2021 ASEE ANNUAL CONFERENCE
Virtual Meeting | July 26–29, 2021 | Pacific Daylight Time
Paper ID #32737

Building Computational, Social, Emotional Learning Skills into Undergraduate Computing Education Through Student-led Coding Camps

Dr. Gloria Washington, Howard University

Gloria Washington is an Assistant Professor at Howard University in Computer Science. At Howard, she runs the Affective Biometrics Lab and performs research on affective computing, computer science education, and biometrics. The mission of ABL is to improve the everyday lives of underrepresented and/or underserved humans through the creation of technologies that utilize human physiological and behavioral characteristics for identity recognition and/or understanding of human emotions. Currently, she is leading research that explores the role of affect and imposter syndrome on performance in computer science courses. Additionally, she is exploring the link between technology, mental health, and Black women's hair texture. Finally, she also works closely with clinicians within the Howard University Hospital to develop technologies for improving the lives of children and teenagers with Sickle Cell Disease through creation of tools for keeping track of their pain and encouraging them in moments of depression. The ABL is currently funded by the National Science Foundation, National Security Agency, and Microsoft. Before coming to Howard, she was an Intelligence Community Postdoctoral Research Fellow in the Department of Computing Science at Clemson University. She performed research on identifying individuals based solely from pictures of their ears. Dr. Washington has more than fifteen years in Government service and has presented on her research throughout industry. Ms. Washington holds M.S. and Ph.D. in Computer Science from The George Washington University, and a B.S. in Computer Information Systems from Lincoln University of Missouri.

Dr. Marlon Mejias, University of North Carolina, Charlotte

Dr. Marlon Mejias is an Assistant Professor in the Department of Software and Information Systems at UNC Charlotte. His research interests lie in the field of Socio-technical Systems, Educational Technology and Human Computer Interaction. He is interested in the application of persuasive technology and gamification to solve problems that are socially relevant. The primary thrust of his current research is in designing and implementing a socio-technical approach to improving the holistic education of undergraduate computer science students. Dr. Mejias has a B.Sc. in Systems and Computer Science from Howard University, a M.Sc. in Systems Engineering from The George Washington University and a Ph.D. in Computer Science from Howard University.

Dr. Marlon Mejias

Dr. Legand L. Burge III, Howard University

Dr. Burge is Professor and Executive Director of the Howard West Initiative and former Chairman of the Department of Computer Science at Howard University. His primary research interest is in distributed computing. Dr. Burge is also interested in Computer Science Education and Diversity, and Tech Entrepreneurship and Innovation. His work in CS Education and Diversity has primarily been focused on informal and personalized learning, and on the use of technology to aid in the socio-technical enculturation of underrepresented students in CS, K-12 initiatives, and diversity, equity, and inclusion beyond compliance. Dr. Burge practices design thinking as an innovative teaching methodology and promotes immersive learning and learning by doing. He co-teaches the Bison Startup and Bison Accelerate courses co-developed with YCombinator, in which students are guided through the process of founding technology startups. Dr. Burge has a strong interest in developing university innovation ecosystems for HBCUs as a way to create alternative revenue streams, attract and retain students, and prepare students with 21st century skills. He currently directs the HowU Innovate Foundry; which has consistently incubated on average 15 student led tech startups per year. Dr. Burge is a certified Lean Launchpad Educator, and Stanford D-School Design Thinker. He is a co-founder of XediaLabs, a DC-based incubation firm that provides training and technical consulting to local startups. He has been featured in several articles such as Bloomberg Business Week regarding diversity and inclusion in tech, and conducted a TedX talk on HBCUs role in the innovation and entrepreneurship ecosystem for African Americans. Dr. Burge is a Fellow of AAAS, BEYA Innovation Award recipient, and a Fulbright Scholar recipient.

Building Computational and Social & Emotional Learning Skills into Undergraduate Computing Education Through Student-Led Coding Camps

