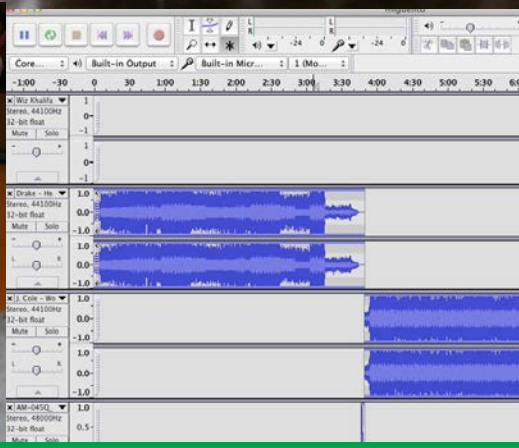
learning exchange



Spring 2012 Program Review

June 19, 2012

by Danielle Rothermel and Sam Birch

sponsored by the Dean of the College through Science Center Outreach

INTRODUCTION

Learning Exchange is an after-school program run by Brown University students. It is designed to excite Providence middle-school students about solving problems from the classroom and the real world through project based learning. We want to create a community of young people who are excited about their ideas and have the confidence and skills to implement them. In Spring 2012 we addressed this with a program in computer programming and a program in music mixing.

The goals of the program were as follows:

1. To **build mentoring relationships** between students at Brown and in Providence.
2. To **engage students in creating projects** and learning skills of which they are proud.
3. To **connect mathematical and analytical thinking** to these projects in order to make math more relevant to students.

The Spring 2012 program ran 4 sections with 25 students and 14 teachers. The program expanded to an additional middle school, although with the same number of sections. The focus of the curricula was moved from last semester's math-driven approach to a project-driven one. This approach was quite successful for computer programming and led to strong quantitative and qualitative improvement in math positivity.[1] It was significantly more difficult for music mixing, which will be replaced at least for the coming fall session.

PROGRAM STRUCTURE

During the spring of 2011 Learning Exchange expanded from Del Sesto (DS) middle school to additionally work with Nathanael Greene (NG) middle school. We also became a formal PASA provider at Del Sesto, instead of operating in the "Club AfterZone" time slot (PASA does not have an operation at Nathanael Greene).[2]

Students in the Computer Science program at Nathanael Greene.

At each middle school we ran a computer programming and a music mixing program for two hours a week each, for a total of 10 weeks. Sections either met Monday/Wednesday or Tuesday/Thursday for an hour each day. In the computer programming course we used MIT's Scratch environment to make animations and games, and in the music mixing course students used Aviary's Roc, Audacity, and Sony's ACID.

Each section varied in their attendance and in the number of students to complete the program:

NG computer programming (3 instructors):

6 regulars, 6 completions

NG music mixing (3 instructors):

5 regulars, 4 completions

DS computer programming (5 instructors):

5 regulars, 8 completions

DS music mixing (4 instructors):

7 regulars, 8 completions

[1] We did not measure concrete math skills as we had in previous assessments.

[2] We found that the formalization helped enormously with the attendance problems we faced last semester. We theorized that (1) because students signed up ahead of time instead of being recruited we got a more engaged group, and (2) taking attendance and being able to call parents made students take the programs more seriously.

There were a total of 25 students who completed the program, with one student completing both computer programming and music mixing portions. Two students at Del Sesto had taken a Learning Exchange program in the previous semester.

A student who returned for a second semester of Learning Exchange.

The instructor to student ratio varied between 1:1 and 1:2, varying with attendance as well as per section. We found this ratio extremely important as many students required one-on-one attention for progress and behavior. At Del Sesto we had an effective computer to student ratio of 1:1 (although sometimes the music mixing students worked in pairs). However, because both Nathanael Greene programs were run at the same time, both programs there had students operating in pairs with one laptop.

The 14 instructors[3] applied via online survey and were interviewed in person, with the exception of many who stayed with the program from the previous semester. We recruited new teachers via Brown morning mail and an information session in the fall, but for the most part they were found through previous teachers' recommendations. We did some simple training to establish standards and reviewed material from the previous semester's teacher training provided by Citizen Schools.

[3] 14 is not the sum of the number of instructors in each section because Andrew DiMarco taught Del Sesto computer programming *and* music mixing. Note that, with the exception of Andrew, every teacher taught in only one section.

Throughout the semester, each curriculum's group of instructors met twice per week to plan lessons for the next day, reflect, and coordinate. Planned activities and assignments were delegated to pairs of instructors and then were presented and refined in the meeting before they were used. We found these meetings extremely useful, especially for coordinating between the two different schools. In addition we had occasional all-hands meetings for reflection and planning with regard to the Learning Exchange as a whole.

This semester we ran a combined end-of-semester celebration for both curricula and schools, replete with ice cream. Students demonstrated their work from the semester to their parents as well as to Brown students and faculty.

