Code Beats: A Virtual Camp for Middle Schoolers Coding Hip Hop

Douglas Lusa Krug
Department of Computer Science
Virginia Commonwealth University
Richmond, VA, USA
Instituto Federal do Paraná - IFPR
União da Vitória, PR, BR
krugdl@vcu.edu

Edtwuan Bowman
Department of Computer Science
Virginia Commonwealth University
Richmond, VA, USA
bowmanel2@vcu.edu

Taylor Barnett
Department of Music
Virginia Commonwealth University
Richmond, VA, USA
barnettt@vcu.edu

Lori Pollock
Department of Computer and Information Sciences
University of Delaware
Newark, DE, USA
pollock@udel.edu

David Shepherd
Department of Computer Science
Virginia Commonwealth University
Richmond, VA, USA
shepherdd@vcu.edu

ABSTRACT

In spite of the efforts to provide computer science education for all, the percentage of Black and Latino Americans entering the computer science (CS) field has been stagnant for years. In an effort to attract and engage students many summer camps and after-school clubs use robotics, video-games, and even IoT devices, but these approaches seem to only attract those already considering STEM careers, a population low in Black and Latino students. To attract Black and Latino students to computer science a promising approach is to engage with their culture, making CS relevant to them personally. To this end, we present an approach that teaches middle school students to program using hip hop beats, intentionally leveraging a genre of music that appeals to a wide array of urban youth of color. This approach, called *Code Beats*, uses extensive scaffolding to support beginning students, authentic-sounding beats to engage students, and a expressive programming environment to support creative freedom. We present the results of our pilot camp, where students clearly showed an increase in computing enjoyment, confidence, belonging, and persistence. By the end of this course, *all* students were able to create their own, original beat from scratch, suggesting their progression to the Create phase of the Use-Modify-Create framework.

CCS CONCEPTS

• **Social and professional topics → K-12 education**; **Computational thinking**; • **Applied computing → Sound and music computing**.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
SIGCSE '21, March 13–20, 2021, Virtual Event, USA
© 2021 Association for Computing Machinery.
ACM ISBN 978-1-4503-8062-1/21/03…$15.00
https://doi.org/10.1145/3408877.3432424

KEYWORDS

computer programming, music, minorities, Sonic Pi

ACM Reference Format:
Douglas Lusa Krug, Edtwuan Bowman, Taylor Barnett, Lori Pollock, and David Shepherd. 2021. *Code Beats*: A Virtual Camp for Middle Schoolers Coding Hip Hop. In *Proceedings of the 52nd ACM Technical Symposium on Computer Science Education (SIGCSE '21), March 13–20, 2021, Virtual Event, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3408877.3432424

1 INTRODUCTION

Educators and business leaders alike are realizing that Computational Thinking (CT) is a fundamental skill that students and workers need to know in order to be active participants in the digital economy [23]. The demand for these skills is huge; not only are there perpetually hundreds of thousands of open computer science positions [19] but computational skills are becoming a core competency for many modern jobs [17, 18].

Unfortunately, even though Computer Science (CS) is increasingly seen as a key skill, CS is still not available to all [21]. The problem starts in the early years of education, as CS has traditionally had a low presence in K-12 schools. In the United States, only 15 states require all high schools to offer CS courses and only five states require CS for elementary and middle school students [6]. Worse yet, schools with high concentrations of minority students are less likely to offer CS [6]. The current CS for All movement across the United States seeks to address this challenge by putting equity at the forefront in an effort to democratize CS education. Unfortunately, as we can see from the continued lack of diversity in CS AP test takers [3], these efforts have not dramatically increased the percentage of Black and Latino youth entering CS.

It is especially unfortunate that current efforts do not target middle school children [6], as these years are crucial to their future; one study found that students interested in a STEM career when they were in middle school were three more times likely to graduate from college with a science degree [22]. When targeting middle school children one key issue is interest and engagement.

To address this issue many informal learning activities focus on the robotics and video game domains. Unfortunately, these domains may present many conceptual and logistical challenges to learners and teachers (i.e., equipment). Furthermore, these domains tend to attract students that are already focused on STEM, which does not help increase diversity.