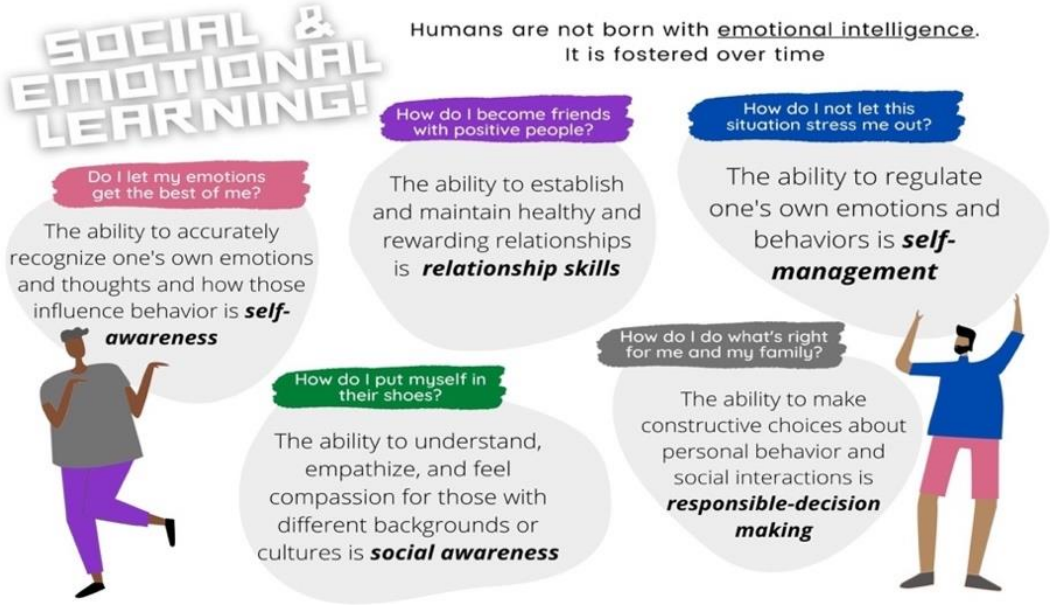
1 Introduction

ABET student outcomes like 1) "communicate effectively in a variety of professional contexts", 2) "recognize professional responsibilities and make informed judgments in computing practice based on legal and ethical principles", and 3) "function effectively as a member or leader of a team engaged in activities appropriate to the program's discipline" are soft skills that most undergraduate computing programs believe help students have longevity in their software careers [1]. Tech companies hiring undergraduate software engineers state that soft skills like ability to listen effectively, empathize with others, and be agreeable and cooperative during team discussions [2] are skills that new graduates often lack. Undergraduate computing capstone courses and sometimes software engineering courses are usually a student's first introduction to both working on a team-based project, creating a prototype solution, and learning how to identify the needs of the users and potential customers. These courses help students become socially aware of their behavior and practice responsible decision making while completing the milestones to develop a software application. Unfortunately, these experiences rarely embody the true experience of working on a diverse software engineering team with project managers, user experience designers, back-end and front-end coders, testers, and non-technical subject-matter experts. Literature in effective traits for tech employees [3] has shown that computing students can benefit from practicing social and emotional learning (SEL) skills before going into their fulltime careers. Courses like "*Introduction to Speech*" that are normally implemented outside of computing academic programs may not be enough to provide the "fail-fast"; fast-paced environment context for implementing SEL skills.

Mentorship and having representative examples of persons succeeding in a field is important to a computing student's sense of belonging [5] – [6], self-confidence, and success of undergraduate students as well as high school students. Mentorship by older generations such as professors, departmental leadership, or advisors helps college students avoid the pitfalls of repeating mistakes in undergraduate education. However, there are times when near-peer mentorship, or mentorship from slightly older students, may be more helpful than receiving mentorship from older individuals [4]. These cases include when younger generations feel like advice given from a mentor is out of touch with modern culture or the mentor's knowledge has exceeded its expiration date due to innovations that have created new career fields and require more advanced technical skills outside of a mentor's experience. In the past decade, the field of computing has introduced many new tech innovations and has created jobs that require advanced skills to be successful in these fields. Undergraduate students are expected to intern at tech companies in as early as their first year of undergraduate education and may have useful knowledge gained from working in the summer at these hi-tech companies. High school students may benefit from receiving mentorship from computer science undergraduate students in developing their computational thinking skills; as well providing examples observed from working or interviewing at a tech company.

Undergraduate led tech innovation camps may provide benefits that are two-fold: 1) undergraduate students can learn and practice computational thinking skills in a new setting and can further improve on their social and emotional learning skills and 2) high school students can learn computational thinking skills and social and emotional learning skills by working on projects centered around creating tech specific to their and their community's needs. With teaching high school students and acting as mentors; undergraduate students are forced to adapt quickly to different students' learning styles, remain clear and calm to teach coding concepts, and utilize code-switching techniques that allow them to communicate more effectively with younger, more slang-using audiences. Practices described in this research can be implemented in two-part capstone courses, independent study courses, or elective courses that require undergrads demonstrate hard skills as well as soft skills.

Figure 1: Social and emotional learning skills fostered in undergraduate students in this camp learning experience.



The primary research question this paper explores is how does acting as near-peer mentors and leading a tech innovation and entrepreneurship camp for high schoolers foster social and emotional learning skills in undergraduate computing students? Qualitative analysis was performed on feedback from undergraduate students and observational notes were gathered on these students for two iterations of the camp in the Summer of 2018 and 2019. The results of this work, along with background information on mentorship, computational thinking, and SEL skills is provided in the rest of this paper.

2 Background

2.1 Near-peer mentorship

Near-peer mentorship is the process of having students in a more advanced level of their education mentor students of novice level. The relationship between the mentor and mentee

is said to be successful because the participants are close in their matriculation. This closeness helps the mentor communicate and the mentee views the mentor as more relatable. The Walter Reed Army Institute of Research (WRAIR) implemented a version of near-peer mentoring between undergraduate and post-baccalaureate groups [7]. The mentor is hired through WRAIR's summer teaching internship program and serves as a teacher/mentor to middle and high school Gains in the Education of Mathematics and Science (GEMS) mentees. The researchers in this work found that the model was beneficial to both the mentee and mentor. It increased the interests and engagement of the high school students interested in STEM disciplines and developed deeper personal, educational and professional interests within the mentors.

The introduction of slightly older mentors with similar academic interests and backgrounds was described by Carle et al to study how undergraduate student's academic performance and social acclimation was impacted through the use of near-peer mentors in first year learning communities [4]. Since then, near-peer mentorship has proven effective in secondary education to engage high school students in their classroom experiences. Near-peer mentorship through using undergraduates to be role models for high school students in computing has also shown to have a positive effect on the grades and excitement of the high schoolers for attending class [8]. Additionally, undergraduate students that have acted as mentors to high schoolers described their experience as rewarding, helped them improve their self-confidence, communication skills, and enthusiasm for higher-level learning [8].