CURRICULA

Computer Programming

Introduction

Computer science (CS) is a rapidly evolving field that allows users to express their creativity through the development of software. Part of the beauty of CS is that it often requires the computer scientist to understand and exercise mathematical reasoning. Doing math becomes more fun and rewarding because it is viewed as a concrete input into making a computer program.

The concept of using CS as a tool for learning math was inspired by the work of Project Bootstrap, an after-school organization that teaches math to middle schoolers by introducing them to video game development in a language called Scheme.[4] In Fall 2012 a strong emphasis was placed on teaching math concepts and then structuring programming projects around those concepts. We found that this approach caused students to lose interest quickly and did not put sufficient emphasis on learning programming concepts necessary for completing their projects.

[4] More information about Bootstrap is available at www.bootstrapworld.org.

Therefore, this semester we transitioned from a math-driven curriculum to a project-driven one. We implemented an entirely project-based curriculum and encouraged and taught math where it came up. Math was present in many projects, as we expected, but we hope to find ways to improve this further next year. We also placed a stronger emphasis on teaching key programming concepts.

Environment and Class Structure

Computer programming was taught again using MIT's Scratch software, which we found sufficient for our needs. Scratch provides a simple and graphical interface for programming games, animations, and interactives. The schedule of the program went approximately as follows:[5]

2 week introducing Scratch

2 weeks creating an animation

4 weeks creating a game

2 weeks working on final project

In order to introduce key programming concepts we used a combination of interactive activities and short worksheet-based projects. For example, to teach while-loops we gave the students candy and allowed them to eat the candy *while* the teacher's hand was raised. A similar approach was used to teach for-loops. Broadcasting, a method of flow control, was taught through a game similar to Simon Says.

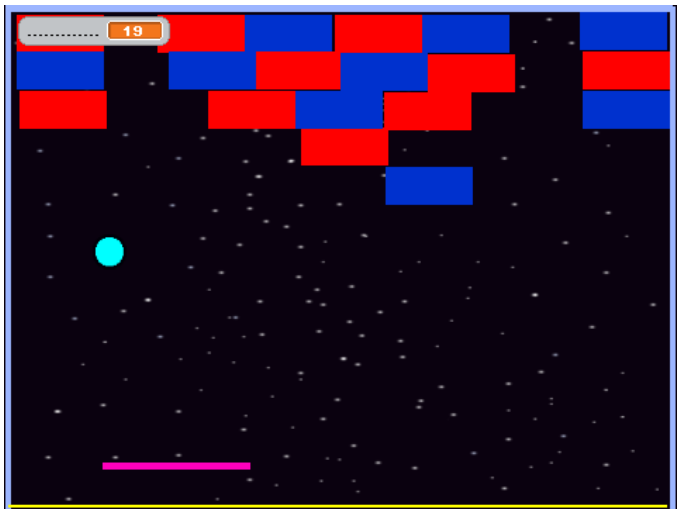
The concepts taught through interactive activities were combined and practiced in the short projects. These activities were intended to give a simple example of the type of project the students would next be completing. To prepare for animations, students filled in a Spongebob animation stencil, and to prepare for games they added many blocks (lines of code in Scratch) to complete a two player game based on the movie Up.

After finishing the introductory activities and mini-projects, the students worked on an original project. During the animations many coordinate-based math concepts arose as well as flow control programming questions. When programming their games, students encountered more complex math questions. They tackled concepts such as reflection angles and acceleration to allow the rebound of a ball and to make a jumping character appear more realistic.

After each project students were asked to present their creation to the class. They were responsible for explaining their motivation for the project, listing tricky problems they came upon and providing their solutions to these problems. For the final projects students were then encouraged to consider these reflections and to choose their favorite project to improve.

At the end of semester event we gave all of the computer programming students a thumb drive loaded with all their projects. In addition we provided copies of the Scratch software with instructions in the hope that they would use it after the program on their own (some students, and even their younger siblings began to work on projects at home mid-semester).

Student final projects ranged from an interactive music trivia quiz to a game similar to Temple Runner. Most student projects are available anonymously online.[6]



A brick breaker game created last semester and improved to be a final project this semester.

[5] Some students and the sections varied in the actual amount of time spent, but the schedule is representative of the mean.

[6] http://scratch.mit.edu/users/learningexchange

Challenges Faced

Some problems we encountered with this past semester's implementation of the computer programming curriculum were:

1. Although much better than the previous semester, sporadic attendance made a continuous narrative impossible for some students.
2. Some types of projects brought up very little math.
3. Some students resisted learning math even when it did come up.
4. Even by the end of the semester some students couldn't perform all basic programming tasks.
5. Not fully understanding even one or two basic concepts precludes independent creation, which necessitated very close work with many students.

