Development and Testing of a Curriculum for an Introductory Course for Makerspace in High School

A thesis submitted in fulfillment of the requirement
for the degree of Bachelor of Science in
Physics from the College of William and Mary in Virginia,

by

Seamus H. Herriman

Professor Wouter Deconinck
Advisor

Professor Irina Novikova
Senior Research Coordinator

Williamsburg, Virginia
May 2018

Contents

Acknowledgments

I would like to thank my family and friends for their support and encouragement through the process of developing this curriculum. I would also like to thank the students and teacher whose feedback helped me to improve the curriculum to its current form. Finally, I would like to thank Professor Deconinck for his support and direction through the development process.

Abstract

I developed a high school curriculum to introduce students to Makerspace. The curriculum was developed as a series of modules focused on different elements, or tools, within a Makerspace. To begin development, I first identified my goals for what students should accomplish over the course of the curriculum in general, as well as for what they should accomplish over the course of each module. Modules were created through researching the element in question and configuring a series of lessons and tasks that would accomplish my goals for the module overall, with specific goals for each lesson. With the completion of each module, I, with the help of my advisor, organized a trial for the module, where I taught key elements of the course to students at the College of William and Mary. I requested feedback on the course and on a section of the curriculum that I was not able to include in the course and used this feedback to edit the curriculum. I sent this revised version to high school science teachers for additional feedback, which I also applied to the curriculum.

Chapter 1

Introduction

The goal of my research was to develop a curriculum to introduce high school students to Makerspace. While Makerspace is typically less structured than other high school academics in order to allow for creativity and innovation on students' part, the lack of direction or formal education in using different Makerspace elements may turn prospective students away from taking advantage of the Makerspace. My curriculum is designed to help students who wish to take it to overcome the initial learning curve and subsequently take part in the exploration and innovation that a Makerspace offers. By making the Makerspace more accessible to a broader range of students, this curriculum makes a Makerspace a more worthwhile investment for schools and helps to de-mystify science, technology, and engineering in the eyes of students who may be aware of opportunities in these fields but may not see any obvious path to taking advantage of such opportunities.

While most high school science curricula already include a lab element, this curriculum approaches the laboratory in a different way than standard high school lab curricula. Most often, lab curricula are designed to have students perform experiments done by past scientists that were fundamental in defining the laws and theories that govern the natural world. While such experience is important for connecting in-class abstract lecture material to physical reality and developing an understanding of the scientific method, it does not address practical usage of existing knowledge and tools for future development and application, nor does it offer students the opportunity to experiment more freely with the tools at their disposal or encourage them to do so themselves. My curriculum is designed to leave students with the practical experience of using tools and knowledge that are available to them to creatively solve real world problems and needs. This is an area that is rarely addressed in standard science curricula, though it is readily apparent in industrial usage of science.

Chapter 2

Theory

2.1 Educational Theory

I based my curriculum on the educational theories of active learning and Sternberg's three ability-based learning styles. I discuss these theories below.

2.1.1 Active Learning Curriculum

In most high school classes, learning is carried out through the medium of a teacher lecturing while students take notes. Exceptions to this may include English classes where there is more writing and discussion. While this style of learning imparts a large amount of information in a reasonably short period of time, when that information is techniques to be carried out rather than facts to be recited in an assessment, a lecture style of class can be ineffective. Research [5] on the knowledge acquisition between an active learning curriculum and a standard lecture curriculum found that students in a medical school retained more information through the active learning curriculum. Further research [4] on active learning at the undergraduate level found large impacts on students performance in science, technology, and engineering, and math courses when taught via active learning instead of passive learning curricula.

Finally, research [6] conducted at the high school level on active learning with biology courses demonstrated increased knowledge acquisition with active as opposed to passive learning.

On account of these findings, I argue that active learning will also produce better learning outcomes in the Makerspace, where the most important concepts for students to master is the application and combination of techniques which are taught in class. With this in mind, I designed the curriculum so that students would practice the material as they were taught it. This material also builds, so that as they learn new material, they are consistently using previous techniques. In this way, the students are more easily able to master the entire system as a whole than they would if they were simply given lectures about how the techniques should be carried out in theory.