One alternative way of attracting students is to create an approach that engages with their culture, which has been shown to be effective [8, 13]. To that end, we chose hip hop, an art form that has long been a centerpiece of Black and Latino culture [16], as a way of engaging students in our three-week online camp for middle-school aged students. We used Sonic Pi [1, 2], a code-based music creation tool that allows students to write code that produces high quality beats, and a set of scaffolded exercises to provide students with hands-on STEM activities. *Code Beats*, which uses scaffolded exercises in professional musical coding software provides a low-floor high-ceiling environment that is important for engagement and also enable all students doing the activities with chance to extend it to higher levels. This approach allows students to move from merely users to becoming creators (see the Use-Modify-Create framework [12]). In a pre and post survey aimed at measuring students attitude towards computer science all four categories of questions that were asked showed a statistically significant difference. Furthermore, 100% of students were able to create at least one of their own beats by the end of class, suggesting substantial progress according to the Use-Modify-Create framework.

2 RELATED WORK

Several groups have used music with the goal of motivating students to learn programming. EarSketch [14], a programming environment for remixing music developed at Georgia Tech, has been shown to increase engagement for all, and especially for both females and racial minorities. The EarSketch approach "..focuses on the level of beats, loops, and effects more than individual notes, enabling students with no background in music theory to begin creating ..music immediately, with a focus on higher-level musical concepts such as ..mixing." [14]. That is, EarSketch emphasizes immediacy. When viewed through the "Use-Modify-Create" framework, EarSketch primarily supports the Use and Modify phases; users can play existing music clips and combine them, but the creation of new melodies and harmonies is not possible. Unfortunately, the Create phase, which EarSketch does not fully support, is crucial to exercising CT skills like abstraction, automation and analysis [12]. Building on EarSketch's learnings, *Code Beats* uses scaffolding to ensure immediacy, but also affords users a high ceiling, enabling them to create their own melodies and harmonies.

While EarSketch favors immediacy, other platforms emphasize depth. JythonMusic [15] allows users to generate individual notes or chords instead of just mixing existing audio files. However, because JythonMusic has been used in the context of a first year university music appreciation course at the College of Charleston, where many genres of music are discussed, a deep knowledge of music theory is required by its current curriculum [15]. TunePad [10] is another approach to engaging students via music mixing, created in a computational notebook style. Its mix of visualization and computation, with its focus on usability and ease-of-use, make it well-suited for a broader audience. Our approach uses a more traditional coding environment, as our goal is to give students an introduction to standard computer science tooling and concepts.

3 *CODE BEATS*

Code Beats uses a musical approach to teaching CS concepts. Its lessons provide graded scaffolding that allows beginners to thrive, intermediate students to grow, and experts to go deep. It uses hip hop to increase student engagement and cultural relevance.

3.1 Software Tooling

To realize the *Code Beats* approach we had to select a music coding platform; for this we chose Sonic Pi [1, 2]. The Sonic Pi has been used as a live-coding platform by performers at night clubs, festivals, and conferences [24]. To create in Sonic Pi students write code that is similar to Ruby[1]. As shown in Figure 1, to play notes students type the command *play* followed by the name of the musical note, selecting a synthesizer with command *use_synth*, and can play pre-recorded samples with the command *sample*.

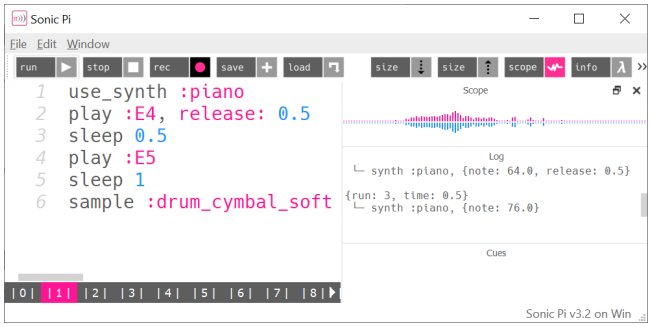

Figure 1: Code Example in Sonic Pi

3.2 Genre Choice

One promising approach to engaging all students in CS is through the use of culturally relevant pedagogy [9, 11]. Culturally relevant pedagogy "validates, facilitates, liberates, and empowers ethnically diverse students by simultaneously cultivating their cultural integrity, individual ability, and academic success" [9]. *Code Beats* uses hip hop to capture the imaginations of a wide array of urban youth of color from diverse places of origin [4]. For instance, one lesson uses Kendrick Lamar's song Humble as a backdrop for student creation. Humble is a significant song in the hip hop canon, from a notable and influential artist.

