5740 Toronto Rd
V6T1L2, Vancouver, BC

November 26th, 2021

Leonard Wang
the president of UBC Computer Science Student Society (CSSS)
ICICS Room 021
2366 Main Mall
Vancouver, BC V6T 1Z4

Dear Mr. Wang,

Here is my report, Determining the Feasibility of Providing New Workshops. In preparing this report, I have conducted a survey and reviewed many works of literature in addressing similar problems to what action is needed to help students acquire coding skills to meet the industry needs and improve the UBC CS program.

The UBC CS program is one of the top programs in computer science education globally. UBC provides academic concepts and principles in the traditional lectures to train students with a qualified understanding of CS and computing thinking. However, industry needs somehow are different from the academic goals set by UBC. Without enough practice in using specific programming languages, new graduates usually feel pressure at the beginning of their careers. This report analyzes the data collected from both previous and current UBC CS students and provides a solution to the mismatch between academic goals and industry needs.

I am glad to have an opportunity to work on this project, and I appreciate all your support with this report. If you have any questions regarding this report, please email me at youngllll2015@gmail.com.

Sincerely,

Yang Liu

Yang Liu

Determining the Feasibility of Providing New Workshops

For

Leonard Wang

President of CSSS

By

Yang Liu

ENGL 301

November 26, 2021

Table of Contents

List of Figures:

Abstract

UBC CS department is one of the top departments in computer science education globally. UBC helps students in computer science acquire computational thinking and problem-solving skills by delivering professional computer science courses like many other top universities. However, the academic targets do not meet the industrial requirements in many ways. Computer science courses in UBC focus on understanding conceptual ideas and principles, while industries require designing and writing a program in any specific programming language.

This research aims to help solve the mismatch between academic targets and industry needs by providing workshops to help students practice their coding skills. The CSSS, a club for all UBC CS students, is expected to take responsibility for the workshops.

Two surveys are used as data sources to help determine the feasibility of the new workshops -- survey for previous UBC CS students and survey for current UBC CS students. Based on the analysis, we can conclude that such new workshops for students are essential. Below are some suggestions for workshop design:

1. A project-based workshop is strongly recommended.
2. Collaboration should be encouraged when working on a project.

3. The workshop should help establish connections between students and local industries.
4. A Technical Lab is preferred to a conceptual lecture.

Introduction

Background:

Computer science is an essential part of society and benefits our lives in many ways. Even though people do not know computer technologies in detail, they connect themselves with computer technologies by using maps on their mobile phones, watching online videos on Netflix and many other ways. Modern businesses also rely on computer science technologies. Business people communicate with emails and transfer using online banking systems. Most companies generate and store any taxing information like bank statements or cash flow tables in the computers instead of manually writing any information down and keeping it in the file storage. Computer science education is more important than ever, and demand for computer science graduates continues to outpace supply. Based on research done at the University of Washington (UW), the number of first-year students who applied to UW in computer science and engineering is about three times that of first-year students in other majors. A consistent result can be found in many other world-famous universities, such as Harvard, Stanford, and Michigan. (Soper)

Problem Statement:

UBC, as a top-ranking university globally, meets a similar demand of interest in computer science study. UBC helps students acquire computational thinking and problem-solving skills by delivering professional computer science courses. However, the academic targets do not meet the industrial requirements in many ways. Computer science courses in UBC focus on understanding conceptual ideas and principles, while industries require designing and writing a program in any specific programming

language. Students from universities are unskilled in using programming languages because university computer science courses emphasize academic principles instead of programming skills. Even though students can practice when they write their programming assignments or participate in co-op programs, their programming skills are not enough to meet the actual industry requirements. A new graduating student faces many challenges in seeking jobs, and technology companies must lower their employment requirements for incoming new graduating students. Many companies even design training sessions in specific programming languages for new graduating students after they get employed, increasing their cost.

- **Proposed Solution and Audiences:**

This report tries to seek a way to help UBC computer science students develop qualified coding skills to meet industrial needs. A possible solution to solve the mismatch between the academic targets in computer science educations and the industrial requirements for IT companies is to provide a workshop. In the workshop, students will get practice in coding skills. The CSSS, a formal club for all computer science students, is expected to take the responsibility to hold this workshop. Therefore, the intended audience for this report is Leonard Wang, the president of CSSS, the club for UBC computer science students.