Digital Sound Processing

Introduction

Music mixing is an integral part of the modern entertainment industry. We have found that students are extremely passionate about hip-hop music, and that we can harness this enthusiasm to teach them a useful skill which involves math and critical thinking.

In Fall 2011 we used Audacity as a platform because it is free and runs well on Windows machines. However we found that Audacity was very challenging to use and made even simple operations difficult, limiting the students' creativity. Audacity was also visually intimidating for the students. To solve these problems, this semester music mixing was taught with Roc, Audacity, and Sony's ACID.

Environment and Class Structure

Each of the three platforms were used in the music mixing program. Roc is a visually appealing and simple drum machine platform that is easy to learn and eased students into understanding music creation. Audacity was used for a very short period of time and served as a bridge between the simplicity of Roc and the complexity of ACID. ACID was the ultimate goal that allowed students to create high quality mixes. The schedule of instruction was:

2 weeks: Basics of beats with Roc[7]

2 weeks: Introducing ACID

6 weeks: Working on projects

Worksheets were used to teach students concepts such as beat-mapping and looping. Other concepts were taught through short discussions. Also, planning sheets were provided to encourage the students to set goals and stay on task and students created daily project logs to track their progress. About half the class time was spent on learning concepts necessary for mixing, and the other half of the time was spent on projects. Towards the end of the semester, however, worksheets and discussions were phased out and the vast majority of time was spent working on the mixes.

While many math concepts did come up naturally in the music mixing process in areas such as beat counting, pitch matching (ratios) and rates of change in terms of BPM, ACID disguised many of the math tie-ins. Therefore additional math problems were given to students as they learned the concepts to demonstrate math's relevance to the topic.

A student playing his mix for a fellow student and an instructor.

[7] As we weren't able to get internet at NGMS until later in the semester, students there used Audacity.

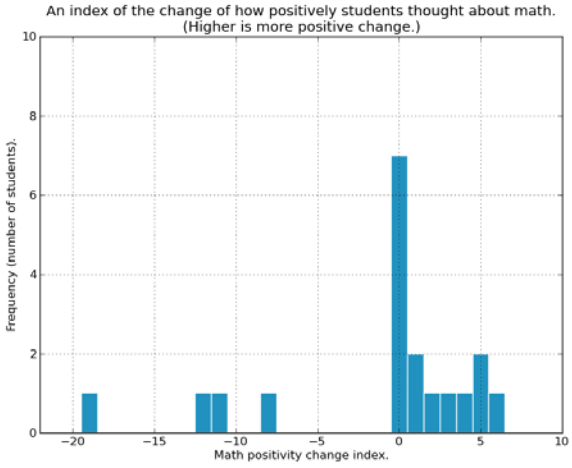
The Future of Digital Sound Processing

The music mixing curriculum posed many challenges this past semester, including most of the general problems encountered in computer programming. One stood out, however: after trying Garageband, Roc, Audacity and ACID, no environment was suitable for our needs. The platforms either don't have a viable connection to math, or are simply too complicated (for middle-schoolers *and* instructors).

After a long discussion with the music mixing instructors, and subsequently a discussion with the organization at large, the decision was made to not offer music mixing for the coming fall. The implications of this will be further discussed in the *Future of Learning Exchange* section.

OUTCOMES

19 of 25 students took a before/after assessment tailored to the curriculum. We found, however, that student responses were not entirely reliable as some students did not take the surveys seriously. This manifests itself in a few outliers.



The math positivity change index is the sum of the increase of student self-evaluation on 9 different questions on a Likert scale mapped 1 - 5 (some negated), e.g.: "I can do cool things with math," and "I don't think I will learn anything useful in math class" (negated).

We created a composite index over 9 questions to give a more stable assessment of how students felt about math. The difference between the post-assessments and pre-assessments is shown in Fig.1. Results are modestly positive: 8 students felt more positive about math afterwards, 4 felt worse, and 7 didn't change at all.[8]

Regarding the program, of the 19 students with data:

- 18 had "lots of fun."
- 16 learned "a lot."

Common themes in the open feedback section were for "better computers" and "food."

Computer Programming

The spring computer programming course seemed more successful than last semester's offering. The project-driven curriculum gave students more time to focus on projects, and starting with an animation project served as a nice segue to the more technically challenging task of implementing a game. A number of students started using Scratch at home outside of the program, and one even got his younger brother to start using it. Feedback from students and parents during the end of semester event was very positive as well -- one student took it upon herself to speak to the audience of 50 about how the program improved her attitude towards math and learning.

Instructors for the computer science curriculum thought that the switch to a project-based curriculum was successful as well. Math was less forced, and instructors enjoyed teaching math in a more personal and relevant way. The emphasis on programming made students more independent, although many students were not independent enough to work without frequent intervention.