2.1.2 Sternberg's Ability-Based Learning Styles

In many high school classes, a premium is placed on a student's ability to analyze the material presented to them in order to recognize underlying trends and factors. While these skills are certainly important, schools currently focus on them to the detriment of other skills. Sternberg describes three ways of thinking or abilities that are important for students to exercise. These abilities are analytical thinking, creative thinking, and practical thinking [9]. Within this theory, analytical thinking refers to one's ability to "analyze, critique,... [and] evaluate information". Creative thinking refers to one's abilities to "create, invent, discover, imagine, suppose, and predict." Practical thinking refers to one's ability to "implement ... and render practical what they know." In his 2008 paper, Sternberg gives examples of questions from different subjects meant to promote each of the three abilities. Examples from the paper are given below:

"Analytical:

(a) Analyze the development of the character of Heathcliff in Wuthering Heights. (Literature)

(b) Critique the design of the experiment. (Biology)...

Creative:

(a) Create an alternative ending to the short story you just read that presents a different way things might have gone for the main characters in the story. (Literature)...

(c) Discover the fundamental physical principle that underlies all of the following problems, each of which differs from the others in the "surface structure" of the problem, but not in its "deep structure ". (Physics)...

Practical:

(a) Apply the formula for computing compound interest to a problem people are likely to face when planning for retirement. (Economics, Math)

(b) Use your knowledge of German to greet a new acquaintance in Berlin. (German)"

As one can see, most high school curriculum focuses primarily on one's ability to analyze, while also touching on one's practical abilities. This is especially true in science and math classes. To correct for this disparity, my curriculum focuses on a mixture of creative and practical abilities, calling on students to use the tools they are given to creatively innovate and apply their abilities to solve real-world problems, recognizing how tools can be used in daily life and approaching issues with an imaginative mindset in order to circumvent them. This is so designed to help students to develop in these areas of thinking to a degree which is not commonly found in STEM courses.

Chapter 3

Methodology

I created my curriculum in the form of individual modules aimed to familiarize students with different tools in the Makerspace and direct them towards creative and practical applications of these tools. In the sections below, I discuss my reasons for choosing the tools on which I base my modules, the design of each module, and the testing and improvement process to arrive at the final version of each module. Through these modules, the curriculum also aligns with Common Core Standards of Learning for Science, where those standards recognize the importance of "applications of science among technology, engineering, and mathematics", which are the focus of this curriculum. [8]

3.1 3-D Printing

The first module of the Makerspace curriculum centers around 3-D printing. I chose 3-D printing for the first module due to its widespread popularity, both amongst people with scientific backgrounds and, more importantly, people without scientific background. A major part of the curriculum's purpose is to attract students who otherwise might be intimidated by the knowledge necessary to work effectively in a Makerspace, and one way to do this is by offering training in those tools that people are most interested to use. For this reason, 3-D printing appeared to be a good starting point. To carry out the goals of the curriculum in this module, it is based on three objectives, which form the bases of its three stages, which introduce students to tools through active learning and give them space to work with said tools using practical and creative thinking. These goals are given below:

Objectives:

1. Learn about functionality and operation of 3-D printer.

2. Learn to use and develop familiarity with 3-D design software.

3. Work as a team to design and create 3-D printed solution to problem.

3.1.1 Module Explanation

The 3-D printing module is built with the structure of three stages centered around three objectives, as given above. The first two of these stages give students information about the printer itself and teaches them the tools to use it through active learning.

The first objective is to learn to operate a 3-D printer and a 3-D print preparation program, in this case the Ultimaker 2+ [11] and its sister program Cura [12]. The Ultimaker 2+ was chosen for its market popularity and its accessibility through the Small Hall Makerspace. The first stage achieves this aim by directing students through each physical piece of the 3-D printer and explaining its operations in relation to the printer as a whole. The stage continues as the teacher guides the students in how to upload STL files of 3-D prints to Cura and in how to edit 3-D models in Cura through movement, rotation, and scaling. The stage ends by connecting Cura and the Ultimaker with the teacher leading students through the process of saving their models in Cura in an Ultimaker 2+-compatible form and submitting the file to the Ultimaker to print.

The second objective is to learn to use 3-D design software. For the purposes of this course, I chose to work with Google SketchUp Make [13] 3-D design software because it was free for educators and students and could be learned quickly while offering high versatility. To accomplish this goal, in stage 2 the instructor builds 3-D shapes while students follow along, learning the major ways in which 3-D objects can be created by actively carrying out the processes alongside the teacher. While there are many ways to create objects in SketchUp, my curriculum focuses on the use of the Push-Pull tool, the Arc tool, the Follow Me tool, and groups and components. These tools, when combined, offer students the potential to create diverse and intricate objects for printing.