- **Scope:**

Three areas of inquiries are pursued to determine the feasibility of this workshop.

1. What programming languages are commonly required in the industry and essential to teach in the workshop session?
2. How willing are students to attend the workshop, and how many hours will they spend?
3. How to quantify a student if he/she is well-trained in a specific programming language? What are the standard requirements for students to complete the workshops?

- **Data Source:**

Two surveys sources are used in this report to answer these questions. The participants for the two surveys are current UBC CS students and past UBC CS students, respectively. A general conclusion regarding this workshop's pros and cons will be given at the end of this report.

Data Session

- **Feedback from Previous Students:**

The finding is based on the survey assigned to 23 previous students.

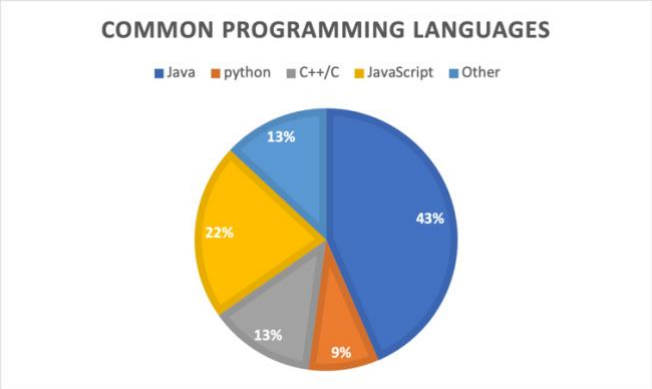

Figure 1. 1 The common programming language in use in industries

The first question investigates the most common languages in use in previous students' career life. As shown in Figure 1.1, over 40% of participants choose Java as the most common programming language in their career life. The second common language in use is JavaScript, where 22% of participants use JavaScript. Other languages like C/C++, python roughly divided the remaining participants. However, the distribution varies as the participants' job changes. All the participants who work as Frontend developers mentioned that JavaScript is the primary programming language in their life. For software testers and backend developers, Java is still the most common language.

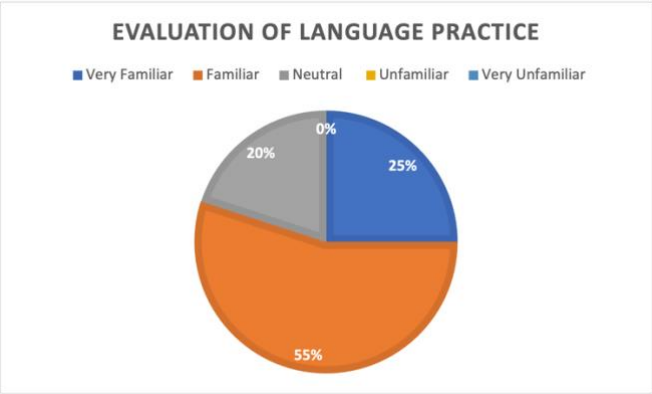

Figure 1. 2 How Familiar are they with these languages

Regarding how familiar they are with their most common language chosen in question 1, 35% of participants think they are "very familiar" with their commonly used languages in the language practice evaluation part. They can do almost all the work by themselves in their career life without seeking help from others. They know the efficiency of their code and continually optimize the efficiency. (Code efficiency refers to a good time complexity and space complexity in their code.) 48% of participants think they are "familiar" with their commonly used languages. Like those "very familiar" participants, they finish their tasks most of the time and sometimes with some help. They can guarantee the complexity in their code with limited exceptions. The remaining participants choose "neutral." and no participant chooses "unfamiliar." or "very unfamiliar" in the survey.

