Programming Perception Differences among CS and non-CS majors

By James Penton
Advisor: Dr. Clark Turner
June 8, 2012

Table of Contents

Programming Perceptions

- Definition
  - Significance: the amount of effort required for the programming task, in terms of difficulty, time, and monetary resources.
- Problem
  - Software developers and customers are not always on the same page
- General Question
  - How differently to programmers and non-programmers view programming task significance?

Programming Perceptions

- Thesis Question
  - Is there a significant difference in how computing and non-computing majors view programming task significance?
- Hypothesis
  - Students in computing majors (CSC/CPE/SE) will be better able to differentiate between significant and insignificant programming tasks than those in non-computing majors.

Related Work

- Previous surveys
  - Mashups (Zang & Rossum)
    - Concluded that most internet users do not understand mashups well enough to correctly identify the difficulty in creating one
  - Non-CS competence (Lurain & Weinshank)
    - Students do not need to be able to program in order to understand programming concepts
  - New CS student competence (McCracken et. al)
    - New CS students are not performing up to expectations regarding programming skills

Related Work (cont.)

- Surveys
  - OO Correlations (Ramalingam & Weidenback)
    - Older study that found programming comprehension is greater in students who learn an object oriented language
  - *Information Week* (Scaffidi et. al)
    - Program experts get familiar with a few program features, and use those throughout all of their programs

Related Work (cont.)

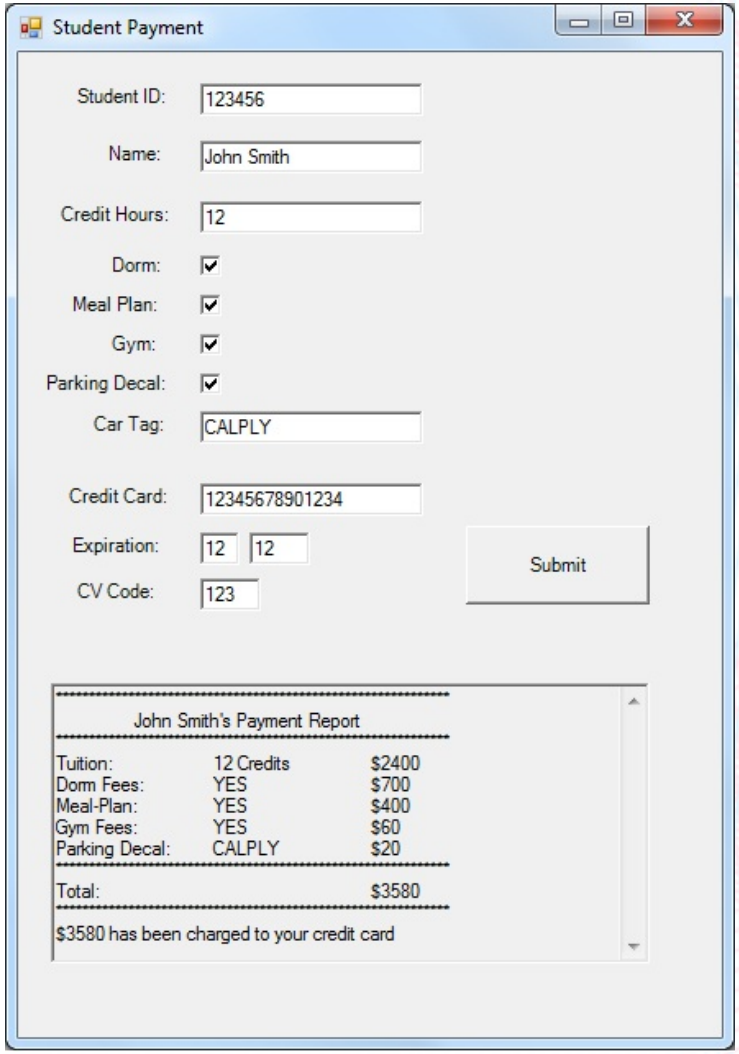
- Studies
  - Math Backgrounds (Pioro)
    - Students who took calculus and discrete math before their first programming course had higher grades
  - Usability (Bevan & Azuma)
    - Various definitions I will probably use in thesis (effectiveness, efficiency, satisfaction)
  - Programmer Mentality (Maiden & Sutcliffe)
    - Explains how expert software developers abstract differently than novice programmers

Design

- Survey!!!
  - Personal Information
  - Scenario involving software upgrades
    - Different upgrades done by 3 teams
      - Team 1: Verification
      - Team 2: UI Upgrades
      - Team 3: Database/Email
  - Questions involving amount of effort, functionality, and user preference of each upgrade
  - Questions to gauge computing competence