The third objective is to work as a team to identify a problem that could be solved with 3-D printing and design a 3-D printed solution to that problem. The purpose of this objective was to offer students an opportunity to find practical applications of their newfound knowledge and skills, to think creatively about how to apply those skills, and to demonstrate group cooperation skills and success in completing the first two objectives. This stage is structured into periods for brainstorming, designing, prototyping, testing, and re-designing. With these structured periods, students get a sense of direction for achieving their overall goal and of how long different parts of the overall process ought to take. The project ends with a presentation that offers students the chance demonstrate their proficiency with the material and their ability to use their skills in effective and innovative ways. This also allows teachers to offer students feedback on their progress through the course.

3.1.2 Module Testing

After completing the module, I tested it through several methods.

I first taught the curriculum over the course of 3 1-hour sessions that went through the materials of Stages 1 and 2 to undergraduate non-physics major college students who expressed interest. After each session, I sent out a link to a survey by which the students were able to give feedback, and used the students' responses to refine and improve my curriculum. The main critique of the class that I received was that the students felt they did not receive enough time to learn to effectively use the tools to which they had been introduced. In response to this, I went through the curriculum and increased stage lengths accordingly.

As a secondary step, I emailed the module to high school science teachers, teachers in the James City County school district and teachers at my alma mater, St. Joseph's College Preparatory School in Philadelphia, PA. The main suggestions I received here were again to increase the time allocated to each stage, to provide more structure for the final stage of 'Team Projects', and to provide a route for teachers to give students feedback on their work. These suggestions led me to again increase stage lengths, and to significantly increase the guidance for students in fulfilling stage 3 through the use of the periods, and to implement the presentation as an assessment.

Unfortunately, I was unable to include stage 3 in the testing classes because the nature of the team projects would require significantly more dedication from attendees than could be expected from busy college students. While I was able to fit the content of the first two stages into a 3-hour time span, this was partly due to the fact that I had a very small class size that significantly expedited these stages because I was able to work with students more on a one-on-one basis. The team projects of stage 3, however, are designed to have students work and problem-solve together, with teacher input only when necessary, so that the expected time would not be shortened by the small group size. To compensate for this, I emailed the participants of the class the lesson plan for stage 3 and requested comments and critiques, to get a sense of how students felt as opposed to how educators felt about what students would be asked to do.

One student suggested that the time given to brainstorm a problem/need might not be enough to research a manageable problem/need. This led me to increase the time available for this step of the project process. The student was also curious about how a team of four would work on one model. This led me to stipulate more clearly in the curriculum that projects may involve multiple pieces built separately and combined after printing. Furthermore, throughout the design and model-creation process, a mixture of practical and creative thinking is necessary to turn the tools at hand into one's desired reality. There are times when this may require multiple people collaborating on how to most effectively achieve the group goal, and being a part of this process does not require always sitting at the monitor. Another suggestion was to change the section on identifying criteria and constraints so that it is a private discussion between each group and teacher, rather than something presented to the class. I made this change because it makes more sense at that stage of the project to give instruction privately, because the teacher may have to re-direct the group to improve their chances of success. A final suggestion that I received and was able to make change on account of was with respect to the time given for presentations. Here, I made clear in the curriculum that the time allotted was for students to prepare to present, because the time for the actual presentations would depend on the number of groups in a class.

3.2 App Development

I centered the second module of the Makerspace curriculum on app development. I chose app development for the second module due to its capacity to operate as an entry point for students into programming, within a framework that would be accessible for beginning students. With the ubiquity of smart phones, students have experience with and can connect to apps. In this way, offering students the ability to begin building their own apps, simple though they may be, will open students to the practical applications and creative possibilities of the course content. Moreover, it will show students their future potential to be not only users of technology, but producers and developers as well. Furthermore, by using OutSystems [7], with its capacity to utilize developer-made programming, the module's lessons will remain useful as students grow in ability, until they are ready for high-code platforms such as the Android Integrated Development Environment. For this reason, app development appeared to have useful characteristics to fulfill the goals of the curriculum and be useful to students' futures. Through OutSystems, students can take advantage of tools available to compose the projects that they envision, allowing them to make full use of both practical and creative thinking.