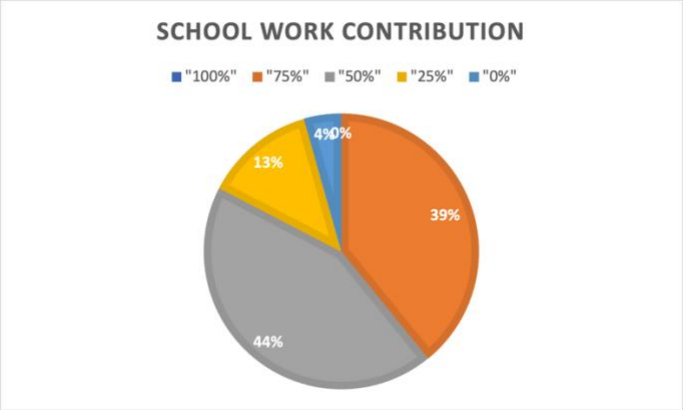

Figure 1. 3 School Work Contributes to Current Coding Skills

Concerning the percentage contribution of code ability from schoolwork, no participants think their code ability is fully contributed by schoolwork like academic projects or assignments. The number of participants who agree that most or half of their code skills benefit from schoolwork is close, with 39% and 44% percentages, respectively. 13% of participants agree that a minority of their code ability comes

from their schoolwork. 4% of participants think schoolwork contributes nothing to their coding skills related to their most common language in their current job.

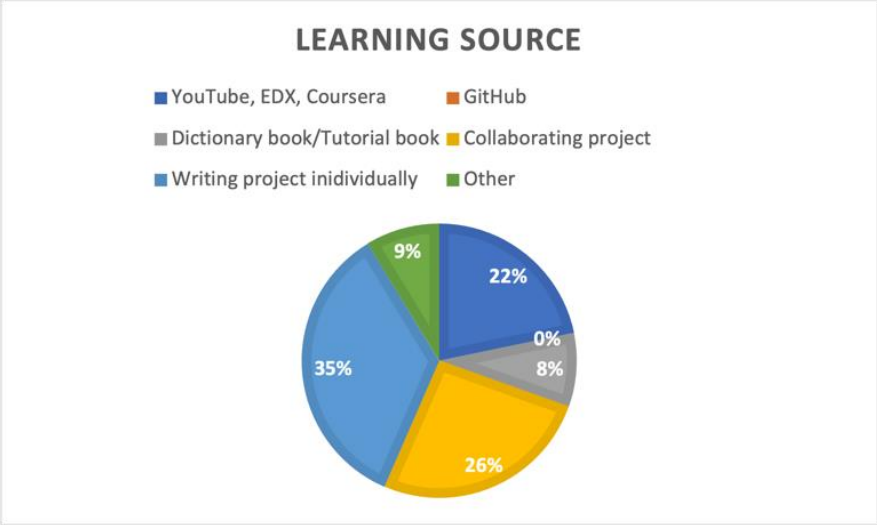

Figure 1. 4 Common Learning Sources

Figure 1.4 shows the learning source that previous students prefer to learn or practice their coding skills. The most common learning source is writing projects individually. Collaborating projects with colleagues and video tutorials from YouTube, Edx, and Coursera share similar percentages with 26% and 22%. Less than 10% of participants choose to improve their coding skills by reading code dictionary books.

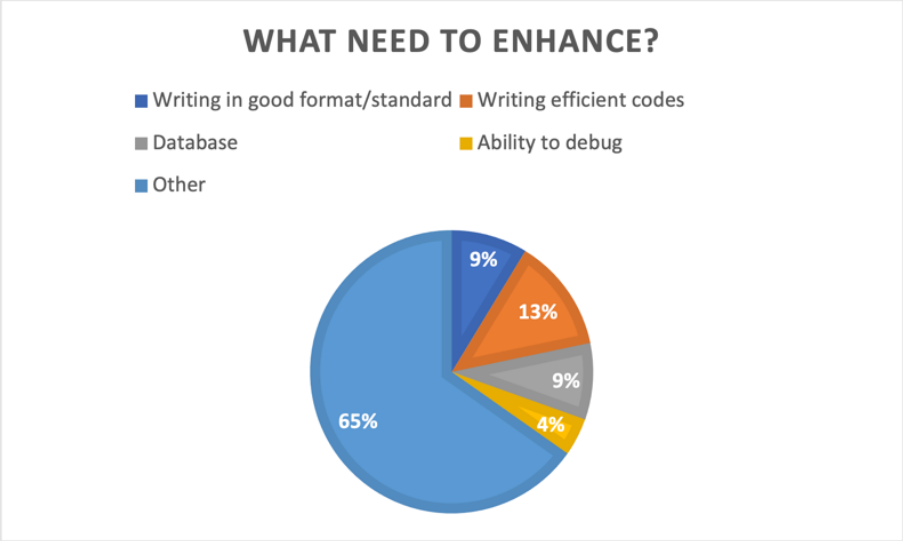

Figure 1. 5 What need to enhance before graduating

What coding ability should computer science students enhance most before graduating? As shown in Figure 1.5, most participants think something other than coding format, code efficiency, database, and debugging ability should be improved before leaving school. Besides, the ability to write efficient codes (i.e., good time and space complexity) is the most important to be considered as an IT jobseeker.