Initial Program

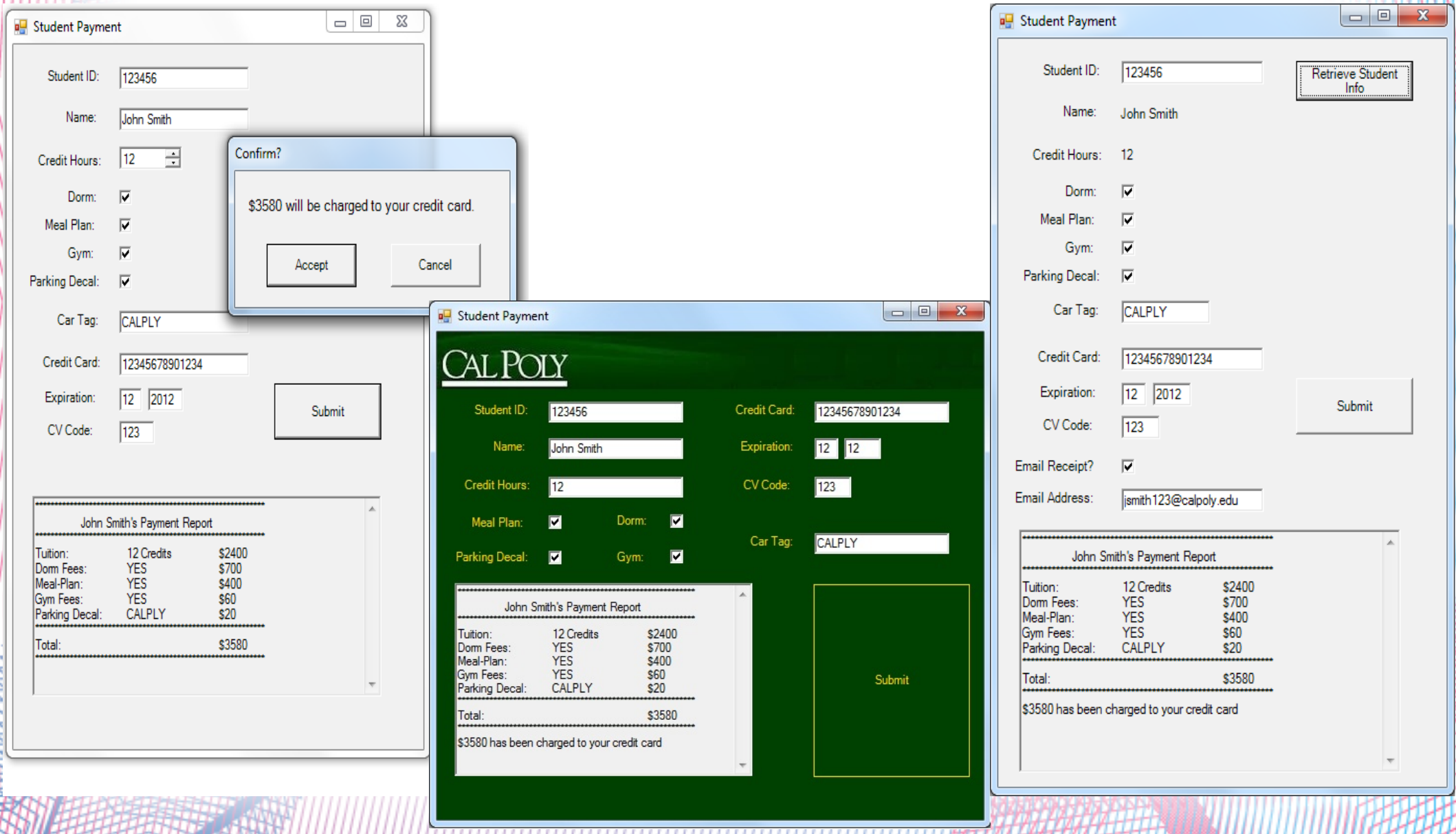
- Student Fee Calculator
  - Input your ID, name, and various information
  - Click submit
  - Receipt printed to screen
- Very simple and easy to understand



Team 1(Verification)

Team 2(Graphics)

Team 3(DB/email)