To give students the tool kit to successfully carry out this development on their own, the module has three objectives, which form the basis of each stage, preparing students through active learning to develop practical and creative thinking through the act of app development. These objectives are given below:

Objectives:

1. Introduce students to OutSystems.

2. Gain familiarity with OutSystems mechanisms and learn process of app development.

3. Have students create own apps in pairs.

3.2.1 Module Explanation

The app development module is built similarly to the 3-D printing module, with the structure of three stages centered around the three objectives. Here, the first two stages introduce students to the OutSystems platform and engender in students a framework for approaching the process of app development.

The first objective is to introduce students to the low-code app development platform OutSystems and its capabilities. OutSystems was chosen for its combination of a developer-side graphical user interface, which helps beginner developers to see and understand what is going on, and its facilitation of more advanced programming via the option to employ pure JavaScript code into internal processes. This graphical user interface allows the developer to more easily see what they are creating and how parts fit together, and allows them to do so without extensive coding. The first stage achieves this aim with the teacher walking students through the OutSystems' tutorial. While the tutorial can be accomplished very quickly, the instructor leads students through the material at a much slower pace, highlighting the implications and uses of different elements of OutSystems' operations, and pointing out the possible uses and consequences of using alternative routes that are available within OutSystems but are not used by the tutorial. Through this method of having students carry out the tutorial themselves on their own computers while the teacher emphasizes various options within the software, students can practice practical thinking for how different options can be used to accomplish different goals in the future.

The second objective is to become more familiar with OutSystems, and more importantly, to guide students through the process of app development, of breaking an app concept down into manageable, programmable pieces. To accomplish this goal, in stage 2 the instructor takes an app concept and, together with the students, breaks down the concept into pieces and program those pieces. This exercise gives students a sense of what building an app is like in reality, and shows them how to go about debugging faulty programming, in the cases when the suggestions offered by students and the requirements of OutSystems protocols result in errors that prevent successful compiling of the app. Thus, this stage highlights the active learning and practical thinking robustly as students work alongside the teacher to bring about a working app.

The third objective is to work as a team to identify a problem that could be addressed with an app, and then design and build an app solution to that problem. Stage 3 consists of a project in which students carry out this objective. This project is designed, similarly to the last module, in a series of steps, guiding students through the process of brainstorming and researching a problem/need, designing, testing, and redesigning. The project ends with a presentation so that students may expound on their development process and receive feedback from their teacher. The project also allows students to demonstrate group cooperation skills and success in completing the first two objectives. Through this project, stage 3 offers students an opportunity to innovate and imagine diverse solutions to problems and apply their knowledge to bring those solutions to completion.

3.2.2 Module Testing

After I completed the app development module, I again tested it by through a short course for undergraduate non-physics major college students and by emailing it to high school teachers for feedback, and emailing the stage 3 lesson plan to students for feedback.

In the short course for undergraduate students, I taught the app development process of stage 2, giving students information on different aspects of OutSystems from stage 1 where those aspects would be applicable to students' future attempts at app development. After going through the process of developing the BirthdayToWeekday app which is described in stage 2, students' main concern was that they were unable to type the JavaScript code into the window and listen effectively to the breakdown of how the code operated at the same time. Because of this, they suggested having the code in a separate document available to students online, from which they could copy the code and paste it into the window, giving them more freedom to listen to and learn from the explanation of the JavaScript code. I acted on this suggestion and included the new document with the module.

After emailing my curriculum to high school teachers for feedback, I received a few suggestions for the curriculum, but none that I was able to implement in the time available. I discuss these suggestions in the Future Research section of the Conclusion.