3.3 Adaptable Scaffolding

Code Beats is based on the three-stage Use-Modify-Create framework, which was used to support youth's acquisition of CT [12]. In this framework, students progress through the following stages: **Use** - students are consumers of someone else creation; **Modify** - students modify existing code with increasing levels of sophistication; and **Create** - students design and implement their own ideas.

[1] http://www.ruby-lang.org/

Introductory *Code Beats* lessons begin with students simply using and listening to existing code. They are provided a complete file that creates a beat and are instructed on how to load it into the Sonic Pi and play it. Next they move onto the Modify stage, where they are provided a track that needs updating. It contains a drum, bass line, and percussion track but requires the student to modify melody track, as the melody starts. As shown in Listing 1 students would be asked to read the comments and modify the existing (single-note) melody, which appears at the bottom. By altering the notes, playing the file, and listening to the changes they begin to understand the implications of their code. Beginning activities focus on fill-in-the-blank coding, Parson's problems [20], and other modification tasks.

Listing 1: Code Example 1

```
live_loop :melody do
  ##################################
  #  Create a melody by modifying the code below.
  #  To match the background music, use these notes:
  #    Ab4, Bb4, Cb5, Db5, Eb5, Fb5, Gb5
  #  To match the dominant chord, emphasize these notes:
  #    Ab4, Cb5, Eb5
  ##################################
  play :Eb5
  sleep 1
  play :Eb5
  sleep 1
  play :Eb5
  sleep 1
  play :Eb5
  sleep 1
  ##################################
end
```

Listing 2: Code Example 2

```
# You can increase or decrease the velocity of your sound
use_bpm 90 # BPM = Beats per Minute
##################################
# Complete your beat by adding a melody to the first live_loop
##################################
# First live_loop - here you can create your melody
live_loop :melody do
  # You can choose any synth that you would like to use
  use_synth :beep
  use_synth_defaults amp: 1
  #  You can use these notes to create your melody:
  #    Eb5, Fb5, Gb5, Ab5, Bb5, Cb6, Db6
  #  You should emphasize these notes:
  #    Eb5, Gb5, Bb5
  sleep 4
end
# Second live_loop - this controls the bass drum
live_loop :bassDrum do
  use_sample_defaults amp: 1
  # Emphasize beat #1 in this loop.
  sample :drum_heavy_kick
  sleep 2.5
  sample :drum_heavy_kick
  sleep 1.5
end
# Third live_loop - this controls the snare drum
live_loop :snareDrum do
  use_sample_defaults amp: 1
  # Emphasize beats #2 and #4 in this loop.
  sleep 1
  sample :sn_dolf
  sleep 2
  sample :sn_dolf
  sleep 1
end
```

As students progress throughout the class the scaffolding adapts to their knowledge. For instance, more advanced students are given scaffolding that provides some tracks, such as drums and percussion, but asks them to create one or more tracks, such as a melody and bass line. As shown in Listing 2, students are asked to create a melody (first loop). However, they are still given guidance on which notes to play in comments, scaffolding which allows keeps the focus on computing instead of learning extensive musical theory. By the end of the class students progress to a nearly empty scaffold, with only placeholders for each instrument.

3.4 CS Concepts and *Code Beats*

Code Beats enables teaching a variety of CS concepts, including: Sequencing, Parameters, Loops, Data Structures, Functions, Variables, and Conditionals, covering the concepts in Units 4, 5, and 7 in the code.org CS Principles Curriculum Guide [5]. While the above example lessons focus on demonstrating the fundamentals of our approach, such as our adaptive scaffolding, for simplicity they focus on the concepts of sequencing and repetition. However, our activities cover all of the concepts listed above.