- **Feedbacks from UBC current students:**

The second data source is a survey for current UBC students in the computer science department. Thirty-six participants take this study anonymously.

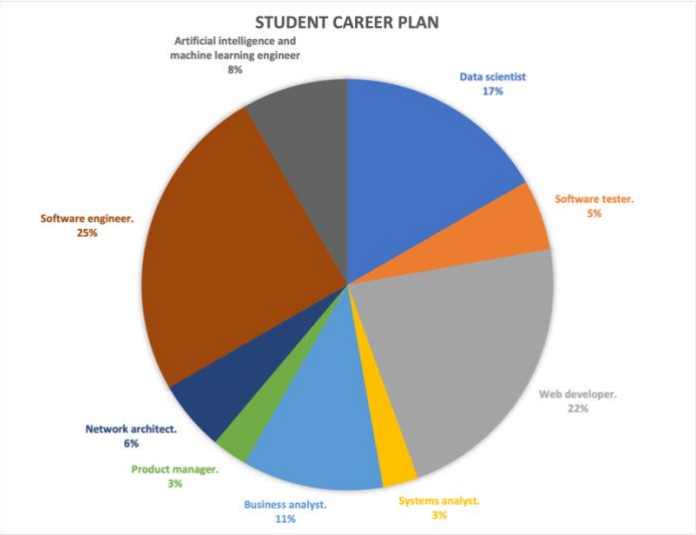

Figure 2. 1 Students' Career Plans

Students' career plan is shown in Figure 2.1. Based on students' career plans, web developers and software engineers are the most popular among all careers, with 25% and 22%, respectively. Jobs related to data analysis like data scientist, business analyst, and machine learning engineers are 17%, 11% and 8% of all. Students who plan to work as network architects, systems analysts, software testers and product managers roughly have the same percentage, around 4%. None of the students plan to do a non-technical job after they graduate from university.

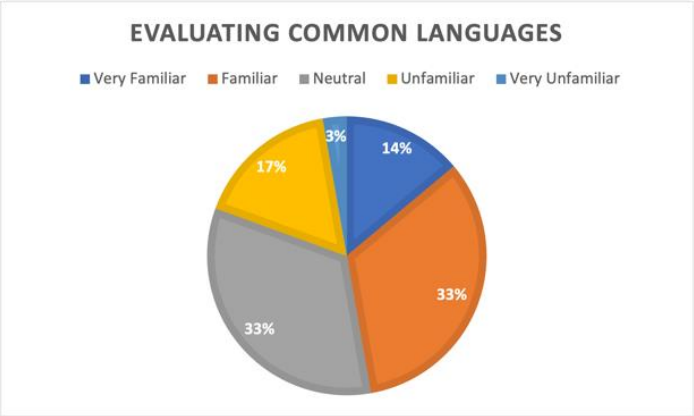

Figure 2. 2 How Familiar they are with their expected most common programming languages (in percentages)

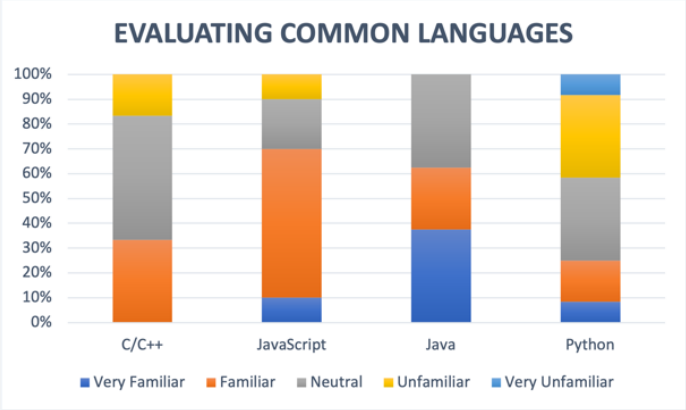

Figure 2. 3 How Familiar they are with their expected most common programming languages (distribution for each language)