2.2 Computational Thinking

Computational thinking (CT) is a 21st century problem solving process that involves the use of fundamental computer science (CS) concepts to create solutions that incorporate the execution of an information-processing agent [9]. The most often cited CS concepts attributed to the CT process are 1) Decomposition; 2) Pattern recognition; 3) Abstraction; 4) Algorithmic design. These four concepts have been defined in Table 1 below. Other authors have also identified debugging and iteration [6] as part of the CT process. It should be noted that computational thinking is broader than CS and is not synonymous with programming [9] – [10]. Programming may be used for the implementation of the algorithmic design identified during the computational process; however, problems can often be solved using computing tools that do not require writing code.

There are many diverse approaches to teaching computational thinking. Some processes focus on using the technology as a tool to help solve a problem while others focus on problems that are to be solved entirely using technology. The AP CS curriculum focuses on a six-step process for CT; 1) connecting computing; 2) creating computational Artifacts; 3) abstraction; 4) analyzing problems and artifacts; 5) communicating; and 6) collaborating. This approach is beneficial for K-12 students because it emphasizes problem solving. It is also beneficial for recent graduates because many are transitioning into jobs where employers value persons that can analyze ambiguous software requirements, innovate, and solve problems [2].

Table 1: Concepts attributed to the computational thinking process.

Concept	Definition
Decomposition	Breaking down of a complex system or problem into smaller, more manageable parts.
Pattern Recognition	Identifying similarities or repeated processes in the decomposed parts of the problem.
Abstraction	Representing or modeling only the important details of the parts of problem or processes that show a pattern.
Algorithmic design	Developing a set of step-by-step instructions that can be implemented on an information processing agent to solve all or part of the problem.

2.3 Social & Emotional Learning Skills

Emotional Intelligence (EQ) is a human's ability to recognize and/or discern the feelings of oneself and other [4]. Emotionally intelligent persons try to understand others' attitudes, feelings, and emotions. Humans are said to have low emotional and social intelligence if they are unable to connect with others. Emotional and social intelligence (ESI), Table 2, have four main areas: 1) self- awareness; 2) social awareness; 3) self-management; and 4) relationship management.

Table 2: Four main areas of Social and Emotional Learning Skills.

Concept	Definition
Self-awareness	Understanding yourself and how you communicate with others and their perceived emotion from that communication
Social awareness	Understanding other people's reactions and emotions to communications
Self-management	Managing one's behavior in communications to influence emotions
Relationship management	Intercommunications between persons

2.4 Tech Innovation & Entrepreneurship

Innovation is the process of creating/applying better solutions to existing problem or creating/applying new solutions to meeting new requirements [3]. Entrepreneurship is the application of innovations to create revenue [3]. Tech innovation and entrepreneurship requires the entrepreneur to identify a problem that can be solved or improved using technology apply and then to interact with potential customers to figure out if there is are potential customers for that technology. Successful tech entrepreneurs need a high level of computational thinking to be able to identify tech opportunities and a high level of emotional intelligence and empathy in order to understand customer's wants and needs and to be able to effectively communicate and motivate employees [3].

Table 3: Five stages of Design Thinking defined.

Stage	Definition
Empathizing	Conducting user research to put aside one's own bias and understand the problem/need of the potential customer.
Defining	Stating the identified need or problem of the potential customer.
Ideating	Generating ideas to address the need or problem
Prototyping	Implementing inexpensive, scaled back, representation of possible solutions to the need or problem
Testing	Presenting the prototype to potential customers to get feedback on its potential to address the identified need or problem.

Design Thinking [11] is an iterative process where participants go through a five-stage process, shown in Table 3. The Lean Startup [12], shown in Table 4, approach is a method for launching entrepreneurial efforts in a more experiential way than traditional business plan development. The key features of the Lean Launchpad model are proposing a business model, exposing the underlying hypotheses associated with customers of that model, and performing experiments to validate or invalidate assumptions about that model's customers.

Table 4: Four stages of Lean Startup Process.

Stages	Description
Customer Discovery	The search for product-market fit. At this stage, startups are searching for the detailed understanding of their customer ecosystem.
Customer Validation	Team members should be able to articulate: customer workflows; customer ecosystems; customer-type mapping; or customer archetype profiles. Teams at this stage, work on developing minimum viable products (MVPs) which satisfy customer needs. Through rapid iteration, participants will work to refine an MVP that their target customer is willing to pay for.
Customer Creation	Requires teams to leverage elements of their Get-Keep-Grow strategy to test sustainable models to establish long-term Customer Relationships. The focus of the training is on developing hard metrics for Customer Acquisition Costs (CAC). At this stage, team members must have a product or MVP.
Company Creation	Teams should leverage the acquired knowledge of the business model canvas to begin to build their pitch decks and due diligence documents. While pitch decks are generally associated with high-tech startups seeking external investments, the process of developing the deck and working through projections for operating expenses is important for ventures who intend to use money from "Friends and Family" as well those interested in traditional debt financing.

3 Methodology

This paper describes the result of two iterations of a 6-week innovation and entrepreneurship camp for young high school boys that was taught by undergraduate computer science students in Summer 2018 and 2019. Invitations to participate in the coding camp were sent to local

area high schools in Washington, DC. Howard University professors leveraged their network with computer science teachers to send invitations for participation in the camp; these invitations were forwarded to the parents of students by the high school student if they were interested. Only the parents of the high school students could submit a completed application. Participants were chosen on a first come first serve-basis because there were limited spots in the camp, due to camp space.