For instance, we teach students to use **parameters** to change the balance between musical tracks, ensuring that the drums are quieter than the melody. They use numbered **repetitions** to repeat a measure seven times before breaking the monotony by playing a variation for the eighth measure. They must understand and use chords and scales, which are **data structures**, allowing them to program melodies over chords without having to memorize each note in a given scale. They use **functions** to create unique sounds for a given chord, like slightly offset notes that mimic the strumming of a guitar. Finally, they use **variables** with **conditionals** to play certain notes or samples every few measures, giving an otherwise monotonous beat some spice. As the class moves forward we teach more of these concepts and, once taught, they become a requirement for inclusion in each new beat.

4 *CODE BEATS* VIRTUAL CAMP

This pilot edition of *Code Beats* program was organized as a three week online camp (note: due to COVID-19), with a one-hour class held each weekday. Classes were streamed to students using the Twitch[2] platform where the students could interact with each other using the platform's chat. Students gave feedback to instructors using the multiple choice and open ended questions presented during lecture using the online presentation platform Mentimeter[3], in addition to end-of-class interactive quizzes and after class help sessions. Each day students had an after-class programming assignment to reinforce the knowledge about the content learned so far. The students were strongly encouraged to upload their submissions each day, as assignments were reviewed daily to track students' progress.

To help with engagement, which can be challenging during online classes, the camp was conducted in a format similar to a TV news show. That is, the host introduced concepts, conducted live segments, and introduced pre-recorded, in-depth sessions. The daily show contained three repeating segments. First a live segment called **Tweets and Beats** where students' submissions from the previous day were discussed. This section motivated students to share their work, highlighted strong submissions, and offered constructive feedback. Later the **Music Theory Minute**, a segment included three times per week, featured a music professor (the third author)

[2]https://www.twitch.tv/
[3]https://www.mentimeter.com/

explaining the relevant theory and giving helpful, practical tips to making the students' daily activity sound authentic. Next there was the **Coding with Doug** segment, a daily section conducted by a Computer Science professor (the first author) with 4 years of experience in teaching computer programming for students of a variety of levels. During this section Doug explained the specific coding concepts and syntax within the Sonic Pi with a focus on readying the students for the day's assignment.

One of the concerns of running this camp online was whether students would attend for the complete three-week session. To encourage commitment, a fee of $25 per student was charged, with scholarships available.

5 RESEARCH QUESTIONS

The main purpose of this study was to determine whether using culturally relevant music to teach computer programming for students in middle school would affect their engagement and attitude towards Computer Science (CS). In order to validate our approach we investigated two research questions:

RQ1: How does the use of hip hop in teaching coding affect the engagement (i.e., enjoyment, confidence, belonging, intent to persist) of middle school students towards CS?

RQ2: Does our graded scaffolding approach enable students to progress to the **Create** phase of the **Use-Modify-Create** framework?

6 METHODS

6.1 Demographics

Because the camp was conducted online, recruiting was conducted via social media, with students registering by purchasing a one-month channel subscription ($25) on Twitch. Over the course of the three weeks an average of 45 students attended the *Code Beats* camp. From this group 31 students answered the pre and the post survey, which was not mandatory. From this group, 17 students had no previous exposure to *Code Beats*, and so these 17 students are the only students included in our questionnaire results. This population was, on average 12 years old (min: 10 - max: 14), and was 70.6% of male and 29.4% of female. 41.2% were self-declared minorities and 23.5% Black. 64.7% of the students declared that they knew how to read music and 41.2% declared that had previously taken a programming class.

6.2 Data Collection

The primary way we measured students' perception about Computer Science was by asking them to answer a survey at the beginning of the first day of class and at the beginning of the last day of class. The survey included 19 questions answered via a Likert scale, from 1 (Strongly Disagree) to 5 (Strongly Agree). During the class, students also provided live feedback via Mentimeter, a crowd interaction tool. To provide qualitative support for our quantitative data we performed an open coding process on these written responses, grouping by theme.

At the end of each day of class the students were asked to do an activity, designed to take 15 minutes to complete. They were encouraged (not required) to submit their daily activity from which a subset was used to provide constructive feedback during the next day's class. On the last day of camp we conducted a beat contest, where each student could submit an original hip hop beat created with what they had learned during the camp. They were informed about the contest on the first day of camp.

6.3 Data Analysis

To compute the statistical significance of our survey we have applied the Wilcoxon signed rank test. The Wilcoxon signed rank test is a non-parametric statistical procedure for comparing two samples that are paired or related. The Wilcoxon signed rank test does not require that the data is normalized [7]. With Wilcoxon signed rank test if p-value is lower than 0.05 (p-value < 0.05) we can say that we have statistically significant evidences to reject the null hypothesis (no changes in answers between the first and last survey).

