Homework 03 - Project Revision

Andrew Carter and Beryl Egerter

February 12, 2013

1 Scheduled Interviews

Our scheduled interviews:

1. Interview with NW at 4:30pm on Friday, February 15
2.

2 Description of Topic

3 Empirical Goals

4 Response to Participant Questions

We realize that much of the code that we plan to ask participants to look at will be unfamiliar to them. Though we are interested in how they go about familiarizing themselves with the code, we do not intend to completely baffle them. Thus, we will answer any technical questions such as how the language work. We plan to answer any question that could be answered with a quick google search. We will not answer any questions about how the program is composed.

5 Non-Linear Interview Questions

Our questions are aimed to find a level that challenges the interviewee so that we can see the process more clearly. If an interviewee finds the Tier 1 Questions hard, we will not advance to Tier 2 Questions, and likewise for the Tier 2 Questions. We have prepared two questions each in Tier 1 and 2, and one question in Tier 3. Thus we can start with short, less complicated questions in Tier 1 and move up to Tier 3 which contains concepts the interviewee is unlikely to have seen if we believe the interviewee can handle it.

6 Interview Sequence and Questions

1. Welcome participant.
2. Read assent script to participant.
3. Solicit and answer any questions the participant has.
4. Have participant sign the consent form.
5. Provide participant a copy of the consent form.
6. Turn on the video camera.
7. Have participant work on Question 1.1

(a) Introduce question:

(b) Hand participant the attached handout, Question 1.1

(c)

8. If participant was not confident on Question 1.1, have participant work on Question 1.2
9. If participant was confident enough on Question 1.1 or 1.2, have participant work on Question 2.1
10. If participant was not confident on Question 2.1, have participant work on Question 2.2
11. If participant was confident enough on Question 2.1 or 2.2, have participant work on Question 3.1

7 Handouts

Attached below.

Question 1.1

```
def func2(list, num):
        return func1(list, num, func4)

def func4(a, b):
        return a * b

def func1(list, num, f):
        acc = 0
        for i in list:
                acc += f(i, num)
        return acc

def main():
    print(func3([1,2,3,4]))

def func3(list)
    return func2(list, 4)

main()
```

Question 1.2

```
def function50(i, L):
    return L[i+2]

def function37(L):
    return L[-1]+L

def function52(i):
    return function4() * i

def function1(j, k):
    return (j + k) * function52(1)

def function4():
    return 3

def function188(L):
    return function37(L)+function50(2, L)

def function0():
    return function188([1,2,3,4,5,6,7,8,9])[function1(0,1)]

function0()
```

Question 2.1, File 1

```
#!/usr/bin/ruby

load "ourdate.rb"

d = OurDate.new(2011,1,4)
print "#{d.what_day}"
print "We started writing this file today.\n"
d.forward_time(365)
print "We are almost done now.\n"
print "#{d.what_day}"
```

Question 2.1, File 2

```
#!/usr/bin/env ruby

$months31 = [1,3,5,7,8,10,12]
$months30 = [4,6,9,11]

class OurDate
    attr_accessor :year
    attr_accessor :month
    attr_accessor :day

    def initialize(year, month, day)
        @year = year
        @month = month
        @day = day
    end

    def is_equal?( d )
        puts @year == d.year and
            @month == d.month and
            @day = d.day
    end

    def is_leap_year?
        if @year % 400 == 0
            puts 1
        elsif @year % 100 == 0
            puts nil
        elsif @year % 4 == 0
            puts 1
        else
            puts nil
        end
    end
```

```
def check_month
    if @month == 13
        @month = 1
        @year = @year + 1
    elsif @month == 0
        @month = 12
        @year = @year - 1
    end
end

def tomorrow
    @day = @day + 1
    if @day > 31
        for i in $months31
            if @month == i
                @day = 1
                @month = @month + 1
                check_month
            end
        end
    elsif @day > 30
        for i in $months30
            if @month == i
                @day = 1
                @month = @month + 1
                check_month
            end
        end
    elsif @day > 28 and @month == 2
        @day = 1
        @month = @month + 1
        check_month
    end
end

def yesterday
```

```
        @day = @day - 1
        if @day == 0
            @month = @month - 1
            check_month
            for i in $months31
                if @month == i
                    @day = 31
                end
            end
            for i in $months30
                if @month == i
                    @day = 30
                end
            end
            if @month == 2
                @day = 28
            end
        end
    end

    def forward_time(n)
        for i in 0..n
            tomorrow
        end
    end

    def reverse_time(n)
        for i in 0..n
            yesterday
        end
    end

    def what_day
        puts "Today is #{month}/#{day}, #{year}!"
    end
end
```

Question 2.2, File 1

```
#!/bin/env python3
from random import choice
from sys import stdin

class Board(object):
    def __init__(self, width=7, height=6):
        self.board = [[] for i in range(width)]
        self.width = 7
        self.height= 6
    def drop(self, player, column):
        if column < len(self.board):
            self.board[column].append(player)
            return True
        return False
    def __str__(self):
        result = ""
        for r in reversed(range(self.height)):
            result += "|"
            for c in range(self.width):
                if r < len(self.board[c]):
                    result += self.board[c][r]
                else:
                    result += " "
                result += "|"
            result += "\n"
        result += "-" * (2 * self.width + 1)
        return result
    def full(self):
        return all(len(col) >= self.height for col in self.board)
    def score(self, player):
        for c in range(self.height):
            for r in range(len(self.board[c])):
                p = self.board[c][r]
                for dc,dr in ((0,1),(1,0),(1,1),(1,-1)):
                    for i in range(1,4):
                        nc = c + i*dc
                        nr = c + i*dr
                        if nc < 0 or self.width <= nc:
                            break
                        if nr < 0 or len(self.board[nc]) <= nr:
                            break
                        if self.board[nc][nr] != p:
                            break
                    else:
                        return 1 if p == player else -1
        return 0

other = {’X’ : ’O’, ’O’ : ’X’}
player = ’X’
board = Board()

while True:
    try:
        c = int(input("%s > " % player))
    except TypeError:
        continue
    if not board.drop(player,c):
        continue
    print(board)
    if board.score(player):
        print("Player %s Wins!!!" % player)
    elif board.full():
        print("Tie")
    else:
        player = other[player]
        continue
    board = Board()
    player = ’X’
    print(board)
```