Finally, I emailed the lesson plan for stage 3 to students, as I had done before with the 3-D printing module, in order to obtain feedback on material that I did not have the capability of testing during the class due to time constraints and students' availabilities. Part of the feedback I obtained from students for stage 3 of the app development module was similar to the feedback I had received for the 3-D printing module, with the student suggesting a longer period of time offered to brainstorm a problem/need for which to develop a solution, and asking to the feasibility of two students programming one app. This led me to increase the time allotted for researching and identifying a problem/need, and to more explicitly specify in the lesson plan the expectation for students to work collaboratively together to use the methods available in the programming platform to innovate and in so doing attain a solution to their initial problem, even if only one student can physically type at the computer at a time. Another suggestion was to make the 'identification of criteria and constraints' section of the project more specific, so that groups know what is important to identify going into the development phase. The suggested specifications included what the app's input would be, what the app's output would be, and what options within OutSystems might be used to achieve these goals. I added this detail into the lesson plan because it would help to direct the work of the groups to be more effective. I also received a suggestion, like one for the previous module, to change the section on identifying criteria and constraints so that it is a private discussion between each group and teacher, rather than something presented to the class. I made this change because it makes more sense at that stage of the project to give instruction privately, because the teacher may have to re-direct the group to improve their chances of success. A final suggestion that I received and was able to make change on account of was with respect to the time given for presentations. Here, I again made clear in the curriculum that the time allotted was for students to prepare to present, because the time for the actual presentations would depend on the number of groups in a class.

3.3 Arduino Circuitry

I based the last module of the Makerspace curriculum around Arduino circuitry. I chose Arduinos for the final module for their integration of hardware, with the Arduino main circuit and its components, and software, with the Arduino Integrated Development Environment (IDE) used to program the circuit. This offers students a chance to see into the worlds of both electrical engineering and computer science, allowing students to learn about future career options. Also, while Arduino can be operated on a very basic level, it also offers space to make more elaborate circuits, allowing students to take the knowledge learned within the course and expand on it within the same framework. By introducing students to the basics of programming with Arduino and basic circuit elements and their possible uses, the module gives students the chance to engage in creative and practical thinking. For this reason, Arduino circuitry showed to have useful characteristics to fulfill the goals of the curriculum and be useful to students' future career planning.

In the compiling of this module, I drew largely from the work of Linda Barton [1], who created a short college-level lesson in Arduino circuits.

To prepare students to engage in development with Arduino circuitry, I developed the module with three stages centered on three objectives, educating students through active learning to develop practical and creative thinking skills through Arduino circuitry design and programming. These objectives are given below:

<u>Objectives</u>:

1. Introduce students to Arduino Board, Arduino Integrated Development Environment (IDE), and various Arduino components.

2. Teach students rules and limits to safely work with Arduino electronics and begin practicing basic circuits.

3. Have students develop their own Arduino circuits with personalized coding.

3.3.1 Module Explanation

The Arduino module is built similarly to the first two modules, with the structure of three stages centered around three objectives, with the first two stages preparing students for the work of stage 3 via active learning.

The first objective is to introduce students to the Arduino Uno Rev 3 board, the Arduino IDE, Arduino components, and the breadboard. The Arduino Uno Rev 3 board was chosen as the main board due to its use in the Arduino college lesson developed by Barton [1]. The objective is achieved in stage 1 with the teacher guiding students through the parts of the Arduino board and their functions in circuit development. The instructor then has students look through various components and their respective data sheets, stressing the importance of taking into account component limitations to insure the creation of circuits that do not damage Arduino boards or components. Next, the teacher introduces the breadboard and explains how it is used to build full circuits. Finally, the stage ends as the teacher shows students how to use the Arduino IDE to develop code to control circuits and how to upload that code to a circuit.

The second objective is to learn how a number of components can be used in circuits, gain practice building working circuits safely, and become familiar with the writing of functional sketches. To accomplish this goal, in stage 2 the teacher leads students through several circuits, challenging students to think about why different components are necessary, how the code works to bring about the resulting circuit activity, and what purposes different component actions could be used to accomplish. In this way, while students engage in active learning through circuit building, the teacher reinforces practical thinking, helping students to think about ways in which they might use circuit components or Arduino code for practical purposes, preparing them for stage 3. While there are many components available to use with Arduino boards, the circuits provided in Barton's lessons [1], which the stage uses, focus on LEDs, switches, piezo buzzers, transistors, light sensors, and relays. This set of initial components gives students a glimpse into the many and varied possibilities that are available with Arduino circuits.

The third objective is, like that of the first two modules, to work as a team to identify a problem that could be solved with an Arduino circuit and to design and build that circuit. Like previous modules, stage 3 uses a team project to carry out this objective. This project is designed in a series of steps, guiding students through the process of brainstorming and researching a problem/need, designing, testing, re-designing, making a manual, and presenting. The making of the manual gives students experience codifying their work as developers so that users of their circuit understand its function and purpose. The presentation at the end allows students to both describe their thought process in the development of the circuit with respect to the problem that the circuit addresses and receive feedback from their teacher. The project also allows students to demonstrate group cooperation skills and success in completing the first two objectives.