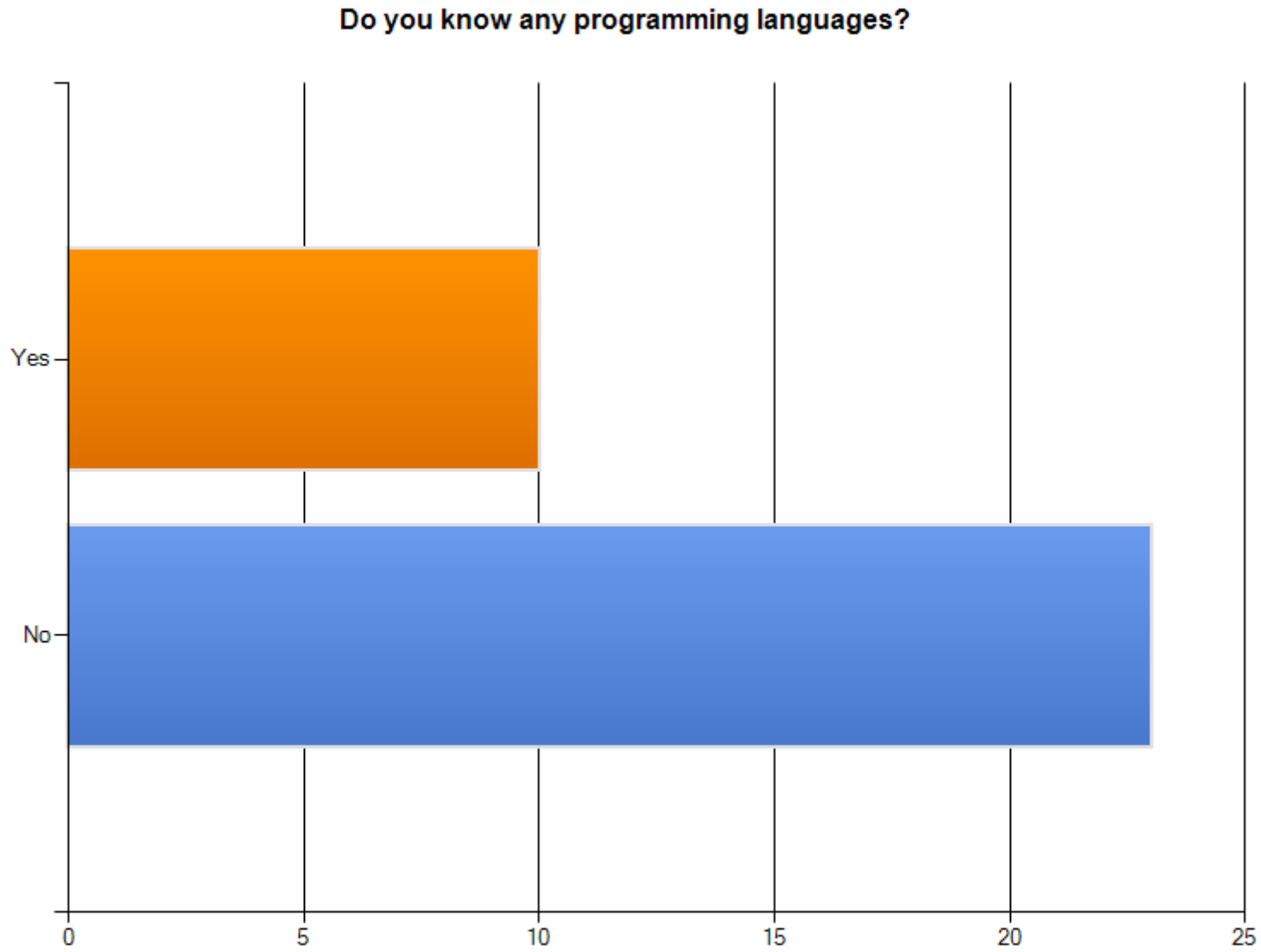
Programming Tasks used in Survey

- Team 1 (Verification):
  - Field data verification of alpha or numeric characters
  - Popup confirmation dialogue
- Team 2 (Interface):
  - UI improvements, including color scheme and field rearrangement
- Team 3 (Database/Receipt):
  - Student Info Retrieval from DB
  - Functionality added to email receipts