Figure 2.2 shows the results regarding the students' expectations about the most common programming languages in their future career life. The self-evaluation about their coding skills in their chosen languages is shown in Figure 2.3. Students who think Python will be the most common language in their future life occupies 33% of all. However, only less than 10% of these students believe they can perfectly handle Python when programming. The majority of these students are unconfident in programming using Python, and they either only know the basic structure and syntax of Python or part of them. The coding efficiency is not guaranteed, and time/space complexity needs to be optimized by others. Some

students even think they know nothing about Python. The second common language in expectation is JavaScript, and students are generally familiar with JavaScript. 60% of students who plan to use JavaScript in the future can write projects individually with some help from others. C family languages (C/C++) and Java have similar percentages, around 20%, but students' coding skills in Java are generally better than their coding skills in C family languages. Figure 3 shows that 60% of students feel either professional or proficient in using Java. None of the students think they are unfamiliar with Java. In other words, they all know the basic syntax and common-used data structure in Java. Regarding future C family language users, 70 % of students think they are neutral or unfamiliar with C family languages. Only about 30 % of students feel confident in using C/C++.

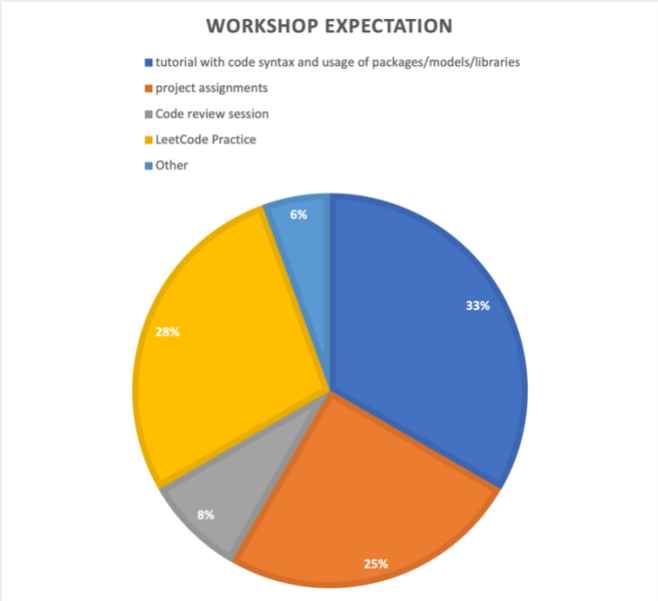

Figure 2. 4 Students' Expectations from Workshop

Figure 2. 5 Time that Students are willing to offer to workshop

Regarding students' expectation from workshop, many students expect to have tutorials about the code syntax and some packages/libraries for specific programming languages if CSSS decides to hold a workshop for students, which takes 33% of all. Besides, assigning projects to students and providing essential help in finishing the project is highly expected from the workshop with roughly 25%. 28% of students hope the workshop can guide students in solving the problems in LeetCode. LeetCode is a commonly used website for companies, where many algorithm design problems are selected to examine students' coding abilities when interviewing. Other parts can be found in Figure 2.4. The time that students are willing to spend in the workshop varies from 2 hours to 16 hours, and most of the students prefer to spend 4 - 6 hours in the workshop each week. 6 - 8 hours and less than 4 hours are also common expectations among students.

Conclusion

- **The Essentiality of the Workshop**

Academic scholars have discussed computer science educations in many ways. Computer science educations based on traditional lecture format teaching have been widely criticized. The conventional teaching method in computer science somehow leads to "inert" knowledge, and students fail to associate their in-school learning with the out-school context. New teaching approaches like project-based, problem-based and constructive learning are adapted to advance computer science education. These new approaches highlight the importance of practices and collaboration in computer science education (Schilling and Klamma 377). Moreover, the conflict between the academic targets in university and actual industrial needs is also an issue to be considered in computer science education. The industries prefer the universities to train students in the most recent technologies, while the universities' concerns are always the long-term educations. Even though some universities make some effort to meet the industrial need by providing project courses or internships, these efforts usually begin late and are not enough (Tvedt et al. 102).