3.3.2 Module Testing

After completing the Arduino module, I again tested it via a short class for undergraduate non-physics major students, emailing the curriculum to high school teachers, and emailing the third stage lesson plan to students.

Over 3 1-hour classes, I taught students the contents of stages 1 and 2 of the Arduino module. Some students were not able to come for all of the days and missed the first day. This led to a staggered class where some students would be working on producing various circuits from stage 2 of the curriculum, receiving my assistance when needed, while I introduced others to the basics that make up stage 1, preparing them to begin building circuits of their own. I made this arrangement possible by creating a Google Doc of circuit diagrams and Arduino code for each circuit, so that students had a basis to work off of for each circuit and only needed to get help from me when they were confused or when the Arduino system was not working as expected. This system seemed to work well, and students appreciated the chance to work at their own pace through the material. Unfortunately, in later feedback, some students did not think the Google Doc was the best format to present code; unfortunately, they did not suggest better options. Another complaint was that a circuit's code as presented in the document only showed code that was new with respect to previous circuits, rather than the full code, leading some to confusion about the section of code in which the new code belonged. I corrected for this lack on code completeness in the Google Doc. Another matter that I recognized while teaching the course was the time that students took to perform each activity, especially to create and work with each circuit. These activities took significantly longer than I expected, and I increased the stage lengths of stages 2 and 3 accordingly.

In emailing teachers, one suggestion that I received was to include in stage 3 the creation of a manual, where students describe the purpose of their circuits, the way to use the circuit to achieve this circuit, and the circuit's limitations to performing this purpose as expected. I added this requirement into the end of stage 3 before the final presentation.

After emailing the lesson plan for stage 3 to students, I received a number of suggestions. The first suggestion was to make the 'identification of criteria and constraints' section of the project more specific, so that groups know what is important to identify going into the development phase. The suggested specifications included what users of the circuit will input into the circuit, what output the circuit will produce, and what circuit components the group will use to make the circuit. I added this detail into the lesson plan, because it would help to focus the efforts of groups to productive work at this stage. Another suggestion which I received was to change the section on identifying criteria and constraints so that it is a private discussion between each group and teacher, rather than something presented to the class. I made this change because it makes more sense at that stage of the project to give instruction privately, because the teacher may have to re-direct the group to improve their chances of success. A final suggestion that I received and was able to make change on account of was with respect to the time given for presentations. Here, I made clear in the curriculum that the time allotted was for students to prepare to present, because the time for the actual presentations would depend on the number of groups in a class.

Chapter 4

Conclusions

4.1 Conclusions and Future Plans

4.1.1 Conclusions

My research succeeded in the creation of an introductory high school Makerspace curriculum, teaching through active learning and providing greater emphasis on creative and practical thinking than is common in high school STEM classes. The curriculum does this through the avenues of three major elements of a Makerspace: 3-D printing, app development, and Arduino circuitry.

I have disseminated my curriculum for use by high school Makerspace teachers via the website ”Teachers Pay Teachers” [10], as well as the website ComPADRE.org [2] with the help of Professor Deconinck. The curriculum is posted under the Creative Commons Attribution-NonCommerical-ShareAlike 4.0 International license [3], allowing others to take and edit my curriculum for their own purposes as they see fit, so long as they do not attempt to profit off of the creations and so long as they put their creations under the same licensing as mine own.

4.1.2 Future Plans

Through the testing and feedback processes, I received some suggestions which I was uncertain about implementing or did not have time to implement. I hope to address these in the future to improve the quality of the curriculum for those who make use of it. One suggestion that I received for both the app development module and the Arduino module were to address the possibility of students who might struggle with creative and practical thinking by providing example projects to pursue for these students. While, it may be necessary to have some ideas ready if students cannot come up with any of their own, doing so also takes away from the curriculum's objectives of getting students to develop their own practical and creative thinking. For this reason, I am uncertain about implementing this suggestion. Another suggestion was to add another intermediate project to the app development curriculum, between the BirthdayToWeekday app and the Team Projects. While I did not have time to implement this in my time this year, I think it would be helpful to have an additional stepping stone and so hope put together an additional project in the future.

Bibliography

[1] "Arduino," Linda Barton. Private communication.