Music Mixing

Music mixing was more successful as well, albeit still frustrating. Not having a satisfying

[8] Quantitative results for computer programming and music mixing were almost exactly the same, so the graph is presented in aggregate.

platform as well as a spate of technical difficulties with the platform distracted from teaching. Students enjoyed the simplicity of Roc, and ACID was something of an improvement over Audacity, although any gains were muted somewhat by its extremely complicated interface. Nonetheless, many students were proud to show off their work and their skills at the end-of-the-semester event.

A combination of students from Del Sesto and Nathanael Greene gathered to show off their work.

Next Steps

Re-evaluating Our Mission

Many of the problems that came up this semester centered on the ability to teach math concepts while maintaining students' interest. For many students even mentioning math means immediate disengagement. Therefore for these students we need to find a balance between teaching math in an interesting way and teaching math as it is used in the classroom. We've tried to address this problem by closely examining our mission.

A broader area than "classroom math" we've considered is problem solving, which arises in every area of life, but especially in mathematical and analytical thinking. Even more important, we think that problem solving is the link between classroom math and its useful application, leading to relevance and motivation. We have decided to keep the project-based approach and to continue to look for natural math tie-ins, but to widen the focus to teaching problem solving skills. For many students who have an interest in math we'll continue integrate math to the extent that it's appropriate. But for those who can't conscience it, we hope that we might make its application more real, or at the least equip them with some broader analytical skills.

We've refined these thoughts and conversations into a new set of objectives for the coming semester:

1. To help students experience and employ real-world applications of the math skills they learn in school in order to connect their knowledge to broader life and career goals.
2. To empower students to take ownership of their education through a sustained, self-designed project that utilizes these skills and expertise;
3. To build mentoring relationships between students at Brown and in Providence.

The Future of Learning Exchange

Considering how best to meet these objectives, we've suspended the music program until a better platform can be found. A group of Brown students and Learning Exchange instructors has begun a project to create a platform that would provide the ease of use necessary for an introductory course while still requiring relevant critical thinking and math skills. Though there is hope that this will be a long term solution, in the intermediate time the music program will not be offered.

In place of the music program a new program is being created centering on design. Basic design concepts are essential to many aspects of life and provide an excellent framework from which to teach problem solving. The design curriculum will be developed over the summertime and will be unveiled Fall 2012.

Our final major emphasis of this next semester is on planning and teacher training. Math and problem solving concepts come up naturally in both computer programming and design, but these lessons can be taught more effectively if there is substantial

forethought about the best ways to teach these concepts. Projects will be planned to require these math and problem solving skills, and solutions to problems will be taught in a way that not only solves the current issue but provides a tool-set to solve any issue.

To achieve these goals, the curriculum must be very clear and well thought out, and the teachers must have a thorough understanding of both the material they are teaching and of exactly what principles are most important to convey. We have always tried to plan ahead, but our foresight has usually been trumped by dramatic shifts due to practical constraints and differences during implementation. We're hoping that we can anticipate many of the major differences ahead of time (as we did to some extent after learning in the pilot and the fall semester).

We are excited about the direction that Learning Exchange is taking and look forward to learning from our mistakes and pursuing the many aspects that have made this semester successful.

Special Thanks

We would like to thank the following people for helping us create and strengthen Learning Exchange:

- Amit and Nabeel, the founders and previous coordinators of Learning Exchange who made Learning Exchange what it is today.
- Our Spring 2012 teachers and curriculum developers: Amit Jain, Ryan McVerry, Liz Neu, Paul Kernfeld, Natan Last, Andrew DiMarco, Danielle Rothermel, Nabeel Gillani, Zach Korol-Gold, Diego Morales, Andy Chen, Sandra Yan, Sam Birch, Shweta Bhatt, Hannah Rosen, and Emily Cacciatore.
- Previous Learning Exchange volunteers
- Our Citizen Schools partners: Emily Stainer for her teacher training, Philip Parham for his recruitment support
- Our advisors: Dean Oludurotimi Adetunji and Jo Browne from the Science Center
- Administrative support: Jodie Gill from the Science Center and Elizabeth Malone from the DOC
- Our PASA sponsors: Camely Machado from the John Hope Settlement House and coordinator of DelSesto's PASA program, Alison Quinn, Co-Director of PASA
- Our Nathaniel Greene Coordinator: Ms. Maia
- Emannuel Schanzer, Founder of Project Bootstrap
- Karen Brennan and Sayamindu Dasgupta of the Lifelong Kindergarten group at the MIT Media Lab
- Other professors and advisors at Brown, namely: Shriram Krishnamurthi, Chad Jenkins, Matt Harrison, Alan Harlam, Alan Flam, and Stephen Spoletto.
- Most of all, our students who make Learning Exchange worthwhile

THANK YOU!