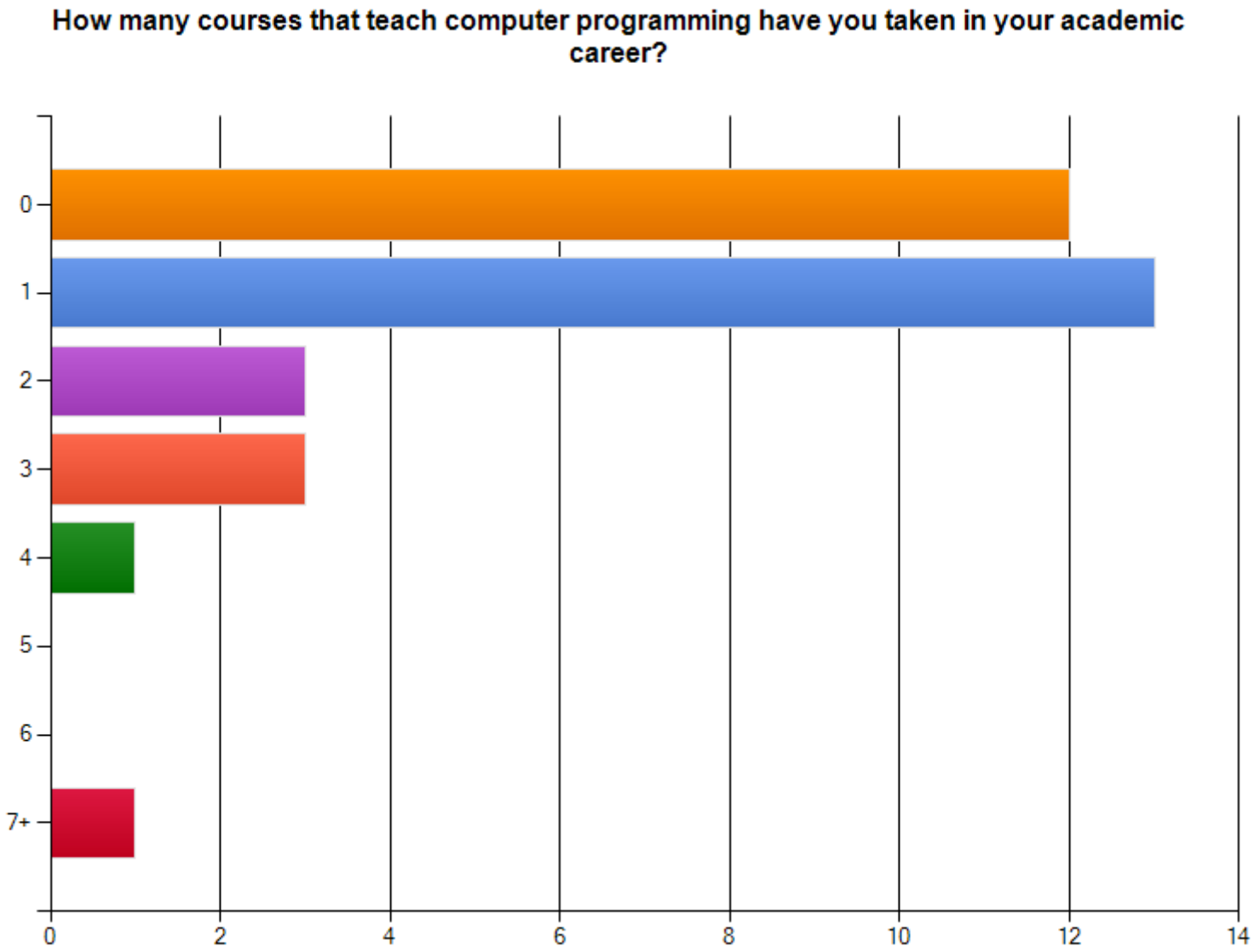
Results

- 46 started survey, 33 completed surveys
- Programming Experience
    - C and C++ most commonly known (3 each)
- 17 seniors, 14 juniors, 2 sophmore
- Most common majors were biology(4) and Business (5)

