Evaluating Face-to-Face vs.Virtual Pedagogical Coding Review Sessions in the CS classroom: An HBCU Case Study*

Edward Dillon*, Briana Williams, Ayomide Ajayi, Zipporah Bright, Quinlan Kimble-Brown
Chauncey Rogers, Myles Lewis, Joseph Esema, Ben Clinkscale
Computer Science Department
Morgan State University
Baltimore, USA
{edward.dillon*, briwil95, ayaja3, zibri1, qukim1, chrog8, mylew3, joese1, becli1}@morgan.edu

Krystal L. Williams
Educational Leadership, Policy, Technology Studies Department
The University of Alabama
Tuscaloosa, USA
krystal.l.williams@ua.edu

***Abstract*— Teaching incoming Computer Science (CS) majors how to program has been an important research topic for some time. Programming is an essential skill that CS majors are expected to develop. Literature show that there is a consistent percentage of CS majors who become stuck or discouraged while learning to program, which sometimes deter them from the major altogether. When it comes to underrepresented groups like Black/African-Americans, the ability to retain such groups in CS is critical.**

Coding reviews are one potential practice that could assist students in their learning process with programming. This practice is common in industry settings and is used by professionals to solve major computational problems. Such practices could be transferred into the CS classroom to help students strengthen their skills for identifying both syntax and semantic flaws and related code defects of developed solutions. Literature also show that modest attention has been placed on the exploration of pedagogical coding reviews (PCRs) in HBCU settings of a CS course curriculum.

A study was conducted to evaluate two distinct semesters of PCRs either face-to-face (Fall 2019) or virtual-based (Fall 2020) at a Mid-Atlantic HBCU in the United States. The students involved during these sessions were either enrolled in a CS2 or an Object Oriented Programming course. The results revealed that these students collectively showed a slight increase in their ability to complete a relatively similar PCR task more effectively after initial exposure. Furthermore, no factors were found to provide a sufficient advantage or disadvantage during either semester setting.

Keywords*— *CS Majors, Pedagogical Coding Reviews (PCRs), Black/African-American students

I. Introduction

When it comes to programming pedagogy and success in the Computer Science (CS) classroom, there has been much discussion regarding ideal approaches and practices for aiding such outcomes [1, 3, 5, 7, 9, 11, 19]. When emphasizing success, one aspect of this is predicated on how well CS majors can perform as they matriculate through a course curriculum to obtain their degrees. Another aspect of success is defined by how these new and prospective CS degree awardees are able to effectively showcase their learned skills as practitioners in real-world and practical settings.

It can be argued that there exists a trend for the demand of jobs in the field of Computing to exceed the supply of candidates that can fill them. One notable reason for this trend is in correlation to the CS degree producing pipeline where there exists a high attrition rate of students leaving the major as they matriculate through a CS curriculum [4]. Another reason can be seen during the latter end of this same pipeline where CS majors, who are on the job market, are unable to showcase the appropriate skill-sets and related proficiencies needed as candidates for computational job opportunities. Both reasons are predicated on a student's deficiency for exhibiting a proficiency for computational problem solving [8, 24]. When emphasizing the latter reason, it is important to examine how CS and related departments are preparing their prospective graduates to meet such expectations imposed by companies, organizations, and related employers in professional and practical settings in the field.

One practice that has been adopted in the classroom to address problem solving deficiency is *coding reviews (or pedagogical coding reviews – PCRs)*. This practice exposes students to written code/solutions to a particular problem where they are required to analyze such content on the basis of *code correctness, code readability* and *code behavior/functionality*. Moreover, this practice allows students to understand and interpret the code syntax, paradigms, concepts, and data structures employed to generate a computational solution to a given problem. The practice of coding reviews are also common in industry settings. In such settings, project teams use this practice to communicate ideas, enhance developed computational solutions, and promote group-based communication and continuity.

Pertaining to minority serving institutions, there have been direct initiatives between companies in industry and historically black colleges and universities (HBCUs) to improve the representation of HBCU graduates in CS as practitioners in industry and related sectors. As part of this growing initiative, employing PCR practices in the minority-serving CS classroom could further assist in aiding these particular groups of students in their computational skills and programming proficiency. This can also further aid in the initiative of increasing HBCU CS graduate representation in industry and related professional settings.