UBC also faces the same challenges as all other top universities. Based on the data collected from previous UBC CS students, 44% of the students agree that schoolwork only contributes to half of their coding skills now. Based on the data collected from current UBC CS students, 50% of students feel neutral or even unfamiliar with the potential programming language in their future jobs. In addition, many courses in UBC are traditional lecture format and emphasize teaching knowledge. It is necessary to have a workshop that provides opportunities for students to associate their knowledge with out-school context, practice their coding and collaborating abilities, and know what their future employers expect

from them.

The Design to Workshop

The CSSS should take responsibility for holding this workshop because CSSS is a club designed for only students in the computer science department. Some suggestions are listed below for workshop design. These suggestions combine insights from previous studies and the findings from two source data.

1. A project-based workshop is strongly recommended.

In recent years, constructivist learning has been introduced to many university courses as case studies to advance computer science education. Constructivism suggests that allowing learners to construct knowledge by themselves is an efficient way to learn. Constructivist learning emphasizes an active learning process instead of passively transmitting knowledge from instructors. A common implication of constructivist learning in computer science education is providing a project-oriented course (Hadjerrouit 102). Moreover, our data also suggest that writing project is one of the most common ways for students to practice coding skills. Over 60% of past UBC CS students state that writing projects individually or in team format helps improve their coding abilities and is the major learning source.

2. Collaboration should be encouraged when working on a project.

A collaboration involving social interaction and negotiation is another crucial aspect of constructivist learning. The collaboration includes group formation to finish the project, communication between teams, and between instructors and students. Instead of providing students with solutions, instructors should take a role to guide them in searching relevant (helpful) information, evaluating their projects and providing feedback to improve (Hadjerrouit 102). The collaboration benefits students in sharing the knowledge and thereby actively learning. (Schilling and Klamma 377) Some consistent conclusions can be found in our data as well. 26% of previous UBC CS students in the survey suggest that collaborating projects with classmates best help them train their coding skills.

3. The workshop should help establish connections between students and local industries.

Connections between students and local industries during the workshop help students understand the needs of industries. The workshop can invite local industries to involve in workshop programs by assigning the project to students, and the project can be part of tasks to be finished in the companies. This connection intends to benefit both local industries and students. Students also learn necessary industrial skills and reflect upon their own experiences. The industry needs, including exposing students to new technologies, teamwork, large-scale development, and management activities, are addressed by connecting with students. (Tvedt et al. 102) In addition, any new concepts and methods in computer science education are also transferred from universities into company practices (Schilling and Klamma 377). In work done by Ellis et al. (120), the characteristics of successful collaborations between

universities and companies are discussed. The common characteristics of companies include a solid initiative to seek professional training from university, a relatively strong background (enough finance and big size).

4. A Technical Lab is preferred to a conceptual lecture.

As a top-ranking university, UBC has provided enough academic concepts and principles for its students in the CS department. This workshop intends to enhance students' coding skills and other industrial skills to help students seek jobs. Therefore, a technical lab regarding any techniques and software engineering methods is preferred to a regular lecture. Based on the survey from current UBC students, students basically know what common programming languages are but have a bad performance in some of them. For example, students are generally unfamiliar with python and C family languages. The workshop can assign projects in these programming languages to practice their coding skills. The technical lab can provide technique teaching based on the project objectives and the project plan.

- **Limitations: Heavy projects may discourage students' voluntariness.**

According to work done by Schilling and Klamma (377), most of the students attend the project-based course because the course is a mandatory course for their studies. Few students take such courses due to the project character of the courses. Even though a question regarding willingness is in the survey, a heavy project-based workshop is expected to discourage students' voluntariness. A heavy project-based

workshop takes time, and students may feel pressure to balance the course work and workshop content. Based on the survey assigned for the current UBC CS students, most of the students prefer 4 - 6 hours each week spent in the workshop, which is not enough to complete a heavy project. The result could be that students either refuse to take the workshop or fail to complete the project at the end of the term. Also, the unstable supply of projects at the end of the term may cause the workshop to be untrustworthy to workshop collaborators and result in further problems.