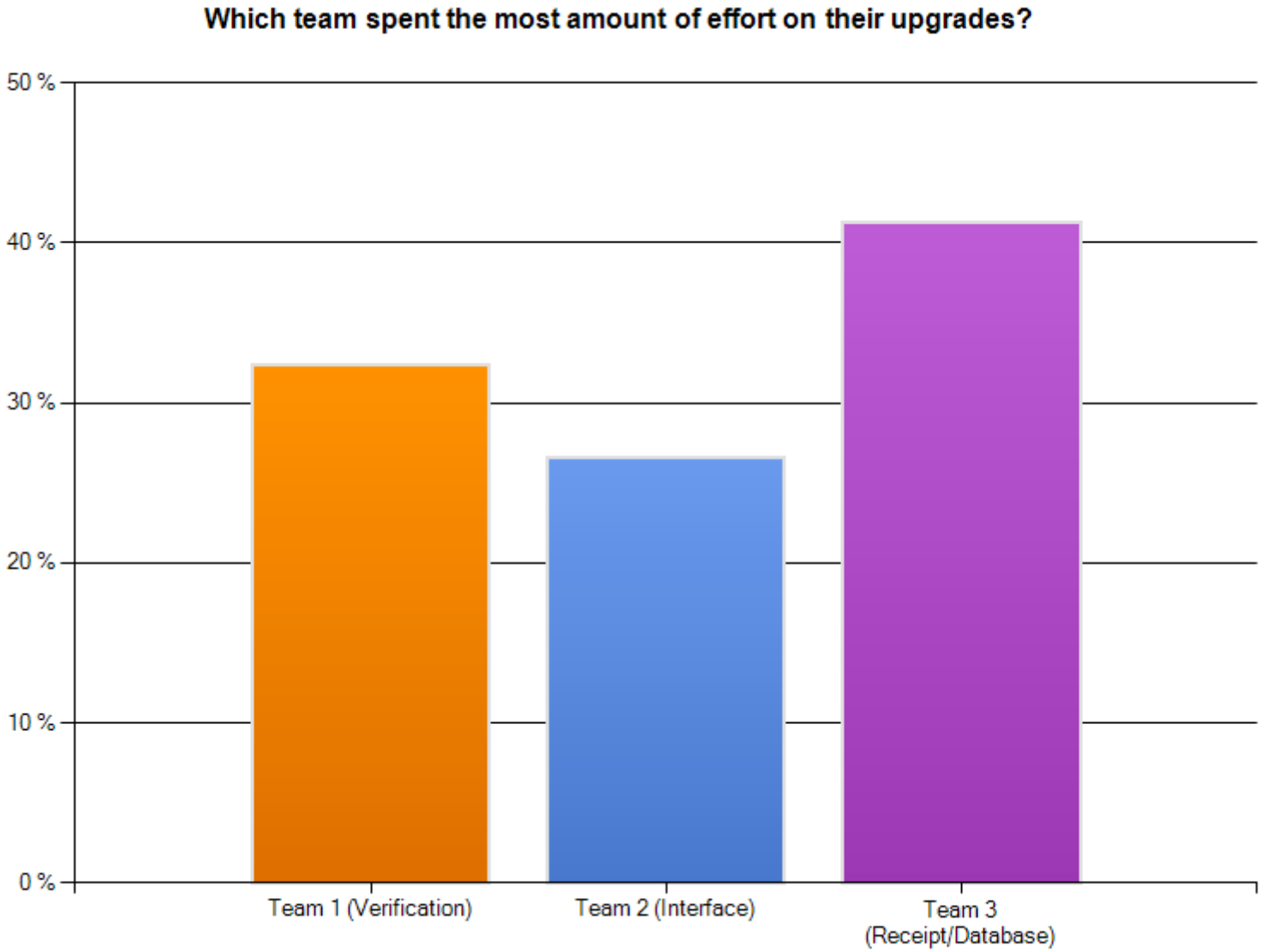
Results



Results

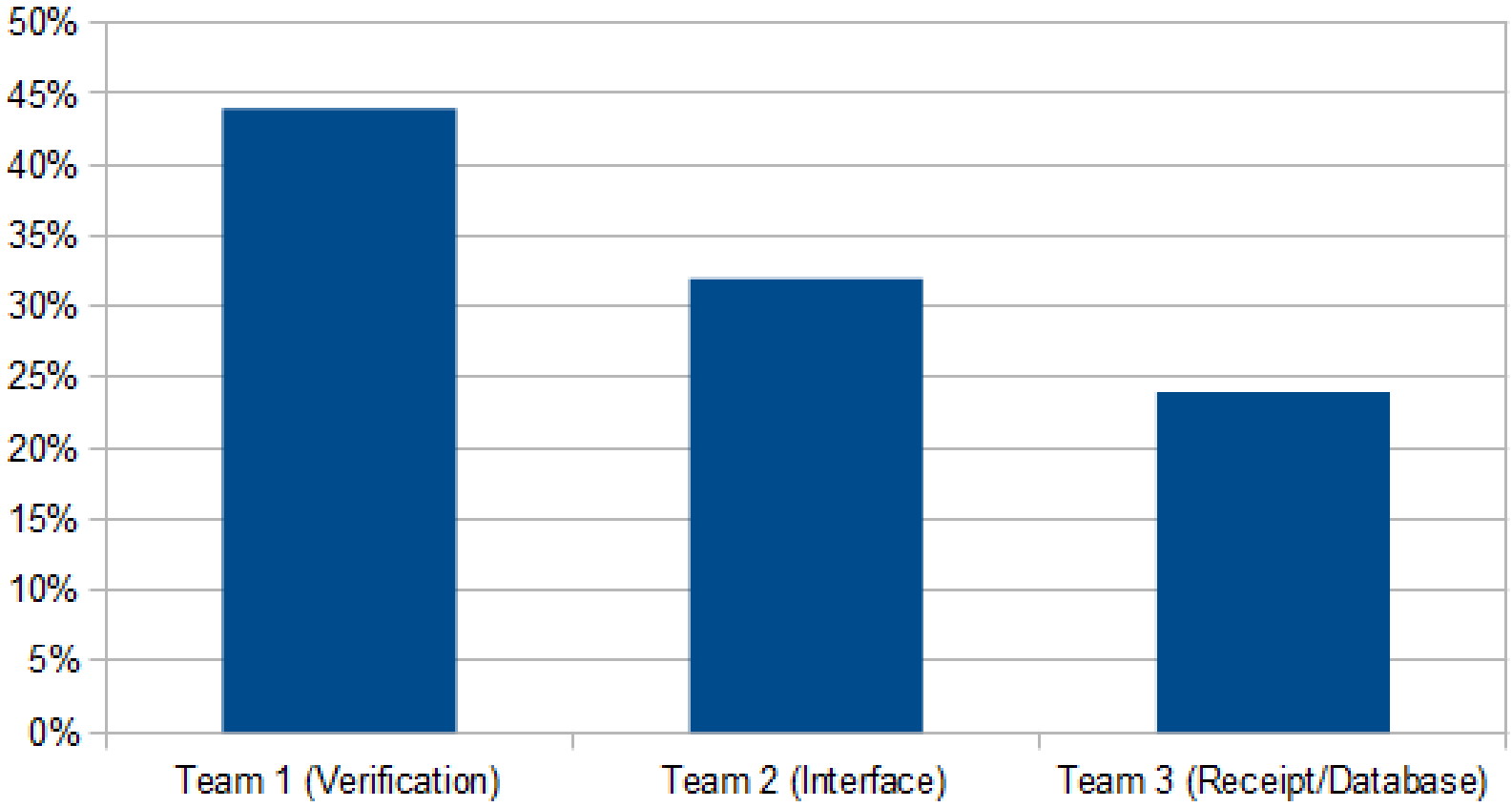

How many courses that teach computer programming have you taken in your academic career?
0
1
2
3
4
5
6
7+
0
2
4
6
8
10
12
14