This article looks to address the impact of PCR employment in an HBCU CS classroom setting by discussing an empirical study comprised of *PRE* and *POST* coding review assessments that were conducted during the *Fall 2019* and *Fall 2020* semesters, respectively, at an HBCU located in the Mid-Atlantic region of the United States. The targeted audience for these assessments were primarily comprised of Black/AA CS majors enrolled in either a CS2 or Object-Oriented Programming (OOP) course in the CS department at this particular HBCU.

Moreover, the COVID-19 pandemic has allowed virtual teaching to become an alternative norm in many academic classroom settings. This dynamic imposes an additional challenge for effectively teaching students in the classroom. Due to the timing for when these PCRs assessments were conducted, this study will also address any notable similarities and differences for exposing early CS majors to PCRs while being either face-to-face (*Fall 2019*) or virtual (*Fall 2020*).

II. Literature Review

Literature has shown a substantial amount of attention devoted to efforts that emphasize the impact of coding review practices in a variety of computational settings. Two notable emphasis have being placed on *modern coding reviews* [2, 10, 23] and *pedagogical coding reviews (or PCRs)*. When emphasizing PCRs, a subset of the literature has examined the ability for coding reviews to improve the programming skills of CS and related majors both in early and later stages of a course curriculum [14-15, 17-18, 20]. Hundhansen, Agrawal, & Agrawal's work, in particular, revealed that conducting PCR exercises in early CS courses can enhance a student's ability to establish coding analysis skills, while also developing soft skills such as positive attitudes toward programming, teamwork, and communication [13]. Furthermore, a specific technique such as continuous inspection has been noted to help students identify poor coding habits and tendencies, as they learn to adopt code efficiency and quality practices during their coding experiences [17]. Pertaining to the specific intent of this article, Hundhasen et al.'s work[14] consists of a similar study involving *virtual* vs. *face-to-face* learning using PCR at the early CS courses. From this study, it was found that students who were face-to-face tended to have more of an impactful experience with PCRs than those who were virtual. This was found to be true when measuring their *self-efficacy, peer learning practices, quality of an actual review,* and *general attitudes* [14]. One notable difference between Hundhasen et al.'s study and our personal study is the institutional settings for this work. Hundasen et al.'s work was conducted at an predominantly white institution (PWI), while our work was conducted at an HBCU that is comprised of predominantly Black/AA CS majors.

Literature involving systematic and empirical PCR efforts in HBCU classroom settings is minimal. Nevertheless, there have been initiatives by tech companies to address the engagement and recruitment of URMs in HBCU settings [16, 21]. Likewise, these companies have begun working closely with minority-serving institutions in an effort to provide insight on the type of computational skills and programming proficiency a student (or a future prospective employee must exhibit in order to be successful in these particular sectors [6, 12, 22]. One common insight from their observations concerns a candidate's ability to exhibit proficient critical thinking skills to solve problems through computational programming. Based on its noted benefits, coding review experiences could be one practice that can assist such students in meeting such skill expectations.

Funding Agency: NSF grant nos. *HRD-1912098* and *HRD-2011793*.

Fig. 1. a) Logitech C920 HD Pro – Web Camera, Google Loom recording software application; b) Zoom Video/Web Conferencing System

III. Methods

The objective of the PCR assessments were to examine the students' computational thinking models at different points throughout the semester. These assessments also exposed students to written code/solutions of a particular computational problem where they were required to analyze the written solution on the basis of *code correctness*, *code readability* and *code behavior/functionality*. The following subsections provide details for how these PCR assessments were administered to both the Fall 2019 (*face-to-face*) cohort of students and Fall 2020 (*virtual*) cohort of students enrolled in either CS2 or OOP, respectively.

A. Fall 2019 (Face-to-Face)

During this particular semester, both the CS2 and OOP students were assigned *one Logitech C920 HD Pro web camera* and instructed to create a *Google Loom recording software account* via the Cloud (Fig. 1a). As an alternative, students were also given the option to use the embedded web cameras on their personal laptops for this assessment. For both the *PRE* and *POST* assessments, there were three tasks assigned. The students were provided with either a paper-based or an electronic artifact that lists each of the three tasks to complete. Before *Task 1* began, each student was given a handout to help them get accustomed to using the web camera and Loom technology appropriately. For *Tasks 1* and *2*, the students were given a program solution to review. They were instructed to check for any syntactical and semantical errors that the solution contained and verbally explain aloud the corrections that are needed. Likewise, they were instructed to verbally explain the intended output that this program needed to project upon execution. Finally, the students were instructed to use a tool editor to build and verify their perceived solution for this program. For *Task 3*, the students were required to review a computational problem, conceptualize the necessary concepts, data structures, and programming paradigms needed to solve it while thinking aloud. Next, they were instructed to build their perceived solution using a tool editor. The students were given 30 minutes for each task. Afterwards, the students were instructed to stop the recording, save, and submit their recorded video as a *.mp4* file. Lastly, each student was instructed to complete a survey that assessed their experience with the coding review assessment. For both courses, the targeted paradigms used during the *PRE* assessment was *procedural* *programming*, while the *POST* assessment was comprised of *OOP programming* tasks.

