An Empirical Study of Programming Bugs in CS1, CS2, and CS3 Homework Submissions
Morgan Hall, Keri Laughter, Jessica Brown, Chelynn Day, Christopher Thatcher, Renee Bryce
Computer Science & Engineering Department
University of North Texas
Renee.Bryce@unt.edu

ABSTRACT
In this work, we use the IEEE Standard for Software Anomalies to classify the types of bugs that CS1, CS2, and CS3 students submit on programming assignments over the course of one semester. We also classify the types of bugs that students bring to a Computer Science Tutor Lab so that we can compare the types of bugs that students seek help for in comparison to those in their homework submissions. Using nine high level categories, Logic problems are the most common type of problem brought to the tutor lab (58% of tutor visits) and also the most frequent as observed on homework submissions (30%). However, the frequency of Logic problems brought to the tutor lab was quite higher than those in homework submissions. Computational and Data problems accounted for much of this difference. These results are being used in our ongoing work that strives to help students to avoid the most common bugs that are brought to our tutor lab and submitted on assignments.

1. INTRODUCTION
In this paper, we discuss the results of the types of bugs that students from CS1, CS2, and CS3 courses submit in programming assignments and compare the results to the types of bugs that brought to our Computer Science Tutor Lab. We collect this data to better understand the types of problems that students encounter.

Our main motivation behind this study comes from a survey that we conducted in 2010. We notice that approximately 50% of our students withdraw from the Computer Science major. To better understand why students changed majors, our earlier work administered an optional online survey to our CS1 students. There were 92 responses to the survey. We asked if anything in their course was frustrating and if so, to elaborate on the frustrations. We provided a textbox for students to enter their response so that we did not introduce bias by giving them examples to select from. Programming bugs and debugging were the most frequently reported frustrations and were reported by 38% of the students.

Previous work has taken different approaches to analyze student programming bugs and debugging strategies. For instance, Fitzgerald et al. identify that locating bugs is often more difficult than correcting them [4,5]. Bryce et al. classify the types of programming bugs that students bring to a tutor lab over a one-year period and discover that students most often seek help with bugs related to problem solving [2]. Pedersen et al. have students log data about their bugs on several parallel programming projects and classify the types of bugs that students
encounter [8]. This previous work can feed into solutions to improve debugging related problems.

Many novel approaches promote making software testing fun in order to help students. Due to space limitations, we highlight only three examples. Werner et al. make software testing concepts fun through the use of games for students [9]. Elbaum et al. provide “BugHunt” as an engaging online resource that teaches students about software testing [3]. Bryce et al. use an exercise called “Bug Wars” where students create buggy code and challenge each other to find their bugs [1].

The work in this paper makes a new contribution as we look at bugs that are submitted on programming assignments in addition to those brought to our tutor lab. In the remainder of this paper, section 2 describes how we collected and classified data, section 3 discusses results, section 4 discusses on going work, and section 5 discusses future work.

Figure 1 - Part 1 of the tutor form (available at http://bryce.cs.usu.edu/tutor)

3. DATA COLLECTION AND CLASSIFICATION
We collect data for this study from two sources and classify the bugs. First, students and tutors work together to formally document problems are brought to the Tutor Lab. Second, we use random samples of students in CS1, CS2, and CS3 courses and classify the bugs (if any) that they submit in each programming assignment throughout the semester.

Classification
We use the IEEE Standard for Software Anomalies to classify data [7]. Due to space limitations, we report the data using nine high level categories that are applicable in our study: Logic,
Tutor Form
Our department provides a Computer Science Tutor Lab that is paid for through mandatory course fees. Students may visit the tutor lab as often as they wish at no additional cost. Students that visited our tutor lab documented data about their bugs, including their course, programming language, instructor, description of the assignment, number of variables and functions, and a summary of the bug and how they solved it. They also uploaded their code for us to cross check with the data that they submitted. Figure 1 shows a screenshot from the web-based form that we use. We report the data for the Spring 2012 semester.

Homework Assignments
We examine the results from a randomly selected group of 30 students from CS1, 30 students from CS2, and 19 students from CS3. The CS3 sample was smaller due to the size of the section. For every coding assignment, we identified any bugs and classified them. Written assignments that did not contain code were excluded from the study. This includes 5 assignments in CS1, 11 assignments for CS2, and 10 assignments for CS3. We report the data for the Spring 2012 semester.