[2] ComPADRE Digital Library. Compadre.org Homepage. ComPADRE.org - Resources for Physics and Astronomy Education, ComPADRE Digital Library, www.compadre.org/.

[3] Creative Commons License Deed. Creative Commons - Attribution-NonCommercial-ShareAlike 4.0 International - CC BY-NC-SA 4.0, creativecommons.org/licenses/by-nc-sa/4.0/.

[4] Freeman, S., et al. Active Learning Increases Student Performance in Science, Engineering, and Mathematics. Proceedings of the National Academy of Sciences, vol. 111, no. 23, 2014, pp. 84108415., doi:10.1073/pnas.1319030111.

[5] Inra, Jennifer A, et al. An Active Learning Curriculum Improves Fellows Knowledge and Faculty Teaching Skills. Advances in Medical Education and Practice, Volume 8, 2017, pp. 359364., doi:10.2147/amep.s135538.

[6] Mueller, Ashley L., et al. Exploring the Effects of Active Learning on High School Students Outcomes and Teachers Perceptions of Biotechnology and Genetics Instruction. Journal of Agricultural Education, vol. 56, no. 2, 2015, pp. 138152., doi:10.5032/jae.2015.02138.

[7] OutSystems LLC. OutSystems. The #1 Low-Code Platform for Digital Transformation, 2000, www.outsystems.com/.

[8] Physics Standards of Learning For Virginia Public Schools. Virginia Department of Education, Virginia Department of Education, Jan. 2010.

[9] Sternberg, Robert J., et al. Styles of Learning and Thinking Matter in Instruction and Assessment. Perspectives on Psychological Science, vol. 3, no. 6, 2008, pp. 486506., doi:10.1111/j.1745-6924.2008.00095.

[10] Teaching Resources & Lesson Plans. Teachers Pay Teachers, www.teacherspayteachers.com/.

[11] Ultimaker 2+: Engineered to Perform — Ultimaker. Ultimaker.com, ultimaker.com/en/products/ultimaker-2-plus.

[12] Ultimaker Cura: Advanced 3D Printing Software, Made Accessible — Ultimaker. Ultimaker.com, ultimaker.com/en/products/ultimaker-cura-software.

[13] SketchUp Make. SketchUp, www.sketchup.com/products/SketchUp-make.

Appendix A

Makerspace Curriculum

The following pages show the Makerspace curriculum in PDF pages.

3-D Printing Module – By Seamus Herriman

Assumes Ultimaker 2, or other printer that can take file types produced by Cura.

Objectives:

1. Learn about functionality and operation of 3-D printer.
2. Learn to use and develop familiarity with 3-D design software.
3. Work as a team to design and create 3-D printed solution to problem.

Lesson Plan

Stage 1: Introduction to 3-D Printing, Thingiverse, and Cura

Objectives:

1. Become familiar with 3-D printer and its operations.
2. Learn to download and edit STL files for printing with Cura.
3. Learn to upload edited STL files to 3-D printer.

Prerequisites:

1. All students must have computers with software Cura downloaded. Software SketchUp will need to be downloaded by Stage 2.
2. Have 3D printer set up at front of classroom with room for students to circle around and observe you walk them through its parts.

Stage Length: 1.5 – 2 hours

PowerPoint Link:
https://drive.google.com/file/d/0B9MQUUNEzlyTbzJtSHBMb25NM1E/view?usp=sharing

Teacher's Instructions:

3-D Printer Walkthrough – 15-20 minutes

Students should come into class and sit in groups with computers. When ready to begin, welcome the class, and have students come forward to examine the 3D printer as you walk them through its parts.

Overview the components of the 3-D printer with students. Below is a suggested breakdown of the overview.

3D printer uses a heated extruder along with three axial drivers to melt filament and arrange it on an adjustable bed according to a precise, given design.

What is the 3D printer made of?

- Motors, Glide Rails, and Drive Belts for X and Y axes

BY-NC-SA

- Screw motor and Rail for Z Axis
  a. Ask students: Why is there no drive belt?
  b. Answer: Only moves in one direction, so no need for belt. Screw is slower and more stable, while drive belt allows for more versatile motion in X and Y.
- Extruder – melts and deposits filament
  - Adjustable bed – provides foundation on which to build 3D print
  - Electronics – control all motors and the extruder to construct given 3D models
  - Frame – holds components in place

What does the 3D printer print with?