B. *Fall 2020 (Virtual)*

Due to the COVID-19 pandemic, majority of the courses at this particular HBCU were held remotely during this particular semester. As an adjustment, a Zoom Video/Web Conferencing system was used to conduct the PCR assessments (Fig. 1b). The students received an electronic artifact that lists each tasks to complete using the a similar construct as noted for the Fall 2019 semester. Afterwards, the students were divided into groups of two and assigned a breakout room to conduct the PCR assessment. While in these breakout rooms, the paired students were granted access to record their breakout session for this assessment while also being able to share their personal computer/laptop screens to show their tool editor used to complete the assigned tasks. For the *PRE* PCR assessment, only two tasks were assigned in both the CS2 and OOP courses, respectively. This was done in order to adjust to this new virtual protocol and also account for any unforeseen challenges that could have arrived from this virtual style of assessment. For the *POST* assessment, the traditional three tasks were assigned. Using a similar protocol as noted during the Fall 2019 semester, the students were given 30 minutes to complete each task. Afterwards, the designated student who recorded the breakout session was instructed to stop the recording, save, and submit this recording as a *.mp4* file. Finally, each student completed a survey to assess their experience with the coding review assessments.

C. *Research Questions*

There were three primary research questions proposed for this study. Corresponding results and findings for these questions are addressed in Section IV (Results & Findings).

RQ1: Referring to Hundhasen et al.'s findings regarding this particular perceived expectation [14], *did the students involved in face-to-face coding review assessments performed better than those who did so virtually?*

RQ2: *Did the students in either courses and cohorts, successfully complete a later task more often than a previous task when the task expectations were the same?*

RQ3: *Did the students perform better at completing tasks during POST coding review assessments than PRE coding review assessments due to acquiring a general familiarity with the PCR assessment protocol?*

IV. Results & Findings

The type of data collected during this assessment was two-fold. One instrument collected *.mp4* recordings that showcased the students' performance while conducting PCR activities. The other instrument was the corresponding survey that succeeded after the students completed a PCR assessment. For this article, the results and findings discussed only reflect the data collected via the corresponding surveys for each corresponding assessments (*PRE and POST),* courses (*CS2 and OOP*), and semesters (*Fall 2019 and Fall 2020*).

A. *Demographics*

With exception to the *Fall 2019 OOP cohort*, majority of these cohorts for both courses during these two semesters were comprised of Black/AA students. Furthermore, majority of these cohorts were primarily male. The demographic makeup for the Fall 2019 OOP cohort were comprised of majority Black/AA students in addition to one Caucasian female student and one non-binary student of Latinx descent. Both CS2 cohorts were comprised of freshmen, sophomores, juniors, and seniors; where the upperclassmen were primarily comprised of Screen Writing and Animation (SWAN) majors, while the CS majors were primarily freshmen and sophomores. Both OOP cohorts were primarily comprised of sophomore and junior CS majors.

B. *PCR Performance Analyses*

The following tables and corresponding content will reflect PCR performances as it relates to *task completion, task matriculation,* and *assessment matriculation*.

1) Task Completion: This particular attribute measured the students' ability to effectively complete a given task as listed in given artifact. Completion was defined based on the combination of the students' ability to not only complete an assigned task, but also exhibiting an accurate review for correcting the *syntactical* and *semantical* errors that appeared in a given problem solution. Tables I and II provide descriptive details on the students' ability to complete a given task. To specifically address ***RQ1***, two two-tailed T-Tests were conducted. The first T-Test was a direct comparison on the task completion performances between CS2 students enrolled in the face-to-face (Fall 2019) and virtual (Fall 2020) courses. Task 3 was excluded from this comparison. The mean (μ) and standard deviation (SD) for both the *face-to-face* and *virtual* groups were found to be $\mu = 0.58$, $SD = 0.50$ and $\mu = 0.84$, $SD = 0.37$, respectively. The results did not reveal a statistical significance *(p=0.17)*. Similarly, the second T-Test was a direct comparison on the task completion performances