3. RESULTS
Classified data
Figure 2 provides data on the frequency of each type of bug reported in the tutor lab and submitted on homework assignments. Logic problems were the most common in both homework submissions and tutor logs. However, the overall contributions of Logic Problems were much higher in tutor logs (58.15%) as compared to homework submissions (26.9%). Some examples of logic problems in a submission include: forgetting cases or steps, missing condition tests, duplicating logic, using/creating unnecessary functions, misinterpretation, checking the wrong variable, and iterating a loop incorrectly.

Data handling bugs (19.82%) were the second most common type of problem brought to the tutor lab. Example problems that students brought include: incorrect initialization, unset flags, referencing data incorrectly, referencing data out of bounds, using the wrong variable type, subscripting variables incorrectly, and problems with scope. Data handling bugs were also prevalent in homework submissions as 54 assignment submissions included such a bug and they contributed to 12.41% of the total bugs in the homework submissions.

Computational Problems (19.08%) were the third most common bug in homework submissions. Commonly observed Computational Problems include: missing computations, incorrect operands or operators, parenthesis used incorrectly, and rounding or truncation mistakes. Computational Problems were also brought to the tutor lab, but only account for 7.93% of the bugs brought to the tutor lab. Indeed, approximately ¾ of the bugs brought to the tutor lab involved either Logic or Data Handling.

All of the Interface/Timing, Data, Documentation, Interoperability, and Standards Conformance Problems account for only 14.1% of the total bugs brought to the tutor lab. On the other hand,
these categories contribute to 41.61% of the total bugs submitted on assignments. The largest differences are that of Data, Standards Conformance, and Interoperability Problems. Examples of Data Problems included incorrect operator data and incorrect output data. Examples of Standards Conformance Problems include a student using a struct instead of a class or not following requirements for an assignment. While a program sometimes technically worked, if they did not conform to assignment requirements, students lost points. Interoperability Problems that were reported include submitting files with .txt extensions instead of .cpp or .h extensions.

We acknowledge that no silver bullet is likely to exist for the majority of problems. Nonetheless, creative solutions that focus on common problems have the potential to help many students. In the remainder of this paper, we present our ongoing work that uses the data that we collected to create a recommendation system that allows students to enter data about their bugs and then provides them with recommendations of bugs that they should avoid in the future.

<table>
<thead>
<tr>
<th>Classification Type</th>
<th>CS1</th>
<th>CS2</th>
<th>CS3</th>
<th>Total</th>
<th>Overall %</th>
</tr>
</thead>
<tbody>
<tr>
<td>Logic</td>
<td>53</td>
<td>64</td>
<td>15</td>
<td>132</td>
<td>58.15</td>
</tr>
<tr>
<td>Computational</td>
<td>9</td>
<td>7</td>
<td>2</td>
<td>18</td>
<td>7.93</td>
</tr>
<tr>
<td>Interface/Timing</td>
<td>3</td>
<td>6</td>
<td>1</td>
<td>10</td>
<td>4.41</td>
</tr>
<tr>
<td>Data Handling</td>
<td>18</td>
<td>26</td>
<td>1</td>
<td>45</td>
<td>19.82</td>
</tr>
<tr>
<td>Data</td>
<td>2</td>
<td>4</td>
<td>0</td>
<td>6</td>
<td>2.64</td>
</tr>
<tr>
<td>Documentation</td>
<td>1</td>
<td>12</td>
<td>0</td>
<td>13</td>
<td>5.73</td>
</tr>
<tr>
<td>Interoperability</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>0.88</td>
</tr>
<tr>
<td>Standards Conformance</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0.44</td>
</tr>
<tr>
<td>Total</td>
<td>86</td>
<td>121</td>
<td>20</td>
<td>227</td>
<td>100.00</td>
</tr>
</tbody>
</table>