The Wilcoxon signed rank test was applied to each question individually, but also to groups of categorized questions. For that, the questions in our survey were placed in four groups: computing enjoyment (5 questions); computing confidence (5 questions); identity and belonging in computing (5 questions); and intent to persist in computing (4 questions). Note that, because some answers were positively phrased and others negatively phrased the value of the negative statements were inverted prior to analysis. For example, a 5 (Strongly Agree) answer to the question "Computers are not for me" was converted to a 1 (Strongly Disagree) to be compatible with answers to the question "Computers are cool". Then the answers for the questions of each group for each respondent were added and the total for pre and the total for post was considered to calculate the p-value.

7 RESULTS

7.1 RQ1: Student Engagement

When questions were grouped, as described in Section 6, all four categories of questions showed a statistically significant difference (p-value < 0.05) between pre and post surveys, as shown in Table 1. Drilling down to individual questions, we show positively-phrased questions in Figure 2. As you can see, when moving from the pre survey (top) to the post survey (bottom) students' answers became more positive, as indicated by the increasing amount of *Agree* and *Strongly Agree* statements, shown in green. Eight of these twelve questions showed a statistically significant difference when analyzed individually (p-values to the right of each question).

Category	p-value
Computing enjoyment	0.006
Computing confidence	0.001
Identity and belonging in computing	0.011
Intent to persist in computing	0.022

Table 1: Differences between pre and post surveys

In Figure 3, we show only negatively-phrased questions, such as "I cannot be a computer scientist". When moving from the pre survey (top) to the post survey (bottom) students' answers changed, becoming more negative, as indicated by the increasing amount of *Disagree* and *Strongly Disagree*, shown in red. Three of these seven questions showed a statistically significant difference when analyzed individually.