3.1 Program Logistics

To help high school participants identify entrepreneurship opportunities, recently graduated undergrads were used to teach innovation and entrepreneurship. The "Design Thinking for Product Innovation" and the "Lean Startup" approach were used. The undergraduate computing students acted as near-peer mentors for the high school students in the camp. The age of the camp mentors was between 20 and 22. The age of the high school students was between 14 and 17. A total of twelve undergraduate seniors (six each summer iteration) were observed acting as the camp leaders and mentors in the summer before they started a tech job. A total 40 high school students (20 each summer iteration) were observed for this research. The camp mentors were assisted and observed by 2 graduate students (1 each iteration) and an independent evaluator (1 each iteration). The social and emotional learning skills of the near-peer mentors were observed and analyzed throughout the camp by both the independent auditor, graduate student supervisor, and the camp director. Focus groups were conducted before the camp during a 2-week pre-camp training, midway at the 3-week mark and at the end of the camp. High school students completed pre-camp focus groups, daily surveys that included open-ended questions were provided throughout the six-week experience, as well as reflection exercises.

3.2 Innovation and Entrepreneurship Camp Activities

As mentioned, the experience was more than just a coding camp; it gave high schoolers the ability to think outside of their "tech box", design and develop a prototype, and communicate the idea to an audience of experts and persons from within their local community. High school students operated under the assumption that their tech startup "business" would be eligible to work with Howard University NSF i-Corp [12] mentors from the University and would be given $500 if they pursued the idea after the camp with the University mentors. Undergraduate mentors were responsible for listening to high school student's ideas and giving the students feedback to ensure that the student's ideas were well thought out and well presented.

The program consisted of eight modules, see Table 5, taught over the six-week period, with each day running from 9 A.M. to 4 P.M. Various coding activities were designed to take students through all four computational thinking process components (decomposition, pattern recognition, abstraction and algorithmic design). Coupled with these activities was a Business Model Canvas [13] and Pitch Activity that required the high schoolers to 1) connect computing to solving a problem in their community; 2) create computational artifacts that prototype their solution; 3) model abstractions of the data that they would need to solve the problem; 4) analyze existing artifacts that may already address parts of their problems; 5)

collaborate to design and 6) communicate/pitch their potential solutions to judges comprising of members from various disciplines in the university community.

All undergraduate students hired had taken two undergraduate tech innovation and entrepreneurship courses at Howard university called Bison Accelerator and Bison Startup. All understood the steps of Design Thinking and Customer Discovery. Before starting the camp, all undergraduate student mentors learned the basics of user experience design. Mentors were given a lesson plan for all of the modules with the autonomy to update and influence content with socially and culturally relevant examples. An example of a module created by the near-peer mentor included ways to simply teach inclusive design and accessibility concepts to high schoolers by showing them examples from the web. They created a classroom project within the Web Design activity that showed the students how to provide constructive feedback to other projects to improve their initial design for the prototype solutions.

Table 5: Camp activities showing unit name, sample lesson, and a description.

Unit	Activity Description
Intro to CS and the Design Process	Given a list of 15 African Americans in computing, research information on each using your favorite web search engine and create a collage presentation.
Intro to Programming and Robotics	Create a robot program that will help students communicate their feelings through movements.
Design Thinking	Use technology to solve a problem in your community.
Canvanizer	Create a business model canvas for technology to address a problem in your community using the Canvanizer online tool.
Game Programming	Create a game on App inventor.
Computational Thinking, Problem Solving, and Python	Walk peers through a white boarded solution to a given problem.
Web Design	Create a website for the projects you built in the coding camp.
Mobile Application Development	Create a mobile application that builds your family tree, based on info you receive from your family.

In addition to the focus group interviews, the near-peer mentors engaged in self-reflection and formative assessment with the high school students at the end of each day. This would allow for them to course correct the next day. Mentors used the Think, Pair, Share technique [14] to discuss the good and bad of every daily lesson. High school students went first and provided their contributions and then the mentors would talk about the things that they learned from the students and how they could have helped them better that day. This information was captured in audio segments and transcribed by the camp administrator and also independent

evaluator.

High school students were assigned a near-peer undergraduate mentor and divided into groups of four. They were carried through elements of the AP CS computational thinking process [10] as well as the Design Thinking process. Students were asked to collaborate to identify a shared problem in their communities and connect how computing could be used to address that problem. The near-peer mentor was tasked with helping the high school students think through the various problems associated with their community or things that were important to them. In the first weeks of the camp, the near-peer mentors along with supervision from the graduate student took the high school students out into their local community to observe the common challenges associated with the neighborhood. The high school students could take pictures, after showing these pics to the undergrad mentor, on mobile devices so that they could remember the experience once they returned to the classroom. Once back in the classroom, the near-peer mentors would ask probing questions as exhibited from the "Empathize" step in Design Thinking to get the students to think deeply about their potential customers or users.