Results

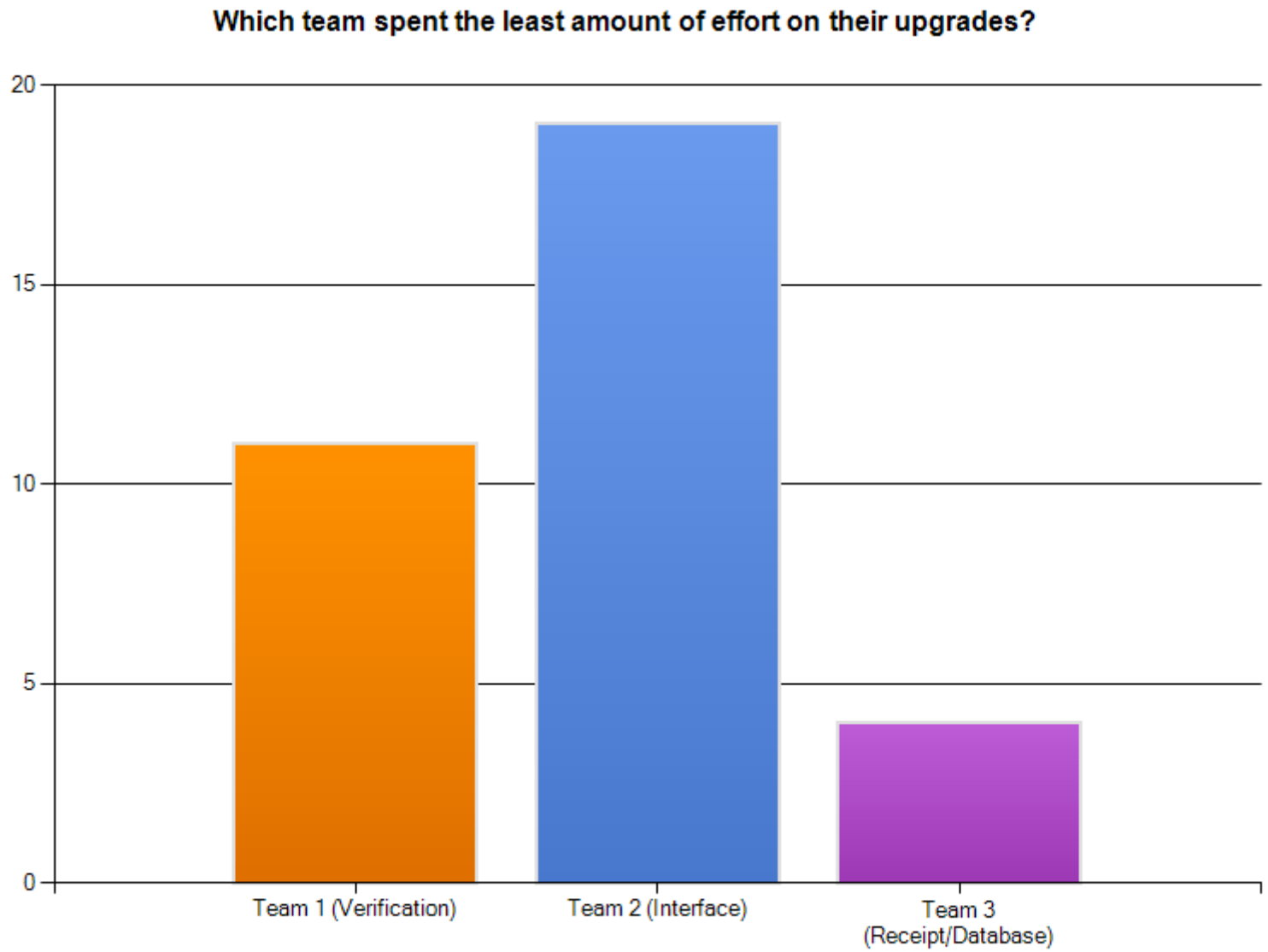

Which team spent the most amount of effort on their upgrades?
50 %
40 %
30 %
20 %
10 %
0 %
Team 1 (Verification)
Team 2 (Interface)
Team 3 (Receipt/Database)

