

# Scripting Languages G22.3033-002 Summer 2008

## hw10

Assigned Th 7/24/2008, due Fr 8/1 at 9pm. 50 points.

<http://www.cs.nyu.edu/courses/summer08/G22.3033-002/>

### Homework instructions

Homeworks are due on Fridays at 9pm. This deadline will be strictly enforced.

Email your answers to Robert Soulé [robert.soule@gmail.com](mailto:robert.soule@gmail.com). Please put your solutions to VBA programming problems in a powerpoint presentation. For all other questions (including programming problems in other languages), just send a simple text file, such as what you get when using Emacs, Vi, Notepad, or the "save as text only" feature in Word.

Please make sure that your code works with the compilers and tools installed at CIWW. Specifically, please test:

#### JAVASCRIPT

Your JavaScript code must run with **both** Mozilla's Firefox browser **and** with Microsoft's Internet Explorer browser.

PHP Your PHP code must run with PHP 5.1.6 on the CIMS web servers, see <http://www.cims.nyu.edu/systems/userservices/webhosting/>.

PERL Your Perl code must run with Perl 5.8.8 for Linux/x86, for example on doowop1 (see <http://www.cims.nyu.edu/systems/resources/computeservers/>).

VBA Your VBA code must run with Microsoft Office 2003 for Windows on the machines in the labs CIWW 502 or CIWW 624.

### Concept questions

#### hw10-1 Iterating with Coroutines

(4+6 = 10 points) Consider the following Python script.

```
#!/usr/bin/env python
class Tree:
    def __init__(self, value, left=None, right=None):
        self.value = value
        self.left = left
        self.right = right
    def preorder(self):
        yield self.value
        if self.left:
            for v in self.left.preorder():
                yield v
        if self.right:
            for v in self.right.preorder():
                yield v

my_tree = Tree(1,
```

```
Tree(2,Tree(4),Tree(5)),
Tree(3,Tree(6),Tree(7)))
```

```
def print_preorder(tree):
  print 'preorder:'
  for y in my_tree.preorder():
    print y

print_preorder(my_tree)
```

- Predict what the code should print. Then run it. What does it print?
- Add another generator method `Tree.inorder` that walks the tree in-order instead of pre-order. An in-order walk first visits all nodes in the left subtree, then the current node, then all nodes in the right subtree. The following picture shows a tree for which an in-order walk would visit nodes in alphabetical order:

```
      d
     / \
    b   f
   / \ / \
  a  c e  g
```

After you write the generator `Tree.inorder`, the following code:

```
def print_inorder(tree):
  print 'inorder:'
  for y in my_tree.inorder():
    print y

print_inorder(my_tree)
```

should cause the following output to be printed:

```
inorder:
4
2
5
1
6
3
7
```

## hw10-2 Iterating with Block Parameters

(4+6 = 10 points) Consider the following Ruby script.

```
#!/usr/bin/env ruby
class Tree
  attr_accessor :value, :left, :right
  def initialize(value, *leftRight)
    @value = value
    if 0 < leftRight.length()
      @left = leftRight[0]
```

```

        @right = leftRight[1]
    end
end
def preorder()
    yield @value
    if @left
        @left.preorder {|v| yield v }
    end
    if @right
        @right.preorder {|v| yield v }
    end
end
end

$my_tree = Tree.new(1,
                    Tree.new(2, Tree.new(4), Tree.new(5)),
                    Tree.new(3, Tree.new(6), Tree.new(7)))

def print_preorder(tree)
    puts 'preorder:'
    tree.preorder {|y| puts y.to_s + "\n" }
end

print_preorder($my_tree)

```

- Predict what the code should print. Then run it. What does it print?
- Add another top-level function `print_sum` that uses the iterator to compute the sum of all values in the tree. For example, the following call:

```
print_sum($my_tree)
```

should cause the following output to be printed:

```
sum:
28
```

### hw10-3 Coroutines vs. Block Parameters

(7+4+4=15 points)

- Briefly explain the difference between coroutines and block parameters.
- Give an advantage of coroutines compared to block parameters.
- Give an advantage of block parameters compared to coroutines.

### Programming exercises

#### hw10-4 Iterating with Closure Parameters (PHP)

(15 points) Consider the following skeleton PHP script:

```
#!/usr/bin/env php
<?php
```

```

class Tree {
  # --- add your code here ---
}

$myTree = new Tree(1,
                  new Tree(2, new Tree(4), new Tree(5)),
                  new Tree(3, new Tree(6), new Tree(7)));

function printPreorder($tree) {
  echo "preorder:\n";
  $tree->preorder(create_function('$v', 'echo "$v\n";'));
}

printPreorder($myTree);
?>

```

Implement the body of class `Tree` so that the construction (`new Tree(...)`) and iteration (`$tree->preorder(...)`) at the bottom of the script work, and produce the same output as the equivalent Python and Ruby scripts above.