of experienced programmers in that language?
    • Takes many years to learn
    • Need it to be a “professional” Python programmer, but not to make a useful program
### Python If

Instruction ::= if (Expression) : Block

Evaluate Expression. If it evaluates to true, evaluate the Block.

It is similar to (if Expression (begin Statements))

Differences:
- Indenting and new lines matter!
- Changing the indentation changes meaning of code
- What "true" means:
  - Scheme: anything that is not #f
  - Python: anything that is not False, None, 0, and empty string or container

### Computability in Theory and Practice

(Intellectual Computability Discussion on TV Video)

#### Ali G Problem

- **Input:** a list of 2 numbers with up to \(d\) digits each
- **Output:** the product of the 2 numbers

Is it computable?

Yes – a straightforward algorithm solves it. Using elementary multiplication techniques it is \(O(d^2)\)

Can real computers solve it?

#### Ali G was Right!

- Theory assumes ideal computers:
  - Unlimited, perfect memory
  - Unlimited (finite) time
- Real computers have:
  - Limited memory, time, power outages, flaky programming languages, etc.
  - There are many computable problems we cannot solve with real computer: the actual inputs do matter (in practice, but not in theory!)
Implementing Interpreters

Inventing a Language

- Design the grammar
  - What strings are in the language?
  - Use BNF to describe all the strings in the language
- Make up the evaluation rules
  - Describe what everything the grammar can produce means
- Build an evaluator
  - A procedure that evaluates expressions in the language

Is this an exaggeration? (SICP, p. 360)

It is no exaggeration to regard this as the most fundamental idea in programming:

The evaluator, which determines the meaning of expressions in the programming language, is just another program.

To appreciate this point is to change our images of ourselves as programmers. We come to see ourselves as designers of languages, rather than only users of languages designed by others.

Environmental Model of Evaluation

1. To evaluate a combination, evaluate all the subexpressions and apply the value of the first subexpression to the values of the other subexpressions.
2. To apply a compound procedure to a set of arguments, evaluate the body of the procedure in a new environment. To construct this environment, make a new frame with an environment pointer that is the environment of the procedure that contains places with the formal parameters bound to the arguments.

\[
def\ \text{meval}(expr,\ env):
\begin{align*}
\text{if } \text{isPrimitive}(expr): \\
\quad \text{return } \text{evalPrimitive}(expr) \\
\text{elif } \text{isConditional}(expr): \\
\quad \text{return } \text{evalConditional}(expr, env) \\
\text{elif } \text{isLambda}(expr): \\
\quad \text{return } \text{evalLambda}(expr, env) \\
\text{elif } \text{isDefinition}(expr): \\
\quad \text{evalDefinition}(expr, env) \\
\text{elif } \text{isName}(expr): \\
\quad \text{return } \text{evalName}(expr, env) \\
\text{elif } \text{isApplication}(expr): \\
\quad \text{return } \text{evalApplication}(expr, env) \\
\text{else:} \\
\quad \text{evalError}(\text{"Unknown expression type: } + \text{str(expr)})
\end{align*}
\]

Implementing meval

Eval and Apply are defined in terms of each other.
```python
def mapply(proc, operands):
    if isPrimitiveProcedure(proc):
        return proc(operands)
    elif isinstance(proc, Procedure):
        params = proc.getParams()
        newenv = Environment(proc.getEnvironment())
        if len(params) != len(operands):
            evalError("Parameter length mismatch: ...")
        for i in range(0, len(params)):
            newenv.addVariable(params[i], operands[i])
        return meval(proc.getBody(), newenv)
    else:
        evalError("Application of non-procedure: %s" % (proc))
```

### Charge

- Friday: Implementing the rest of the interpreter: the evaluation rules, environments, procedures
- Next week: changing the evaluation rules
- Don’t wait any longer to start PS7
- The statement, “There will be a surprise quiz someday this week” might still be true