Once a problem was identified by the high schoolers and vetted by the near-peer mentors, the students were asked to conduct research in their community to find out if their perspective was biased or empathized with their community. Students interviewed their family members, University visitors, and faculty members within the University to gain insight about their ideas. Once students reported back on their findings to the entire class, the near-peer mentors would challenge them to formally define the problem and their solution in terms of "How Might We Statements". After presenting to the class, the near-peer mentors conducted training over the Ideate phase in Design Thinking to show the students how to ideate over the innovative solution they proposed. They high schoolers were asked to decompose the problem into a series of processes or steps that can solve the problem. They were then asked to identify data and patterns in the processing needed for each step of the process. Once the patterns were identified students were asked to identify only the necessary data that would be required to solve the process.

Finally, the high schoolers were shown how to create prototype interfaces using common rapid mobile apps prototyping software like MIT's App Inventor. The high schoolers created the prototype's interface that collect user-inputted data. Then, the near-peer mentors conducted training on how to validate their prototype by testing it with potential users. The high schoolers all tested each other's prototypes. Next, the high schoolers were shown how to analyze the results of their tests to identify usability issues, errors, or interactivity problems experienced by the user. The feedback from these prototyping activities were then used to help students develop their business model canvas.

The near-peer mentors showed the students how to use the online Canvanizer.com tool to develop a business model canvas. The Canvanizer activity required students to identify potential customer segments, state their value proposition, hypothesize their customer relationships and channels, hypothesize their key partners, key activities and resources, cost structure and revenue stream. This step required the groups to work closely with the near-peer mentors to help them understand how to develop an initial cost structure and revenue

stream. The undergraduate student had to carefully break down, by providing example existing applications that the high schoolers may have known, to show how cost and revenue can make or break a tech startup company. Common examples chosen by the undergraduate students were the advertising setup that companies like Facebook, Instagram, and Snapchat utilize to generate revenue. Another common example was add-on gaming packs that companies like EA Games used to generate revenue for gamers that wanted additional functionality. It was important that the near-peer mentors were familiar with technologies that the high school students used daily.

Pitching or effective communication is a crucial part of the program implemented in this work and was reinforced in every week of the experience. Near-peer mentors first showed students how to do they did their "University Intro" which consisted of a person's name, where they are from, their grade-level, and their hobbies. Then, the high school students took this structure and practiced their *"*Camp Intro*"* (their name, year in school, and any hobbies or activities that make them unique). This *"*Camp Intro*"* is important because it gives students the ability to practice communicating and speaking about themselves (and their capabilities) simply and clearly. This activity was engaged in several times during the camp as the students met with the tech entrepreneurs and other undergrad students visiting the camp.

Various coding activities detailed in the previous section were designed to take high school students through all four computational thinking process components (decomposition, pattern recognition, abstraction and algorithmic design). Coupled with these activities was a Business Model Canvas and pitch activity that required students to 1) connect computing to solving a problem in their community; 2) create computational artifacts that prototype their solution; 3) model abstractions of the data that they would need to solve the problem; 4) analyze existing artifacts that may already address parts of their problems; 5) collaborate to design and 6) communicate/pitch their potential solutions to judges comprising of members from various disciplines in the university community. The activities artifact and the presentations of the high school students made it very easy to assess whether or not the high school student was able to achieve the days computational thinking outcome. These tangible artifacts were also coupled with qualitative assessment of the students' responses.

After the students mastered their *"*Camp Intro*"*, near-peer mentors introduced pitching their very first idea. This idea did not necessarily make it into implementation for their final projects. Pitching their idea included communicating the societal need(s) for the technology and how the team is addressing the need through creation of the technology. Students were required to give presentations about any background information surrounding an idea. Students had to identify other applications and how their technology would be different. Students were also encouraged to ask insightful questions to other campers about how their potential ideas could be improved. This helped the students develop effective feedback skills as they learned how to properly ask questions relevant to a topic area. The near-peer mentors, after the student pitches, helped provided effective feedback on a myriad of things including how to internally regulate your emotions so that you do not get stressed out presenting, the content of the presentations and how to improve the usability of the tool, as well as how well the high school team cohesively described their tool. The undergraduates and high school students looked forward to this activity as the teams developed healthy competition with

competing to win the "Best Pitch" voted upon by the entire mentor team, including the camp director.

Customer discovery [12] was performed in the last week of the program as well. The near-peer mentors provided ground rules to the high school students that included clearly identify themselves to the potential customer, no harassing persons that did not want to provide feedback, be respectful of people's time when asking them follow-up questions, and teams must speak to a minimum of ten people. Students interacted with locals around Washington, DC. Students were asked to dress comfortably, yet professional, so that they can engage with their community about their ideas and not be judged based off their attire. Questions the high school students asked the community were: "Would you use a tool like the one developed?", "What features should a tool like this have?", and "How much would you pay if this technology existed?" This lesson was approximately four hours. Once this activity was complete, the near-peer mentors worked with their teams to identify feedback that can be utilized to improve the prototype designs and to accommodate a range of ethnicities and experiences due to the diverse neighborhood around the University.

On the last day of the program, the high school students demoed their technologies and pitched their ideas to a group of five judges from various disciplines, invited guests, and their families. The near-peer mentors were also heavily involved with giving pep talks to their teams to ensure they do not let the new audience members throw them off their "pitching game" and manage the anxiety associated with pitching in front of strangers and their friends and families. The near-peer mentors were not able to vote but did enthusiastically cheer for their group in the audience as the high school teams pitched.