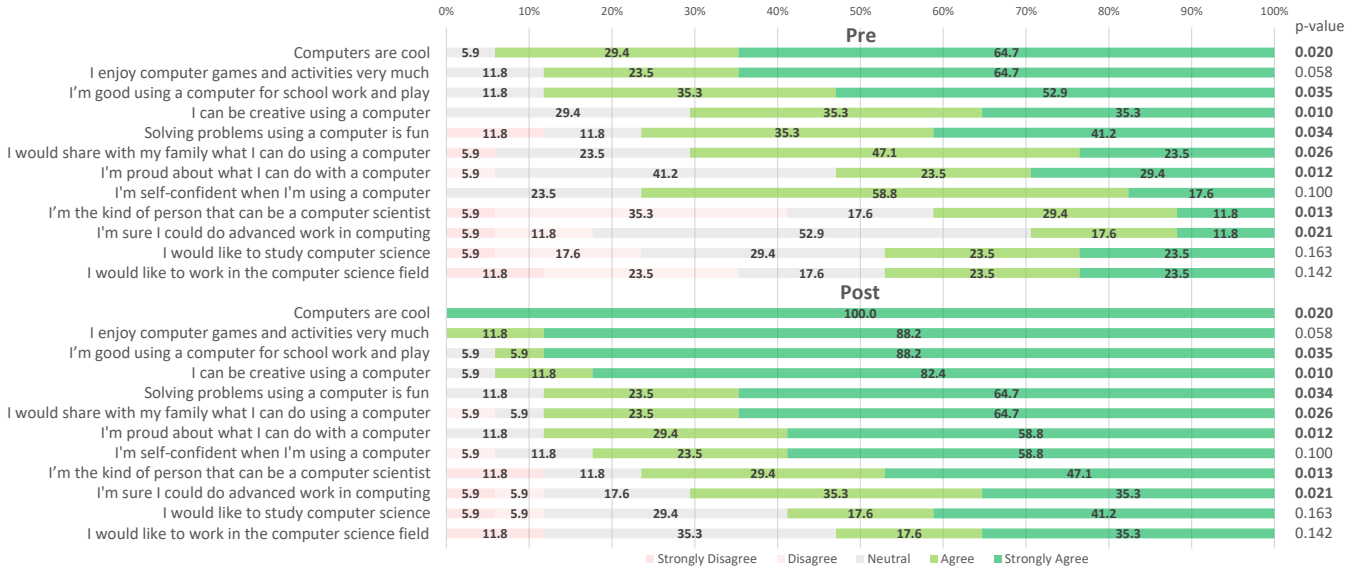

Figure 2: Survey Results: statements with a positive view of computer science

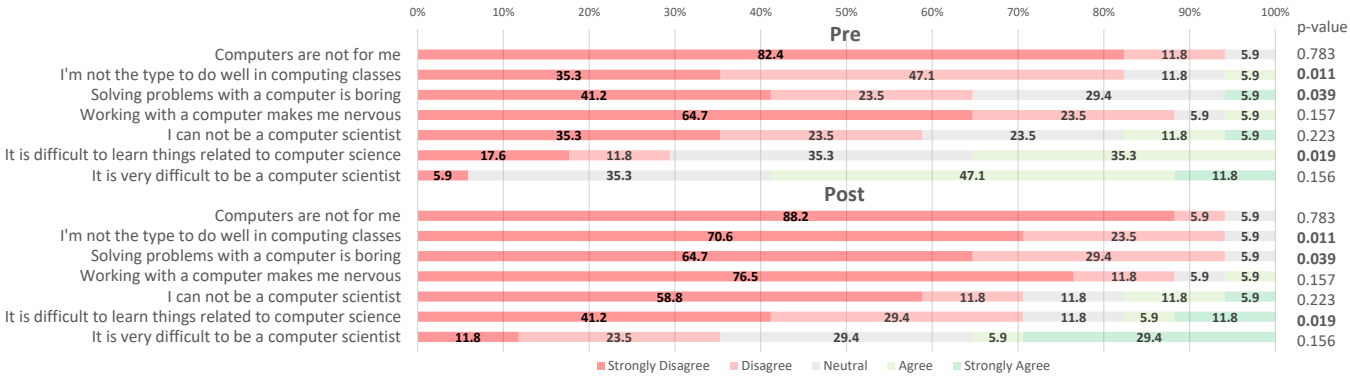

Figure 3: Survey Results: statements with a negative view of computer science

Further supporting this point, other data shows that students remained engaged throughout the three week camp. During the 3 weeks students were assigned 14 activities, where 7 were open-ended activity, including the beat for the contest. On average 10 students submitted an assignment even though it was not required, with student submitting a total of 140 assignments over the course.

When asked on the final day "Do you think you might take a programming course in school?" students had overwhelmingly positive answers such as "yes because i love it #CodeBeats forever", "Definitely!! I really enjoy coding and computer science, and I intend to pursue it", and "yes because i want to have a higher chance to get into the tech center". Even students that were less confident in their future were positive about their experience, saying "maybe because i definitely have fun doing this", "I might, because it is really fun", and "I think I could". Taking into account this qualitative data along with the above quantitative data, we can answer RQ1, noting that using hip hop to teach computing concepts does lead to an increase in engagement.

One potential caveat is that students may become more engaged with CS after taking any CS class. While this may be true, we believe that many of the students in our class would *not* have taken other CS classes. When asked, on the first day, why they signed up for our course many responses directly referred to hip hop and music, saying "I wanted to learn code and I love rap", "I'm very interested in beats like Dr. Dre and I saw this so I want to make or create beats", "I wanted to learn how to make music with code", and "For fun and also learning to code with music. Also I love music so I was like why not make it.".

7.2 RQ2: Developing Creators

If students can create an original beat by the end of the camp they may have achieved the highest level of learning in the Use-Modify-Create framework. To investigate this RQ we analyzed the beats

students submitted for open-ended assignments (i.e., assignments without pre-defined beats) over the course of the camp. From these submissions, we only considered original beats; we filtered any beats that were even partial copies from earlier assignments. As you can see from Figure 4, we have graphed the percentage of students that contributed an original beat over time. By Day 4 (the first submission of a open-ended activity) 76.5% of students contributed an original beat and by Day 14 100% submitted at least one beat. We prepared a playlist[4] of the beats submitted on Day 14, as we believe that their overall quality indicates students' relatively deep learning. Note that many students submitted more than one original beat, which we also show in Figure 4, with 29.4% of students submitting six original beats by Day 14.