Results

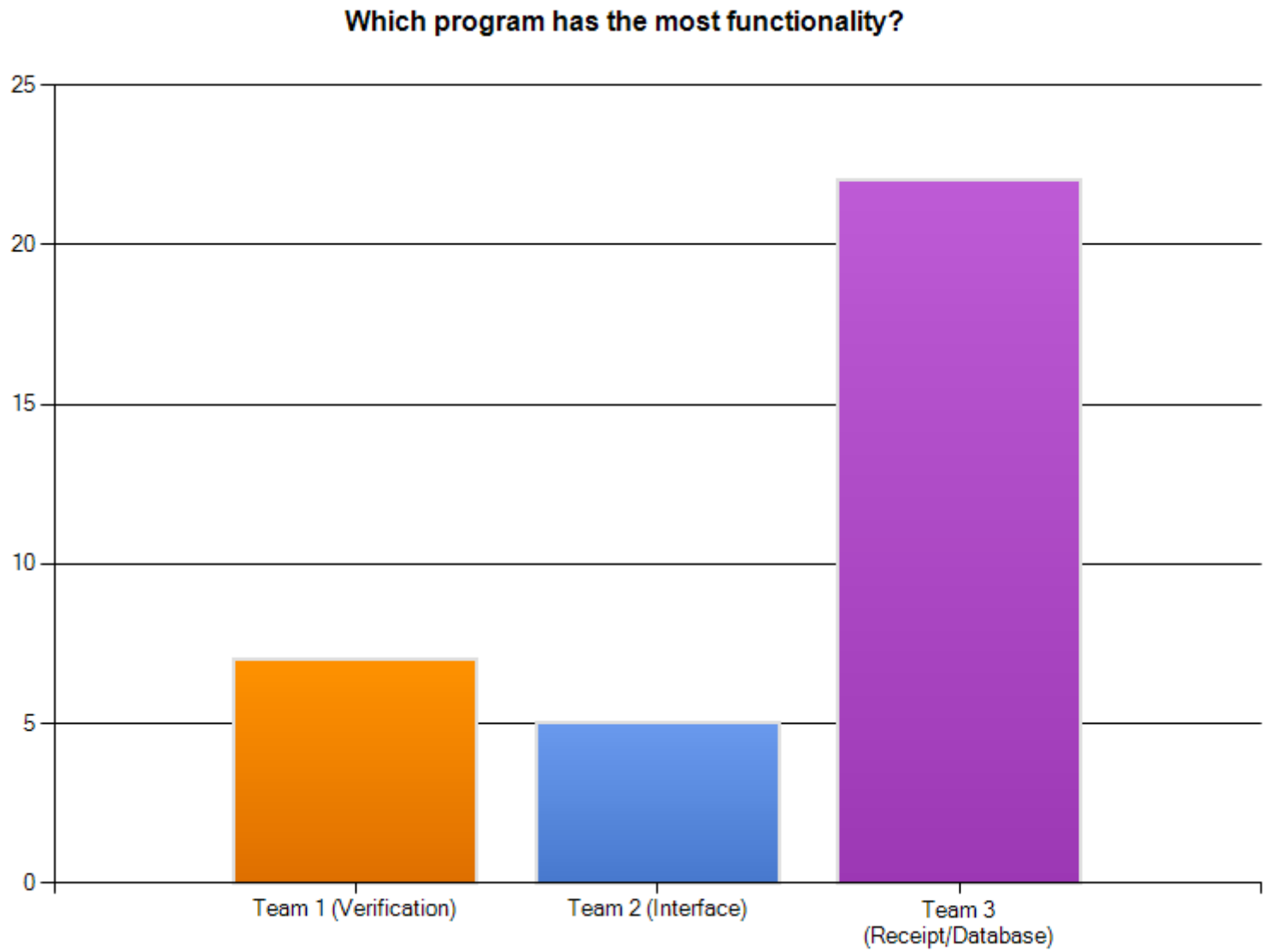
Which team spent the most amount of effort on their upgrades?
(responses omitted for those who have taken more than 1 programming course)