3.3 Assessing Social & Emotional Learning Skills in Undergraduate Mentors

Undergraduate mentors were led through a two-week training before the starting any of the camp activities. Focus groups were held at the midpoint of the camp in week three and at the end of the camp. Questions asked to undergraduates were:

Q1. Do you think that mentorship is important?
Q2. Did you have a mentor in high school?
Q3. What made you get into computer science?
Q4. What makes you feel you can be a good mentor in the summer camp?
Q5. Do you see yourself as a computer scientist?
Q6. What advice would you give to high school students wanting to go into computer science or technology?
Q7. What have you learnt over the course of the camp?
Q8. Would you ever want to work with such a coding camp again?

The same questions were asked at the end of the camp as well. These sessions were videotaped. Additionally, asked were questions about advice they would provide to the high school mentees about computer science and their abilities. Also, undergraduates were asked what tech companies they were starting fulltime employment with and if the activities in the camp would help them in their future positions.
Daily debriefings and planning meetings were held in order to assess changes in the near-peer

mentors. In these meetings, undergrads were asked to self-assess (SA): SA1) how well they were able to connect the concepts being taught to examples and references that were relatable to the student; SA2) how well they were able to guide the creation of the desired learning outcome artifact; SA3) how well they were able to abstract out the important details of the day's activities without bogging down the high school students in unnecessary technical information; and SA4) how comfortable they were communicating their analysis to the high school students' work. During the planning sessions, the undergrad responsible for the next day's lessons met with the graduate student to go over the lesson plans and to discuss ways to bring in socially and culturally relevant examples they were going to use, how they were going to guide the high schoolers through creating desired artifacts, and to brainstorm possible issues the high schoolers may have. A mapping of the focus group questions and daily debriefing self-assessments to social and emotional learning concepts can be seen in Table 6.

The independent auditor met daily with the graduate student supervisor to gather feedback on the day's activities, provide insight into the high schoolers' evaluations, and discuss internal improvements and suggestions noted by the high school students. Once a week the mentors, the graduate student, and the camp director met to evaluate the week's activities and to coordinate plans for the next week.

Table 6: Mapping social and emotional learning concepts to focus group questions (Q) and daily debriefing self-assessments (SA).

Concept	Focus Group Question
Self-awareness	Q3, Q4, Q5, Q6, Q7, SA1, SA2, SA3
Social awareness	Q1, Q5, SA1
Self-management	SA4
Relationship management	Q4, Q6, Q1, SA2, SA3

3.4 Qualitative Data Analysis

Thematic analysis, using techniques described in [15], of undergraduate student feedback was conducted to pull out the common themes mentioned by the 12 undergraduate participants. Because of the small participant number, reflexive thematic analysis was used in which broader patterns and themes were observed from the undergraduate feedback. The steps involved in thematic analysis include 1) focus group interviews and student reflection feedback were transcribed using Google Docs Voice Typing tool, 2) thematic coding of important features, 3) generating and reviewing themes, and 4) mapping these themes to social and emotional learning skills. Additionally, themes generated were reviewed by the independent auditor, along with the research team to determine important areas of improvement for future development of the program. Similar quotes from student feedback were grouped together and iteratively processed to identify important student quotes that clarified how social and emotional learning skills were fostered in the undergraduate participants. These perceptions are reported in the next section.

4 Results

Through reviewing features from data capturing participant feedback, themes related to several broader areas were captured and mapped to social and emotional learning skills. These themes include teaching coding to younger people, capability of the participants to learn new ideas, belief in their ability to teach the students, and fulfilling their desire to be an entrepreneur and teach it to others. Additionally, the theme that resonated with every mentor is the need to be a role model to young black men high schoolers. Quotes from the original responses are included to allow the voice of the undergraduates to tell their own narrative.

Self-awareness Category. Feedback from the near-peer mentors included they did not realize how hard it was to be a teacher to young students that did not always listen.

"Before starting the camp, I was afraid of fear of failure. Some of the kids in the camp have this fear too. Once you know your fear, you have the opportunity to overcome them. Since they are tackling them earlier, they can overcome them sooner."

"I learned a lot about myself...I have a fear of failure and rejection. When we were talking about it and discussing it, a lot of the kids were facing the same issues."

"I learned how to deal with change and with people. I learned I can handle kids from different areas and teach them all."

"There is a lack of Black CS Mentors. There were limited opportunities for me to display my knowledge and show what I discovered about myself and the knowledge that I have to share."

Social Awareness Category. One of the undergraduate mentors mentioned the culture of the University he attended "breeds effective communication skills on the first day" due to the "University Intro" taught to all students in university's Orientation class. This mentor struggled with teaching this effective communication to the high school students that may not be familiar with communicating with different types of people outside of their race, culture, or age group.

"Working in industry, I didn't see a lot of people that don't look like me. This was perfect opportunity to empower the next generation so they can be leaders."

"Growing up, I knew I wanted to go into computer science, but I didn't see many people that looked like me. I can help by being an example because they also don't see examples that look like them too. They then can go on to lead the next-generation too."

"One of things they say is that Black Men do not have good communication skills. This camp really helped them all with speaking and professional development skills that they can use in the workforce."

