# Homework 05 - Interviews and Content Logs

### Andrew Carter and Beryl Egerter

### February 28, 2013

## 1 Interview: ACBE3

### 1.1 Post Interview Reflection (Joint)

- Syntax error - immediately said was never on the line the error gives

- Should have asked people about their previous experiences such as grutoring, which could make a huge difference in looking at code

- Huge variety of skill levels we have been getting

- Much better at finding the problem without understanding everything else than previous participants

- Got both of the bugs in Ruby and continued, but stopped immediately after finding bug in python

- debugging problems, resorted to top down (except when given an error with line number)

- for the first two, really didn't need to write on paper to keep track of what was going on

- was thrown off by list syntax in python where code enclosed in brackets means different things in different places

### 1.2 Interview Summary

Interview length: 53:36m
No technical difficulties encountered. We went through Questions 1.1, 1.2, 2.1, and 2.2, and 3.1 (3.1 cut off by time constraints).

### 1.3 Most Interesting Pieces

- Time: 2:30 - 2:50 (Confusion over func4, in Question 1.1)
  5:28 - 8:25 (Confusion over lists, 1.2)
  Summary: In Question 1.1 participant was confused about func2 using

func4. After reading func1 was able to recover quickly. In Question 1.2, participant was confused about bracket notation and indexing, but was again able to recover quickly.

Relations to Topic: This relates to our topic because he still manages to understand most of the code even though he initially didn't understand the syntax. In particular he looked for other clues in the code to help him understand the syntax.

Possible Arguments: Some programmers (perhaps with more experience) may be able to reason about a program even with an incorrect understanding of syntax, and are able to identify unlikely situations, search for a suitable replacement semantics, and modify their view of the code accordingly without too much trouble.

- Time: 25:50 - 33:25
  Summary: After having solved the bug in Question 2.2, participant was asked if he understood the entire program flow. Participant was hung up on how the score function would return -1. The program would never return -1, but the participant focused a lot of effort towards figuring out what was going on in this return to evaluate program flow.

  Relations to Topic: This relates to our topic because he solved the bug very quickly before hand, and from a debugging standpoint was not worried about the score function. However when asked about the overall program flow, got bogged down in an impossible return case.

  Possible Arguments: When trying to comprehend a program, some programmers assume that all parts of the program have functionality, and focus on trying to use dead code. This behavior could differ between debugging and comprehending.

## 1.4   Content Log

Content log is provided in separate text file labeled ACBE3.

# 2 Interview: ACBE4

## 2.1 Post Interview Reflection (Joint)

- Interesting that changed methods between 1.1 and 1.2

- also, differences between starting point in 1.1 and 1.2 vs. what large clues existed (main function v. not main function)

- If we had switched order of first and second would it have affected things?

- Very verbose

- Much better understanding of functions in 1.1 than in 1.2

- Assumptions that previous coders were intelligent

- Ideas of basic functions, or 'important' functions

## 2.2 Interview Summary

Interview length: 47:47
Technical difficulties:

- No color formatting on paper sheets. Didn't appear to be a serious problem, grayscale affect was visible to differentiate keywords so it wasn't a total loss of information.

- Time constraints meant we needed to hurry/not ask many follow up questions in Question 2.2 - also, participant stuck around briefly after camera was turned off, even though had given us a hard time constraint, which meant that part of the answer to the last question is not on tape.

We went through Questions 1.1, 1.2, 2.1, and 2.2.

## 2.3 Most Interesting Pieces

- Time: 14:40 - 15:40
  Summary: Participant describes differences between Questions 1.1 and 1.2. Mentions that function calls branch more in 1.1. Also mentions that 1.1 has a main function that was a logical start of execution, but in 1.2, had to scan through code to find where statements first occur instead of function definitions.
  Relations to Topic: Gives a sense of what a programmer may first look for in code to find a place to start.
  Possible Arguments: While a programmer can just start from top going down, analyzing code, an efficient approach may be identifying the thread of execution by clues found in function names or syntax.

- Time: 22:50 - 23:48

  Summary: In Question 2.1, participant identifies is_leap_year? as a function to pay attention to. Then, by looking through all the code and seeing that the function is not called anywhere, participant claims that leap year support is not implemented. Participant did recognize that it was possible that actual code implemented without the function.

  Relations to Topic: Shows how a programmer can pay attention to just top level constructs (function names and how functions interact) than the actual code computations.

  Possible Arguments: When debugging specific problems in unfamiliar code, keywords in function names and function calls are an efficient strategy.

## 2.4   Content Log

Content log is provided in separate text file labeled ACBE4.

# 3 Reflect on Expected Focus

Possible Foci :

1. Does the participant look at what the function does or just the name, and to what extent does this affect success and confidence of correctness?

   - Is this focus still relevant? Why? What may have changed about the focus?
     This focus is still relevant because we found that often participants were tripped up on syntax and complicated logic. The most successful and confident answers to questions we obtained occurred when the participant only looked at function names and function calls.

   - What data might be relevant?
     Interview 4, Question 2.1, Interview 3, Question 2.2, Interview 1, Question 2.2, Interview 2, Question 2.2, where the first two are more successful, and the latter two contain instances of confusion.

   - What is the revised focus based upon your interviews?
     Does the participant delve into the details of a function or simply or does the participant quickly obtain a high level summary of the function, and how does this affect success and the confidence of correctness in the solution?

   - How would you rate this focus and your interest in conducting an analysis based upon this focus? (1 = highest rating/highest interest; 5 = lowest rating/lowest interest)
     Rating: 1

2. Did the participant even look at all of the code, or did they quickly (and correctly) narrow down the region that needed to be changed?

   - Is this focus still relevant? Why? What may have changed about the focus?
     This focus is still relevant because we got varying solutions where the participant had explored a large number of functions or very few.

   - What data might be relevant?
     Interviews 3 and 4 often didn't look at all the code, while Interviews 1 and 2 mostly looked at all the code (Interview 2, less so). Also, we believe that the experience levels of our participants ranged from low to high going in order of 1, 2, 3 | 4.

   - What is the revised focus based upon your interviews?
     Is there a correlation between experience level and the amount of code looked at in order to find a region that needs to be changed?

   - How would you rate this focus and your interest in conducting an analysis based upon this focus? (1 = highest rating/highest interest; 5 = lowest rating/lowest interest)
     Rating: 5

3. In what order did the participant explore the functions, and to what depth each time?

  - Is this focus still relevant? Why? What may have changed about the focus?
    This focus is still relevant because individual behavior changed depending on the question, as well, for the first two questions, we got a varied mix of participants starting in different places and different orders.

  - What data might be relevant? All of Interview 4 comparing questions within to each other, Interview 1, Question 1.1 (top-down), Interview 2, Question 1.1 (execution order), interview 3, Question 1.1 (execution order), Interview 4, Question 1.1 ('basic' functions into execution order)

  - What is the revised focus based upon your interviews?
    In what order did the participant explore the functions? Did this depend on the type of program or the type of question?

  - How would you rate this focus and your interest in conducting an analysis based upon this focus? (1 = highest rating/highest interest; 5 = lowest rating/lowest interest)
    Rating: 3

# 4   New Focus?

N/A