(a) Bugs reported in Tutor Lab

<table>
<thead>
<tr>
<th>Classification</th>
<th>CS1</th>
<th>CS2</th>
<th>CS3</th>
<th>Total</th>
<th>Overall %</th>
</tr>
</thead>
<tbody>
<tr>
<td>Logic</td>
<td>23</td>
<td>51</td>
<td>43</td>
<td>117</td>
<td>26.90</td>
</tr>
<tr>
<td>Computational</td>
<td>44</td>
<td>37</td>
<td>2</td>
<td>83</td>
<td>19.08</td>
</tr>
<tr>
<td>Interface/Timing</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>5</td>
<td>1.15</td>
</tr>
<tr>
<td>Data Handling</td>
<td>5</td>
<td>39</td>
<td>10</td>
<td>54</td>
<td>12.41</td>
</tr>
<tr>
<td>Data</td>
<td>15</td>
<td>43</td>
<td>9</td>
<td>67</td>
<td>15.40</td>
</tr>
<tr>
<td>Documentation</td>
<td>16</td>
<td>0</td>
<td>0</td>
<td>16</td>
<td>3.68</td>
</tr>
<tr>
<td>Interoperability</td>
<td>17</td>
<td>0</td>
<td>0</td>
<td>17</td>
<td>3.91</td>
</tr>
<tr>
<td>Standards Conformance</td>
<td>15</td>
<td>0</td>
<td>16</td>
<td>31</td>
<td>7.13</td>
</tr>
<tr>
<td>Total</td>
<td>138</td>
<td>171</td>
<td>81</td>
<td>390</td>
<td>100.00</td>
</tr>
</tbody>
</table>

(c) Bugs submitted in homework assignments

Figure 2 - Summary of results
4. ONGOING AND FUTURE WORK

Our ongoing and future work continues to collect data about programming bugs that students encounter. We are using the data to create a set of “bug recommendations” that are representative of bugs that we observe. Each recommendation includes an example of buggy code, correct code, and a description of why the respective code is buggy or correct. Figure 3 shows a brief example of a bug recommendation. Our hypothesis is that if we can better understand the types of bugs that students encounter over time, we can predict the types of bugs that individual students will encounter and reduce their frustration by providing them with examples of bugs that they should try to avoid in the future. Our current prototype is under evaluation and our future work will provide an evaluation of whether our recommendations that are based on real student bugs are an effective resource.

```
Example Result
Type: File I/O
Problem Statement: Forgetting to include fstream
Buggy Code:
#include <iostream>
using namespace std;

int main()
{
    ofstream myFile;
    myfile.open("myFile.txt");
    myfile << "I like to write to this file!" << endl;
    myFile.close;
    return 0;
}

Discussion of Error: Leaving out fstream means that your compiler does not have access to all the functions required for file I/O.
Correct Code:
#include <iostream>
#include <fstream>
using namespace std;
int main()
{
    ofstream myFile;
    myfile.open("myFile.txt");
    myfile << "I like to write to this file!" << endl;
    myFile.close;
    return 0;
}
```

Figure 3 - Example of a Bug Recommendation

5. CONCLUSIONS

In this work, we classified 390 bugs from homework submissions in CS1, CS2, and CS3 courses. We compared these results to 227 bugs that students brought to our tutor lab. While Logic problems were the most frequent type of bug for both scenarios, they account for a much larger percentage of the bugs brought to the tutor lab. Computational and Data Handling bugs were also frequently encountered. Interface/Timing, Data, Documentation, Interoperability, and Standards Conformance Problems account for only 14.1% of the total bugs brought to the tutor lab while they account for 41.61% of the bugs in homework submissions. This data allows us to better understand the types of programming bugs that our introductory students encounter. Our ongoing and future work uses the data from these studies to create a set of examples that teach students to avoid common programming bugs. Our BugFlix recommendation system allows
students to enter data about their programming bugs and then provides students with examples of bugs that they should avoid. The recommendations of bugs to avoid are based on the data that we collected on the homework submissions and tutor logs in this paper. Our ongoing and future work assesses the BugFlix system to learn whether the system and the recommendations are useful to students.

Acknowledgements
We thank the National Science Foundation for their support through grant TUES-1139840. We also thank the Computing Research Association – Women (CRA-W) and the Coalition to Diversify Computing (CDC) for their support through the Collaborative Research Experience for Undergraduates (CREU) program. We thank Christopher Weber, Arjun RoyChoudhury, Sandeep Nannuri, Nawaf Omar Alsrehin, and our focus group for their contributions.

References