Self-management Category. Most of the Think, Pair, Share comments that were provided by the near-peer mentors related to the ability of the high schoolers to perform a coding task and how they did not believe in themselves. Also, one camp mentor mentioned that he had

“admired the students so much because they were so advanced in their thinking” in relation to when he was in high school. He mentioned if he had had their mindset, he would be much farther and “possibly a millionaire”. He encouraged them to pursue their ideas and to make sure that they don’t give up on themselves. Another mentor said that the ideas of the high schoolers helped him see how he wanted to circle back and pursue entrepreneurship after doing at least one year in industry. Additionally, this mentor provided comments related to the ability of the students to perform whiteboard coding and explain their solutions without feeling stressed out. Many of the students would “fight” to be heard so that they could have the chance to “finesse” their whiteboard skills.

Another undergraduate mentor noticed he had to “become better at not showing frustration to the students” learning the coding lessons taught in the camp. He recognized how his frustration could deter students in the camp from wanting to pursue computer science in the future.

“I learned I can learn how to be a kid! I can teach them at different levels, and they can be able to show others in the program at different levels too.”

Relationship Management Category. Undergraduate students also recognized that the way that you phrase communications to young black men is very important. They mentioned they "had to adjust the way they normally talk to each other", which is usually in a joking antagonist manner. They realized they "cannot do this with most of the high school students within the camp" because some of the students "looked up to them" and would “take it to heart”.

One such mentor talked about one high schooler that would stay at the end of the camp and ask questions to him afterwards. He mentioned that that was his "buddy" and that he "did not realize he could have an effect on people like that". Additionally, he mentioned he would have to "adjust the way that he talks" to him because he looks up to him so much just by him "tagging along with everything that he does".

“Your ideas are that good. If you guys don’t pursue these ideas after this over, I’m stealing them. They are that good!”

“_____ had the best icebreakers. They weren’t generic. I liked working with him on getting the students ready to go in the morning.”

“You’re capable of so much more than you think! Conquer the world.”

4.1 Challenges

Although undergraduate-led coding camp experiences help students build their social and emotional learning skills, they come with many challenges that should be thought about before implementing a similar program. Challenges with running the camp pertained to ways the undergraduate mentors organized some of the activities. The camp administrator found that the camp mentors would play video games with the high school students during breaks

and at the end of the day. Often, the mentors would have to be corrected by the camp administrator to turn off the video games and get back to helping the high schoolers after the lunch break. The administrator noted these corrections occurred outside of the camp participants. Additionally, the camp mentors would invite visitors from outside of the camp to view some of the activities. This proved difficult as the high schoolers had to adjust to visitors and were distracted during activities where visitors were unexpected.

In addition to the social and emotional skills learned by the near-peer mentors during the camp, they were able to practice and mature their computational thinking skills through activities related with the camp. Computational skills like decomposition, pattern recognition, extraction, and algorithmic design were demonstrated by the near-peer mentors. Activities like Design Thinking and Customer Discovery demonstrated the near-peer mentor's ability to create and design an activity for the high schoolers that would help them become more empathetic towards the potential user for their innovation idea. Although the camp administrator supervised the Design Thinking and Customer Discovery activities, the near-peer mentors designed the activity to make sure that the high school students could go out into their environment and talk to people they were designing their technology for.

Before engaging in this activity, the camp administrator talked with the mentors and they broke down the activity into two different parts. The first part was the Design Thinking activity and how to make it relevant to high school students in large cities. The second part was Customer Discovery where the undergraduates discussed with the camp leadership about different neighborhoods that the high schoolers would visit to survey customers or potential users. The mentors' talked about the processes needed to keep the students safe, how many students would be assigned to a particular mentor, what to do in case of an emergency, and the necessary documentation and materials needed to have a successful Customer Discovery experience for the high schoolers. After coming to an agreement, the mentors implemented the activity and made sure that it ran in the allocated amount of time. Much discussion is needed when working with the near-peer mentors on creating activities that are safe for the high school students and mentors when gathering feedback from potential users and customers. To ensure the students acted the part of a typical entrepreneur, they all dressed in business casual clothes.

After the activity, the mentors also made sure that the high schoolers got home safely and documented the results of the activity. This activity alone demonstrated abstraction, algorithmic design, pattern recognition, and composition. because the near-peer mentors were pivotal in designing and creating the best experience for the high schoolers. It also helped them take their computational thinking skills outside of the traditional coding experience and apply it to lessons that high schoolers could remember and want to engage in.

5 Discussion

Some computing programs may lack opportunities to provide experiences for their students to practice social and emotional learning skills. Undergraduate seniors can possibly use the time between accepting a full-time position and starting their careers to participate in a

coding camp centered around tech innovation and entrepreneurship. Most seniors venturing into the career world have not fully developed effective communication skills necessary for being a team-player. Rarely, are they given undergraduate assignments that require them to empathize with persons outside of their culture and background in computing education. Most undergraduates are also unaware of the concept of emotional intelligence and how it might affect their future job performance. Engaging with high school students that look up to them and want to be like them may help undergraduates develop their self-awareness, responsible decision making, and social awareness that can help them in future positions.

The average tech company spends millions of dollars on effective communication training so that its workers can learn how to work in remote or distributed teams. If undergraduate students can better learn social and emotional learning skills in their undergraduate education, they may become more successful in obtaining full time offers and being promoted to leadership at technology companies.

