

CpSc 372: Introduction to Software Development
Spring 2008

Questionnaire and Skill Set Survey

Information Concerning Participation in an NSF-Funded Research Study
(“Computer-Aided Collaborative Reasoning across the Curriculum”)

Overview. We would like to request your participation in this survey as part of a research study being conducted by Dr. Jason O. Hallstrom, Dr. Joan Krone, Dr. Richard Pak, and Dr. Murali Sitaraman. The purpose of this study is to evaluate the teaching and learning benefits of new curriculum materials and software tools that will be introduced as part of CpSc 372 during the Spring 2008 semester. The survey consists of three parts. First, you will be asked to briefly describe your opinions concerning various aspects of the software development process. Second, you will be asked to rate your level of agreement with a series of statements concerning your own software development practices, and software development practices in general. Finally, you will be asked to complete two short exercises related to developing and understanding software.

Benefits. Your participation in this survey has the potential to improve your learning experience during the semester, and to improve Computer Science education beyond your graduation.

Confidentiality. We will do everything we can to protect your privacy. Your identity will not be revealed in any publication that might result from this study. All feedback data will be collected and stored anonymously. **You are, however, asked to sign your submission with the “secret code” that you used for the preliminary questionnaire.** This will allow us to compare your results without revealing your identity.

Participation. Your participation in this research study is voluntary. You may choose not to participate, and you may withdraw your consent to participate at any time. Your involvement and evaluation in the course will be unaffected by your decision. There is no competitive advantage to participating, nor is there a competitive disadvantage.

Contacts. If you have any questions or concerns about this study or if any problems arise, please contact Dr. Jason O. Hallstrom at 864.656.0187. If you have any questions or concerns about your rights as a research participant, please contact the Clemson University Office of Research Compliance at 864.656.6460.

Part II. Please rate your level of agreement with each of the following statements. You will be asked to select from six options: *strongly disagree*, *disagree*, *moderately disagree*, *moderately agree*, *agree*, *strongly agree*. The statements are a matter of *opinion*. There are no right or wrong answers.

1. If I worked for a company and was asked to develop 10,000 lines of software to solve a problem, I would be capable of designing and implementing that software.
 strongly disagree disagree moderately disagree moderately agree agree strongly agree

2. My coursework has prepared me well for a software development career.
 strongly disagree disagree moderately disagree moderately agree agree strongly agree

3. The difficulty in understanding and modifying a 10,000 line software system has more to do with the style in which the software is written, and less to do with how smart I am.
 strongly disagree disagree moderately disagree moderately agree agree strongly agree

4. The difficulty in understanding and modifying a 10,000 line software system has more to do with the style in which the software is written, and less to do with the programming language in which it is written.
 strongly disagree disagree moderately disagree moderately agree agree strongly agree

5. Software development can benefit from carefully designing each component before coding it, as opposed to quickly coding and experimenting with it.
 strongly disagree disagree moderately disagree moderately agree agree strongly agree

6. The main challenge of software construction lies in coding the design in a programming language, and not so much in specifying what needs to be done.
 strongly disagree disagree moderately disagree moderately agree agree strongly agree

7. There are benefits to showing that a software component is correct without running it on a computer.
 strongly disagree disagree moderately disagree moderately agree agree strongly agree

8. It is possible to show that a software component is correct without actually running it on the computer.
- strongly disagree disagree moderately disagree moderately agree agree strongly agree
9. The development of reliable software has a lot to do with mathematical reasoning.
- strongly disagree disagree moderately disagree moderately agree agree strongly agree
10. To understand and reason about a program built using a component, you need to understand all the statements inside the component.
- strongly disagree disagree moderately disagree moderately agree agree strongly agree
11. Testing software thoroughly is the most important way to ensure software correctness.
- strongly disagree disagree moderately disagree moderately agree agree strongly agree
12. It is easy to combine components from different team members and produce working software.
- strongly disagree disagree moderately disagree moderately agree agree strongly agree
13. I believe that there is a strong correlation between a person's mathematical background and their ability to design and implement large systems correctly.
- strongly disagree disagree moderately disagree moderately agree agree strongly agree
14. When I've completed a software component, I often doubt whether that component is 100% correct.
- strongly disagree disagree moderately disagree moderately agree agree strongly agree
15. To guarantee correctness, it is best to develop a system from scratch.
- strongly disagree disagree moderately disagree moderately agree agree strongly agree

16. Having precise mathematical descriptions for each software component improves the likelihood that my software will be correct.
- strongly disagree disagree moderately disagree moderately agree agree strongly agree
17. The programming language used is an important factor in successful software development.
- strongly disagree disagree moderately disagree moderately agree agree strongly agree
18. When working in teams, natural language (*e.g.*, English) descriptions of the different components are sufficient for communication among team members.
- strongly disagree disagree moderately disagree moderately agree agree strongly agree
19. Before I run my code on a computer, I make it a practice to hand trace through the statements on example inputs to see if it works.
- strongly disagree disagree moderately disagree moderately agree agree strongly agree
20. Reasoning about programs involving components requires a thorough understanding of pointers and/or references.
- strongly disagree disagree moderately disagree moderately agree agree strongly agree
21. Working with a friend makes it easier to reason about a program.
- strongly disagree disagree moderately disagree moderately agree agree strongly agree
22. Working with a friend to complete classroom activities is fun.
- strongly disagree disagree moderately disagree moderately agree agree strongly agree
23. My conception of what software is has changed significantly over time.
- strongly disagree disagree moderately disagree moderately agree agree strongly agree

24. My conception of the difficulty associated with developing high quality software has changed significantly over time.

strongly disagree disagree moderately disagree moderately agree agree strongly agree

25. My conception of how to build high quality software has changed significantly over time.

strongly disagree disagree moderately disagree moderately agree agree strongly agree

Part III. Please complete each of the following exercises. The exercises are designed to assess your ability to reason about existing software, and to develop new software.

1. Consider the following C function. After reviewing the source code, answer each of the questions concerning the points marked with “?”.

```
void bar(int A, int B, int C) {
    int X1, X2;
    int R;
    int S1, S2;

    if(A >= B)
        X1 = A;
    } else {
        X1 = B;
    }
    // ? (a)

    if(B >= C) {
        X2 = B;
    } else {
        X2 = C;
    }
    // ? (b)

    if(X1 >= X2) {
        R = X1;
    } else {
        R = X2;
    }
    // ? (c)

    S1 = 0;
    S2 = 100;
    while(S2 > 0) {
        S2--;
        S1++;
        // ? (d)
    }
    // ? (e)
}
```

- a.) What facts can you state concerning the values of variables A, B, and X1 at point (a)?
- b.) What facts can you state concerning the values of variables B, C, and X2 at point (b)?
- c.) What facts can you state concerning the values of variables A, B, C, and R at point (c)?
- d.) What facts can you state concerning the values of variables S1 and S2 at point (d)?
- e.) What facts can you state concerning the values of variables S1 and S2 at point (e)?

-
2. In the space below, provide a complete implementation of a reusable `stack` component. Include at least three operations. You may use any language of your choice.