TABLE I. CS2 – PCR ASSESSMENT ANALYSES

CS2 Course *(means normalized on a scale of 0 to 1)*						
Assessment	***N***	***T1***	***T2***	***T3***	***%+ T1 to T2***	***%+ T2 to T3***
PRE (*Fall 2019*)	13	0.31	0.54	0.69	74.19%	27.78%
POST (*Fall 2019*)	10	0.70	0.80	0.70	14.28%	-12.50%
PRE (*Fall 2020*)	15	0.60	0.73	*	21.67%	N/A
POST (*Fall 2020*)	7	1.00	0.57	0.14	-43.00%	-75.44%
T = Task; %+ = Percentage Increase **Task 3 was not given in PRE (Virtual) assessment*						

TABLE II. OOP - PCR ASSESSMENT ANALYSES

OOP Course *(means normalized on a scale of 0 to 1)*						
Assessment	***N***	***T1***	***T2***	***T3***	***%+ T1 to T2***	***%+ T2 to T3***
PRE (*Fall 2019*)	20	0.55	0.65	0.40	18.18%	-38.46%
POST (*Fall 2019*)	16	**	0.50	0.20	N/A	-60.00%
PRE (*Fall 2020*)	23	0.43	0.87	*	102.33%	N/A
POST (*Fall 2020*)	15	0.73	0.80	0.67	9.59%	-16.25%
T = Task; %+ = Percentage Increase ***Task 1 was discarded for external reasons* **Task 3 was not given in PRE (Virtual) assessment*						

between OOP students enrolled in the *face-to-face* (Fall 2019) and *virtual* (Fall 2020) courses. The mean (μ) and standard deviation (SD) for both the *face-to-face* and *virtual groups* were found to be $\mu = 0.57; SD = 0.50$ and $\mu = 0.70; SD = 0.46$, respectively. Only Task 2 was used during this comparison (due to external factors that directly impacted Tasks 1 and 3, respectively). This result revealed a statistical significance *(p=0.01)*. Overall, these results revealed a mixed outcome regarding the capability for students in one instructional setting to outperform students who are in another. Both CS2 and OOP students who were *virtual* actually showed the potential trend of performing better than the students who were *face-to-face*. Likewise, the exclusion of certain tasks due to external reasons also played a confounding role that impacted the overall outcomes regarding this particular research question.

2) Task Matriculation: This particular attribute measured whether the students collectively showed a greater tendency to complete a later task after becoming accustomed to a previous task. Referring back to Tables I and II, descriptive details on the students' ability to show such tendencies upon matriculation through a given PCR assessment are shown. To specifically address ***RQ2***, quantitative analyses were conducted to determine any percentage increases that occurred between *Tasks 1* and *2* and *Tasks 2* and *3,* respectively, on the basis of task completion for both the *PRE* and *POST* PCR assessments. These analyses revealed that for students enrolled in either course or semester there was an increase in their ability to complete *Task 2* after attempting *Task 1*. This trend was specifically seen when students matriculated from *Task 1* to *Task 2*, with exception to the *Fall 2020 (CS2) POST Assessment*. Likewise, this trend was not noted during *Fall 2019 (OOP) POST Assessment* due to *Task 1* being ommited.

When observing the matriculation trends from *Task 2* to *Task 3*, the results were different. With exception to the *Fall 2019 (CS2) PRE Assessment*, all other assessments revealed a decrease in the students ability to complete *Task 3* after attempting *Task 2*. This was true for both courses and corresponding semesters. This trend was not noted during either of *the Fall 2020 (CS2) or (OOP) PRE Assessments* since *Task 3* was ommited for both. One possible reason for the decrease in the completion rate between *Task 2* and *3* is based on problem complexity. Aforementioned, *Task 3* typically required the students to review the assigned problem, conceptualize what need to solve this problem, then implement a solution to this problem. Unlike the prior two tasks, *Task 3* was composed of additional components for completion.