Educators at the undergraduate level in computing education can leverage activities taught in the coding camp by working with their local high schools that do not have or are lacking computer science teachers. Often, schools cannot retain qualified computer science teachers in high schools because of the high salary that these educators can get in industry. Working with undergraduate mentors in the summer and during the academic semester may help to alleviate the stress put on high schools that rely on computer science teachers that are few and far between. Activities that can be leveraged by undergraduate computing professors are getting undergraduate students to design a coding lesson for their final projects that can be taught in introductory programming classes to high school students. Freshman undergraduate computing students are not that far removed from high school. They are more likely to know points of interest to engage high school students. High School computing teachers could have visit days with their classes to implement the lessons created by the undergraduate students. Additionally, high schoolers could learn about being an undergraduate computing major and seeing examples of young person's surviving and thriving in technology. Both the undergraduate and high school educators could benefit from this activity that engages both students in mentorship and direct implementation of their computational thinking skills.

There are challenges with using near-peer mentors for high school computing lessons. Near-peer mentors that have just graduated from undergraduate computing academic programs may still be immature. They will need guidance and supervision on things that may inadvertently derail lessons like inviting visitors that change the dynamic of a classroom or not learning how to self-regulate their emotions around young persons that look up to them as role-models. Additionally, some high schools may have rules in place that may make it difficult for undergraduate near-peer mentors to give visiting lessons in existing computer science classes. However, administrators at both institutions can work together to build programs that has policies in place to mitigate any issues that may arise due to either the immaturity of the near-peer mentors or the high schoolers.

6 Summary & Conclusion

In this paper we presented a coding camp that helped undergraduate students gain social and emotional learning skills through activities centered around tech innovation and entrepreneurship. Coding camps aimed at tech innovation and entrepreneurship combine computational thinking and emotional intelligence needed for understanding potential users and customers; while increasing the success of working on distributed teams. Additionally, this paper suggested ways in which high school computing and undergraduate computing instructors can collaborate so that both their students can gain computational thinking and social and emotional learning skills. There are challenges involving undergraduate students in teaching high schoolers, however, they can be mitigated if leadership within the undergraduate institution and those at high schools contribute to policies and safeguards essential for maintaining a beneficial partnership.

7 References

[1] "Criteria for Accrediting Computing Programs, 2020 – 2021 | ABET", Abet.org, 2021. [Online]. Available: https://www.abet.org/accreditation/accreditation-criteria/criteria-for-accrediting-computing-programs-2020-2021/. [Accessed: 24- Apr- 2021]

[2] Yilmaz, M., O'Connor, R. V., Colomo-Palacios, R., & Clarke, P. (2017). An examination of personality traits and how they impact on software development teams. Information and Software Technology, 86, 101-122.

[3] F. Yıldırım, I. Y. Trout, and S. Hartzell, "How are entrepreneurial intentions affected by emotional intelligence and creativity?," Period. Polytech. Soc. Manag. Sci., vol. 27, no. 1, pp. 59–65, 2019.

[4] J. E. Zins, Building academic success on social and emotional learning: What does the research say? Teachers College Press, 2004.

[5] D. Jaffee, A. Carle, R. Phillips, and L. Paltoo, "Intended and unintended consequences of first-year learning communities: An initial investigation," J. First-Year Exp. Stud. Transit., vol. 20, no. 1, pp. 53–70, 2008.

[6] M. Mejias, K. Jean-Pierre, G. Washington, and L. Burge, "Underrepresented Groups Threats to Belonging in Computing," in 2019 Research on Equity and Sustained Participation in Engineering, Computing, and Technology (RESPECT), Feb. 2019, pp. 1–4, doi: 10.1109/RESPECT46404.2019.8985905.

[7] J. S. Mokri, N. Okamoto, and S. I. Neagu, "Implementation of a 'Near-Peer' Mentoring Program between a High School Technology Class and a University Senior Design Engineering Class," presented at the 2020 ASEE Virtual Annual Conference Content Access, Jun. 2020, Accessed: Mar. 01, 2021. [Online]. Available: https://peer.asee.org/implementation-of-a-near-peer-mentoring-program-between-a-high-school-technology-class-and-a-university-senior-design-engineering-class.

[8] L. S. Tenenbaum, M. K. Anderson, M. Jett, and D. L. Yourick, "An innovative near-peer mentoring model for undergraduate and secondary students: STEM focus," Innov. High. Educ., vol. 39, no. 5, pp. 375–385, 2014.

[9] V. J. Shute, C. Sun, and J. Asbell-Clarke, "Demystifying computational thinking," Educ. Res. Rev., vol. 22, pp. 142–158, Nov. 2017, doi: 10.1016/j.edurev.2017.09.003.

[10] T.-C. Hsu, S.-C. Chang, and Y.-T. Hung, "How to learn and how to teach computational thinking: Suggestions based on a review of the literature," Comput. Educ., vol. 126, pp. 296–310, Nov. 2018, doi: 10.1016/j.compedu.2018.07.004.

[11] Brown, T. (2008). Design thinking. Harvard business review, 86(6), 84.
[12] NSF Award Search: Award # 1450443 - I-Corps Sites: Howard University. [Online]. Available: https://www.nsf.gov/awardsearch/showAward?AWD_ID=1450443. [Accessed: 28-May-2021].
[13] Blank, S. (2013). Why the lean start-up changes everything. Harvard business review, 91(5), 63-72.
[14] Kaddoura, M. (2013). Think pair share: A teaching learning strategy to enhance students' critical thinking. Educational Research Quarterly, 36(4), 3-24.
[15] Virginia Braun and Victoria Clarke. 2006. Using thematic analysis in psychology. Qualitative Research in Psychology 3, 2: 77--101.