Reference

Ellis, Heidi J. C., et al. “Characteristics of Successful Collaborations to Produce Educated Software. Engineering Professionals.” Computer Science Education, vol. 12, no. 1–2, 2002, pp. 119–40. Crossref, doi:10.1076/csed.12.1.119.8217.

Hadjerrouit, S. “Learner-Centered Web-Based Instruction in Software Engineering.” IEEE Transactions on Education, vol. 48, no. 1, 2005, pp. 99–104. Crossref, doi:10.1109/te.2004.832871.

Schilling, Jan, and Ralf Klamma. “The Difficult Bridge between University and Industry: A Case Study. in Computer Science Teaching.” Assessment & Evaluation in Higher Education, vol. 35, no. 4, 2010, pp. 367–80. Crossref, doi:10.1080/02602930902795893.

Soper, Taylor. “Analysis: The Exploding Demand for Computer Science Education, and Why America Needs to Keep Up.” GeekWire, 21 Feb. 2015, www.geekwire.com/2014/analysis-examining-computer-science-education-explosion.

Tvedt, John D., et al. “The Software Factory: An Undergraduate Computer Science Curriculum.” Computer Science Education, vol. 12, no. 1–2, 2002, pp. 91–117. Crossref, doi:10.1076/csed.12.1.91.8213.

Appendix A – Survey for Previous Students in CS

I am an undergraduate student at UBC engaged in a technical writing project. **The purpose of this survey is** to obtain primary data for an analysis and **investigation that aims to provide recommendations for improving coding ability of students in UBC computer science program. The final formal report will be addressed to** CSSS club members and UBC CPSC students. Together with the reports available from ENGL 301 course website, the data I gather from **this survey will serve the ultimate purpose of providing recommendations for** increasing efficiency and accessibility**. The survey contains 7 multiple-choice questions, and it should take about 5 minutes of your time. Your responses are voluntary and anonymous. Thank you, I appreciate your generous participation in my survey.**

What is your current job?

A. Data scientist.
B. Software tester.
C. Web developer.
D. Systems analyst.
E. Business analyst.
F. Product manager.
G. Network architect.
H. Software engineer.
I. Artificial intelligence and machine learning engineer
J. Other programming-based job: please write your job name down
K. Non-technical job: please write your job name down

What is the common-used/related programming language in your job?

A. C
B. C++
C. C#
D. Python
E. Java
F. JavaScript
G. GO
H. R

I. PHP
J. Swift
K. Ruby
L. MatLab
M. Other Programming language: please write the language name down
N. N/A

To what extent do you think you are familiar with these programming languages?

A. Very familiar, can use these language programming without help/with few helps from others (Internet, programming book, etc.), know how to programming efficiently and avoid codes with a long runtime.
B. Familiar, but somehow unfamiliar with some advanced packages/libraries. Some helps from others (Internet, programming book, etc.) are necessary when programming. Codes are generally efficient, but still write some codes with a long runtime and do not know how to avoid them.
C. Neutral. Know basic (basic data structure, basic syntax, etc.) about these languages, and can finish some easy functions/programs by yourself. Can use some advanced packages (like socket, database) but need check guideline/codebook when writing. Code is bug-free but efficiency is undetermined, do not know how to avoid a long runtime.
D. Unfamiliar. Know some basic about these languages, can finish some easy functions/programs by yourself. Cannot use any advanced packages.
E. Very unfamiliar. Do not know the basic about these languages, cannot finish easy functions/programs by yourself.
F. N/A

To what extent do you think your experience in UBC computer science program contribute to your coding ability in these languages?

A. 100%. Get significantly enough practices in coding. Can handle every coding task in your current job as soon as graduating from UBC. Spend less than 3 hours in learning programming languages every week when got your first job.
B. 75%. Get many practices in coding. Help learn most of the packages/libraries in writing assignments, exams, capstones. Can handle most of coding tasks in your current job as soon as graduating from UBC. Spend 3-6 hours in learning programming languages every week when got your first job.
C. 50%. Get some practices in coding. Help learn all basic packages/libraries in writing assignments, exams, capstones. Can handle some coding tasks in your current job as soon as

graduating from UBC. Need to spend about 6-12 hours in learning programming languages when got your first job.