3) Assessment Matriculation: Similar to Task Matriculation, this attribute measured whether the students collectively showed an increase in their tendency to perform bettter on a given *POST* PCR assessment due to assessment familiarity. Task completion was used as the underlying indicator to determine such performance. Table III provides descriptive details on the students' ability to show a better performance on *POST* assessments. To specifically address ***RQ3***, quantitative analyses were conducted to determine any percentage increases that occurred between *Tasks 1, 2,* and *3* in relation to a given *PRE* and *POST* PCR assessment, respectively. These analyses revealed that this tendency was only true for *Task 1*, with exception to *Fall 2020 (OOP) group's*

TABLE III. ASSESSMENT MATRICULATION (PRE TO POST)

%Increase					
Assessment	***Pre N***	***Post N***	***Task 1***	***Task 2***	***Task 3***
Face-to-Face (CS2)	13	10	125.81%	48.15%	1.45%
Virtual (CS2)	15	7	66.67%	-21.92%	*
Face-to-Face (OOP)	20	16	**	-23.08%	-50.00%
Virtual (OOP)	23	15	69.77%	-8.05%	*

**Task 3 was not given in PRE (Virtual) assessments*
*** Task 1 was discarded for external reasons*

assessments due to *Task 1* being discarded for external reasons. With exception to the *Fall 2020 CS2 & OOP's PRE* assessment, which did not provide a *Task 3*, one possible reason for the decrease in the completion rates between *PRE* and *POST* assessments amongst *Tasks 2* and *3*, respectively, is due to both groups' exposure to more advanced paradigms at this point during a given semester. Typically, around the time a *POST* PCR assessment is administered, the CS2 students are being exposed to OOP-based paradigms and data structures involving *inheritance*, *polymorphism*, and *GUI programming*; while the OOP students are being exposed to *linked-lists*, *stacks*, and *queues* using this same paradigm.

V. CONCLUSION & FUTURE WORK

Based on the results from this study, the findings revealed no sufficient evidence that the *face-to-face* or *virtual* classroom settings, respectively, imposed a significant advantage or disadvantage to the students during these PCR assessments. The students who were virtual in the CS2 and OOP courses, however, tended to show a slightly (but not significantly) better performance on these assessments than their prior cohorts who were face-to-face. Moreover, the students collectively in both courses and semesters exhibited a slight trend for being able to perform better on a later PCR task that was relatively similar to a previous task; case in point *Task 2* vs. *Task 1*. This was found to be true during the majority of these assessments. When comparing the outcomes of this study to a closely relative study that was conducted by Hundhasen et al. [14], our outcomes were found to be slightly different. Students who were *face-to-face* in their study tended to be more productive with their coding review exercises than their *virtual* counterparts. In contrast, our study revealed no sufficient evidence that *face-to-face* or *virtual-based* constructs provided a significant advantage or disadvantage.

One future work will be to further explore the *.mp4* recordings acquired from the PCR assessments to conduct further quantitative analysis, as well qualitative and document analysis on these students' performance during this exposure. An additional future work will be to emphasize measures regarding the students' self-efficacy and attitudes towards these coding review practices.

ACKNOWLEDGMENT

This current work is funded by the National Science Foundation under grant nos. *HRD-1912098* and *HRD-2011793*.

REFERENCES

[1] Astrachan, O., Bruce, K., Koffman, E., Kölling, M., & Reges, S. Resolved: objects early has failed. In ACM SIGCSE Bulletin, Vol. 37, No. 1, pp. 451-452, February 2005.

[2] Badampudi, D., Britto, R. and Unterkalmsteiner, M. Modern code reviews-Preliminary results of a systematic mapping study. *Proceedings of the Evaluation and Assessment on Software Engineering*, 2019, pp.340-345.

[3] Bailie, F., Courtney, M., Murray, K., Schiaffino, R., & Tuohy, S. (2003). Objects First-does it work?. Journal of Computing Sciences in Colleges, 19(2), 2003, pp. 303-305.

[4] Beaubouef, T. and Mason, J. 2005. Why the High Attrition Rate for Computer Science Students: Some Thoughts and Observations. Inroads 37(2), 2005, pp. 103-106.

[5] Brown, R., Davis, J., Rebelsky, S. A., & Harvey, B. (2009, March). Whither scheme?: 21st century approaches to scheme in CS1. In ACM SIGCSE Bulletin, Vol. 41, No. 1, pp. 551-552, March 2009.

[6] Cain, C. C., Buskey, C., Bryant, A. M., Washington, G., & Burge, L. Research Implications of the Tech Exchange: Immersion of Howard University Computer Science and Informatics Students in Silicon Valley, 2019.