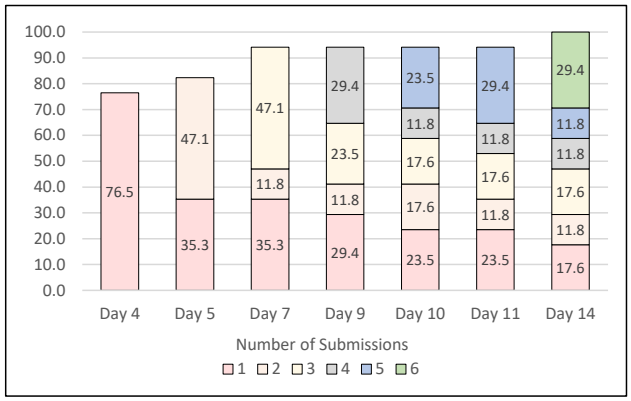

Figure 4: Number of original beats per student over time

8 CONCLUSIONS

Based on survey's results and beats submitted during *Code Beats* and for the contest we can say that the use of hip hop to engage students in computational thinking is promising. We note a statistical significant change in the engagement towards computer science, mainly when we compare the questions in a grouped way. Also, we can mention that the use of the Use-Modify-Create framework was successful as all of 17 students submitted at least one beat of their own creation.

Additionally, there are some points that can be highlighted: (a) The importance of beats contest: we noted a motivation from students to create and show their beats at the contest, it was noted during the classes via chat and with numbers with beats submitted; (b) The quality of beats: at the end of the third week the students were able to create their own beats, as we have some very goods beats, analyzing via musical perspective, and we had some very goods beats analyzing via the code perspective and the use of the Sonic Pi resources.

Despite the good results from this pilot camp, there are some limitations in our work: (a) The demography of our student was not the ideal, as we were planning to have more Black and Latino students, our target audience; (b) Sonic Pi has to be installed or be used as a standalone software, it might need some previous knowledge from students about how to do that. Also, there are some difference between platforms (i.e., Windows and Mac); (c) Due to COVID-19 outbreak this pilot camp was held online, it was initially planned to be held in person. It is initially a limitation, but also opened doors and new ways that were not being considered before.

Finally, as future work we are planning to have more editions of *Code Beats*, as the first results are encouraging, with a broader participation, mainly with Black and Latinos. For that we are planning a new version of *Code Beats* in collaboration with Richmond Public Schools where we can reach a higher population of Black and Latinos students. Also, we are studying a way to turn the curriculum and material from *Code Beats* available.

[4]https://bit.ly/CodeBeatsPlayListContest

ACKNOWLEDGMENTS

To participants from this edition of *Code Beats*, to Nickelus F that helped us motivating the participants and to Prof. Jean Rodrigo Adacheski that helped us with statistical analysis.

REFERENCES

[1] Samuel Aaron and Alan F. Blackwell. 2013. From Sonic Pi to Overtone: Creative Musical Experiences with Domain-Specific and Functional Languages. In *Proceedings of the First ACM SIGPLAN Workshop on Functional Art, Music, Modeling & Design* (Boston, Massachusetts, USA) *(FARM '13)*. Association for Computing Machinery, New York, NY, USA, 35–46. https://doi.org/10.1145/2505341.2505346

[2] Samuel Aaron, Alan F. Blackwell, and Pamela Burnard. 2016. The development of Sonic Pi and its use in educational partnerships: Co-creating pedagogies for learning computer programming. *Journal of Music, Technology and Education* 9, 1 (2016), 75–94. https://doi.org/doi:10.1386/jmte.9.1.75_1

[3] College Board. 2018. *AP Program Participation and Performance Data.* College Board. https://research.collegeboard.org/programs/ap/data/archived/ap-2018

[4] Jeff Chang and S Craig Watkins. 2007. It's a Hip-hop World. *Foreign Policy* 163 (2007).