D. 25%. Get a few practices in coding. Help learn part of basic packages/libraries in writing assignments, exams, capstones. Can handle limited coding tasks in your current job as soon as graduating from UBC. Need to spend about 12-18 hours in learning programming languages when got your first job
E. 0%. Exams, assignments, capstones help nothing in coding ability for these programming languages. Learn by yourself after graduating.
F. N/A

What contribute most to improve your coding except school-works?

A. Tutorial videos from YouTube, EDX, Coursera
B. Learning from others' public work in GitHub
C. Dictionary book/Tutorial book in teaching programming languages
D. Collaborating one project with friends
E. Writing personal project
F. Hiring Personal tutor
G. N/A
H. Others: please write down

What coding ability should computer science student enhance most before graduating

A. Writing in good format/standard
B. Writing efficient codes without long runtimes
C. Common functionality in using database
D. Ability to debug
E. Other: please write down

Name one thing about coding ability that student can only learn in workplace: (write down your answer)

Appendix B – Survey for Current Students in CS

I am an undergraduate student at UBC engaged in a technical writing project. **The purpose of this survey is** to obtain primary data for an analysis and **investigation that aims to provide recommendations for improving coding ability of students in UBC computer science program. The final formal report will be addressed to** CSSS club members and UBC CPSC students. Together with the reports available from ENGL 301 course website, the data I gather from **this survey will serve the ultimate purpose of providing recommendations for** increasing efficiency and accessibility**. The survey contains 6 multiple-choice questions, and it should take about 5 minutes of your time. Your responses are voluntary and anonymous. Thank you, I appreciate your generous participation in my survey.**

Which of the following job titles best describe your favorite future job?

L. Data scientist.
M. Software tester.
N. Web developer.
O. Systems analyst.
P. Business analyst.
Q. Product manager.
R. Network architect.
S. Software engineer.
T. Artificial intelligence and machine learning engineer
U. Other programming-based job: please write your job name down
V. Non-technical job: please write your job name down

What do you think would be the common-used/related programming language in your future job?

O. C
P. C++
Q. C#
R. Python
S. Java
T. JavaScript
U. GO
V. R

W. PHP
X. Swift
Y. Ruby
Z. MatLab
AA. Other Programming language: please write the language name down
BB. N/A

To what extent do you think you are familiar with these programming languages?

G. Very familiar, can use these language programming without help/with few helps from others (Internet, programming book, etc.), know how to programming efficiently and avoid codes with a long runtime.
H. Familiar, but somehow unfamiliar with some advanced packages/libraries. Some helps from others (Internet, programming book, etc.) are necessary when programming. Codes are generally efficient, but still write some codes with a long runtime and do not know how to avoid them.
I. Neutral. Know basic (basic data structure, basic syntax, etc.) about these languages, and can finish some easy functions/programs by yourself. Can use some advanced packages (like socket, database) but need check guideline/codebook when writing. Code is bug-free but efficiency is undetermined, do not know how to avoid a long runtime.
J. Unfamiliar. Know some basic about these languages, can finish some easy functions/programs by yourself. Cannot use any advanced packages.
K. Very unfamiliar. Do not know the basic about these languages, cannot finish easy functions/programs by yourself.
L. N/A

What contribute most to improve your coding except school-works?

I. Tutorial videos from YouTube, EDX, Coursera
J. Learning from others' public work in GitHub
K. Dictionary book/Tutorial book in teaching programming languages
L. Collaborating one project with friends
M. Writing personal project
N. Hiring Personal tutor
O. N/A
P. Others: please write down

If CSSS decides to hold a workshop for all CS students to improves their coding ability, what do

you expect most from this workshop?

A. Providing tutorial with code syntax and usage of packages/models/libraries
B. Assigning project to students and provides help in finishing the project
C. Introducing functionalities in existing project and telling students how the authors achieved these functionalities
D. Problems and solutions workshop. Providing problems from LeetCode and teach students how to solve the problems
E. Other, please write down

If CSSS decides to hold a workshop for all CS students to improves their coding ability, how many hours do you want to attribute to this workshop each week? (In Integer hours)