[7] Cooper, S., Dann, W., & Pausch, R. (2003, February). Teaching objects-first in introductory computer science. In ACM SIGCSE Bulletin, Vol. 35, No. 1, pp. 191-195, February 2003.

[8] Crosby, M. E. and Stelovsky, J. 1990. How Do We Read Algorithms? A Case Study. Computer 23, 1 (Jan. 1990), 24-35.

[9] Dodds, Z., Alvarado, C., Kuenning, G., & Libeskind-Hadas, R. Breadth-first CS 1 for scientists. In ACM SIGCSE Bulletin, Vol. 39, No. 3, pp. 23-27, June 2007.

[10] Fatima, N., Chuprat, S., and Nazir, S. "Challenges and Benefits of Modern Code Review-Systematic Literature Review Protocol," 2018 International Conference on Smart Computing and Electronic Enterprise (ICSCEE), Shah Alam, Malaysia, 2018, pp. 1-5.

[11] Goldwasser, M. H., & Letscher, D. Teaching an object-oriented CS1-: with Python. In ACM SIGCSE Bulletin, Vol. 40, No. 3, pp. 42-46, June 2008.

[12] Gosha, K., Kannan, V., Morgan, L., & Huff Jr, E. W. Strategic Partnerships to Enhance Data Structures and Algorithms Instruction at HBCUs. In *Proceedings of the 2019 ACM Southeast Conference*, pp. 194-197, April 2019.

[13] Hundhausen, C. D., Agrawal, A., & Agarwal, P. Talking about code: Integrating pedagogical code reviews into early computing courses. *ACM Transactions on Computing Education (TOCE)*, 2013, *13*(3), 14, pp. 1-28.

[14] Hundhausen, C.D., Agarwal, P. and Trevisan, M. Online vs. face-to-face pedagogical code reviews: an empirical comparison. In *Proceedings of the 42nd ACM technical symposium on Computer science education*, pp. 117-122, March 2011.

[15] Hundhausen, C., Agrawal, A., Fairbrother, D., & Trevisan, M. Integrating pedagogical code reviews into a CS 1 course: an empirical study. In *ACM SIGCSE Bulletin,* Vol. 41, No. 1, pp. 291-295, March 2009.

[16] Lev-Ram, M. (2015, March 10). Apple commits more than $50 million to diversity efforts [Online Article]. Retrieved November 26, 2018, from http://fortune.com/2015/03/10/apple-50-million-diversity/, March 2015.

[17] Lu, Yao, Xinjun Mao, Tao Wang, Gang Yin, and Zude Li. "Improving students' programming quality with the continuous inspection process: a social coding perspective." *Frontiers of Computer Science* 14, no. 5, 2020: 1-18.

[18] Oeda, S., & Kosaku, H. Development of a Check Sheet for Code-review towards Improvement of Skill Level of Novice Programmers. *Procedia Computer Science*, *126*, 2018, pp. 841-849.

[19] Pears, A., Seidman, S., Malmi, L., Mannila, L., Adams, E., Bennedsen, J., & Paterson, J. A survey of literature on the teaching of introductory programming. ACM SIGCSE Bulletin, 2007, Vol. 39(4), pp. 204-223.

[20] Pirttinen, N., Kangas, V., Nygren, H., Leinonen, J., & Hellas, A. Analysis of Students' Peer Reviews to Crowdsourced Programming Assignments. In *Proceedings of the 18th Koli Calling International Conference on Computing Education Research,* p. 21, November 2018.

[21] Simon, M. Google Sets Sights on HBCUs for Recruitment Efforts [Online Article]. Retrieved November 26, 2018, from https://www.nbcnews.com/news/nbcblk/google-sets-sights-hbcus-recruitment-efforts-n601476, June 2016.

[22] Washington, A. N., Burge, L., Mejias, M., Jean-Pierre, K., & Knox, Q. A. Improving Undergraduate Student Performance in Computer Science at Historically Black Colleges and Universities (HBCUs) through Industry Partnerships. In *Proceedings of the 46th ACM Technical Symposium on Computer Science Education*, pp. 203-206, February 2015.

[23] Where developers learn, Share, & build careers. (n.d.). Retrieved February 28, 2021, from https://stackoverflow.com/.

[24] Wiedenbeck, S., Ramalingam, V., Sarasamma, S., and Corritore, C. L. A comparison of the comprehension of object-oriented and procedural programs by novice programmers. Interacting with Computers 11, 1999, pp. 255-282.