[5] Code.org. 2020. Code.org Computer Science Principles Syllabus and Overview. Retrieved July 16, 2020 from https://code.org/files/CSPSyllabus2020.pdf

[6] Code.org and CSTA. 2018. 2018 State of Computer Science Education. *code.org* (2018). https://advocacy.code.org/

[7] G.W. Corder and D.I. Foreman. 2009. *Nonparametric Statistics for Non-Statisticians: A Step-by-Step Approach.* Wiley. https://books.google.com/books?id=-ufOfzVp6qYC

[8] Betsy DiSalvo, Mark Guzdial, Charles Meadows, Ken Perry, Tom McKlin, and Amy Bruckman. 2013. Workifying games: successfully engaging african american gamers with computer science. In *Proceeding of the 44th ACM technical symposium on Computer science education.* ACM, 317–322.

[9] Geneva Gay. 2013. Teaching to and through cultural diversity. *Curriculum inquiry* 43, 1 (2013), 48–70.

[10] Jamie Gorson, Nikita Patel, Elham Beheshti, Brian Magerko, and Michael Horn. 2017. TunePad: Computational Thinking Through Sound Composition. In *Proceedings of the 2017 Conference on Interaction Design and Children.* ACM, 484–489.

[11] Gloria Ladson-Billings. 1995. Toward a theory of culturally relevant pedagogy. *American educational research journal* 32, 3 (1995), 465–491.

[12] Irene Lee, Fred Martin, Jill Denner, Bob Coulter, Walter Allan, Jeri Erickson, Joyce Malyn-Smith, and Linda Werner. 2011. Computational Thinking for Youth in Practice. *ACM Inroads* 2, 1 (Feb. 2011), 32–37. https://doi.org/10.1145/1929887.1929902

[13] Matthew Lee and Maria R Lee. 2015. Beyond the wearable hype. *IT Professional* 17, 5 (2015), 59–61.

[14] Brian Magerko, Jason Freeman, Tom McKlin, Scott McCoid, Tom Jenkins, and Elise Livingston. 2013. Tackling Engagement in Computing with Computational Music Remixing. In *Proceeding of the 44th ACM Technical Symposium on Computer Science Education* (Denver, Colorado, USA) *(SIGCSE '13)*. Association for Computing Machinery, New York, NY, USA, 657–662. https://doi.org/10.1145/2445196.2445390

[15] Bill Manaris, Blake Stevens, and Andrew R Brown. 2016. JythonMusic: An environment for teaching algorithmic music composition, dynamic coding and musical performativity. *Journal of Music, Technology & Education* 9, 1 (2016), 33–56.

[16] Ernest Morrell and Jeffrey MR Duncan-Andrade. 2002. Promoting academic literacy with urban youth through engaging hip-hop culture. *English journal* (2002), 88–92.

[17] Engineering National Academies of Sciences and Medicine. 2018. *Assessing and responding to the growth of computer science undergraduate enrollments.* National Academies Press.

[18] President's Council of Advisors on Science and Technology (US). 2010. *Prepare and Inspire: K-12 Education in Science, Technology, Engineering, and Math (STEM) for America's Future: Executive Report.* Executive Office of the President, President's Council of Advisors on Science and Technology.

[19] U.S. Bureau of Labor Statistics. 2020. Computer and Information Technology Occupations. Retrieved July 13, 2020 from https://www.bls.gov/ooh/computer-and-information-technology/home.htm

[20] Dale Parsons and Patricia Haden. 2006. Parson's Programming Puzzles: A Fun and Effective Learning Tool for First Programming Courses. In *Proceedings of the 8th Australasian Conference on Computing Education - Volume 52* (Hobart, Australia) *(ACE '06)*. Australian Computer Society, Inc., AUS, 157–163.

[21] Linda J Sax, Hilary B Zimmerman, Jennifer M Blaney, Brit Toven-Lindsey, and Kathleen Lehman. 2017. Diversifying undergraduate computer science: The role of department chairs in promoting gender and racial diversity. *Journal of Women and Minorities in Science and Engineering* 23, 2 (2017).

[22] Robert H Tai, Christine Qi Liu, Adam V Maltese, and Xitao Fan. 2006. Planning early for careers in science. *Science* 312, 5777 (2006), 1143–1144.

[23] Jeannette Wing. 2006. Computational Thinking. *Commun. ACM* 49 (03 2006), 33–35.

[24] Luke Winkie. 2019. That Music You're Dancing To? It's Code. Retrieved July 6, 2020 from https://www.nytimes.com/2019/10/04/style/live-code-music.html