Results


Which team spent the least amount of effort on their upgrades?
20
15
10
5
0
Team 1 (Verification)
Team 2 (Interface)
Team 3
(Receipt/Database)

Results


Which program has the most functionality?
25
20
15
10
5
0
Team 1 (Verification)
Team 2 (Interface)
Team 3
(Receipt/Database)

Results (cont.)

- Which team spent the most effort on their improvements?
    - Team 1 (Verification): 11
    - Team 2 (UI): 8
    - Team 3 (Database/Email): 14
- Which team spent the least amount of effort?
    - Team 1 (Verification): 11
    - Team 2 (UI): 19
    - Team 3 (Database/Email): 3

Results (cont.)

- Which program has the most functionality?
  - Team 1 (Verification): 6
  - Team 2 (UI): 5
  - Team 3 (Database/Email): 22

To-do

- Compare responses against computing majors in equal class
- Statistical analysis once CSC dataset is in
- Write a small paper about it
- MORE RESEARCH!

Future Work

- Expand analysis outside of university
  - Compare responses of expert software developers versus their customers

References

- Scaffidi, C.; Ko, A.; Myers, B.; Shaw, M.;, "Dimensions Characterizing Programming Feature Usage by Information Workers," *Visual Languages and Human-Centric Computing, 2006. VL/HCC 2006. IEEE Symposium on* , vol., no., pp.59-64, 4-8 Sept. 2006.
- N. Bevan and M. Azuma. 1997. Quality in Use: Incorporating Human Factors into the Software Engineering Lifecycle. In *Proceedings of the 3rd International Software Engineering Standards Symposium (ISESS '97)* (ISESS '97). IEEE Computer Society, Washington, DC, USA, 169-.
- Vennila Ramalingam and Susan Wiedenbeck. 1997. An empirical study of novice program comprehension in the imperative and object-oriented styles. In *Papers presented at the seventh workshop on Empirical studies of programmers* (ESP '97), Susan Wiedenbeck and Jean Scholtz (Eds.). ACM, New York, NY, USA, 124-139.
- B.T. Pioro. 2006. Introductory computer programming: gender, major, discrete mathematics, and calculus. J. Comput. Small Coll. 21, 5 (May 2006), 123-129.
- Maiden, N.A.M.; Sutcliffe, A.G.; , "People-oriented software reuse: the very thought," *Software Reusability, 1993. Proceedings Advances in Software Reuse., Selected Papers from the Second International Workshop on* , vol., no., pp.176-185, 24-26 Mar 1993.
- Michael McCracken, Vicki Almstrum, Danny Diaz, Mark Guzdial, Dianne Hagan, Yifat Ben-David Kolikant, Cary Laxer, Lynda Thomas, Ian Utting, and Tadeusz Wilusz. 2001. A multi-national, multi-institutional study of assessment of programming skills of first-year CS students. In *Working group reports from ITiCSE on Innovation and technology in computer science education* (ITiCSE-WGR '01). ACM, New York, NY, USA, 125-180.
- Urban-Lurain, M., Weinshank D.J., Do non-computer science students need to program?, Journal of Engineering Education, 90 (4), 535-541, 2001.
- Nan Zang; Rosson, M.B.;, “What’s in a mashup? And why? Studying the perceptions of web-active end users,” Visual Languages and Human-Centric Computing, 2008. VL/HCC 2008. IEEE Symposium, pp.31-38, 15-19 Sept. 2008.