61A LECTURE 20 – SOCIAL IMPLICATIONS

Steven Tang and Eric Tzeng
July 29, 2013

Announcements
- Midterm this Thursday
- Information posted later today
- 3 exam rooms:
  - aa-gz 2050 VLSB
  - ha-jz 2060 VLSB
  - ka-zz 2040 VLSB
- Project 4 out today
  - Writing a Scheme interpreter in Python
  - “Recursive Art” Contest!

“Recursive Art” Contest!
- Create a visualization of an iterative or recursive process of your choosing, using turtle graphics. Your implementation must be written entirely in Scheme using the interpreter you have built.
- Prizes will be awarded for the winning entry in each of the following categories, as well as 3 extra credit points.
  - Featherweight. At most 256 tokens of Scheme, not including comments and delimiters.
  - Heavyweight. At most 2013 tokens of Scheme, not including comments and delimiters.
- Winners will be selected by popular vote as part of a future homework
- More details online later today!

Past winners
- Look at the side screens!

Midterm Survey Results
- Most people are happy with the way things are!
  - Great!
- Some suggestions:
  - 8am is a terrible time for lecture
  - I agree! Sorry about that.
  - Lectures should contain more jokes
  - Pace of course is too fast
    - In the regular semester, this is a common complaint.
    - No doubt it’s exacerbated during the condensed summer schedule, but we do need to cover the same amount of material
  - Use a grading curve instead of the absolute scale
    - The absolute scale can only help your grade. If grades are too low at the end of the semester, we may add points to everyone's grades to bring up the average.
**Read-Eval-Print Loop**

The user interface to many programming languages is an interactive loop, which

- Reads an expression from the user,
- Parses the input to build an expression tree,
- Evaluates the expression tree,
- Prints the resulting value of the expression

The REPL handles errors by printing informative messages for the user, rather than crashing

A well-designed REPL should not crash on any input!

---

**Raising Application Errors**

The – and / operators have restrictions on argument number

Raising exceptions in apply can identify such issues

```python
def calc_apply(op, args):
    """Apply an operator to a list of args."""
    if op == '+':
        if len(args) == 0:
            raise TypeError('Not enough arguments')
    ... 
    if op == '/':
        if len(args) == 2:
            raise TypeError('Not enough arguments')
    ...
```

---

**Social Implications**

- After the CS61 series, you will become a very capable programmer
- It’s very possible that the code you write in the next few years could affect millions of people
- Life/death situations, catastrophic accidents, privacy invasions, intellectual property disputes... Read Ch. 1. of Blown to Bits
- Be aware of how the code you write can influence the world

---

**Social Implications**

- CS161 - Computer Security. UC Berkeley has several world-renowned security Professors.