- ABS – more durable, does not break down
- PLA – biodegradable, lower heat threshold (easier to fix), more brittle, so it will break down eventually
- Ask students: What would be advantages and disadvantages to using each?

How do we tell the 3D printer what do print?

- Upload digital 3D models in the form of gcode files, which the printer electronics can reach and interpret as instructions for its three motors and the extruder.

Using Thingiverse and Cura – Thingiverse (10 minutes), Cura (40 minutes)

Have students return to their seats. They will be working at their computers for the rest of the class period. Each student should be at their own computer.

How do we get or make digital models?

- Two ways
  - Download pre-made models from Thingiverse, a website that has 3D models of many objects
  - Google SketchUp, a free application that allows us to create designs without extreme modelling and programming knowledge

Explain to students that they will be downloading an item from Thingiverse and using the software Cura to prepare it for printing. Give students about 10 minutes to look at different models on Thingiverse and find something they want to print. The download will be an STL file to be manipulated in Cura.

a. Ask students to examine different models of prints, as shown in their descriptions.
   i. Why are STL models set on the ground in particular positions?
   ii. How could improper positioning when creating model cause issues in printing?
   iii. What are ways to avoid printing issues for complex prints?

Once students are ready, have them open Cura and <u>select the printer your class is using</u>. When this is done, they will see a model of the printer in Cura. Have students 'Open Files' and select the STL file they downloaded from Thingiverse. When this is done, they will see it appear in Cura with the printer.

BY-NC-SA

Steps in Cura

1. Make sure the entire print fits inside the printer and takes an appropriate time to print. Cura automatically updates the print time.
2. Use the movement tool, scaling tool, and rotate tool to find the best way to place their prints.
3. Be aware of the necessity to create structural supports if elements of the print are suspended in midair. This only requires checking off a box, but will increase print time, since the supports need to be printed as well. Supports can be viewed by changing the view mode.

Stage 2: Introduction to SketchUp

Objectives:

1. Learn to build objects using SketchUp.

a. Building 3-D shapes through pushing and pulling.

b. Building curves shapes using arcs.

c. Learn grouping vs. components to create complex objects.

Prerequisites:

1. All students must have computers with software SketchUp downloaded.
2. https://extensions.sketchup.com/en/content/sketchup-stl
3. https://help.sketchup.com/en/article/3000263

Stage Length: About 4 hours, assuming time to get class periods started and end class periods

In this stage, students will build a given house, a series of given curved shapes, and chair with a back and legs to learn about different available functionalities of SketchUp.

Teachers should have Cura opened on their computer and projected in the classroom so that they can walk students through different 3D design options.

Start of Lesson:

Students should come into class and sit in groups with computers. Have them open SketchUp, as they will be spending the entire class section here. Students will work at computers individually. Teachers should carry out the instructions in SketchUp over a projector to help students follow.

Explain that students will learn to make some basic shapes in SketchUp. To start, show students tools to help them navigate the environment.

1. The hand allows you to move the position of what you are viewing.
2. The axis allows you to move the position from which you view the scene.

(cc) BY-NC-SA

Push, Pull and Line Moving – 30 minutes

First, students will make 3-D objects using SketchUp's Push and Pull features by making a house with a pitched roof, a door, steps, and windows.

1. Go over to the menu on the top and select the 2-D rectangle drawer. Use to draw a rectangle on the ground. Draw by clicking once to drop one corner, pulling out the rectangle, and clicking again to drop the opposite corner.
2. Now, to go from 2-D to 3-D, use the Push-Pull tool to pull the rectangle up into a box. This tool is found on the toolbar as an arrow coming out of a rectangle perpendicular to the surface. Using the tool, click the surface you want to move, and move the cursor to push and pull the surface. When you have it where you want it, click to end the move.
3. Next, use the line tool to make the roof sloped. Open the line tool and click on the midpoint of the edge of the roof. The pointer will change color when it is on the midpoint. Go to an adjacent side and click on the edge of the wall to make the connecting line. Start another line from the midpoint of the roof's edge, hover the cursor over the end of the first line on the side of the wall to make an inference, and finish the second line so that its end is at the same height as the first line's end. The inference will let you know when you are parallel to the first line's end. Now, the Push-Pull tool will move any surface that is defined by lines. With this in mind, use the Push-Pull tool to remove push the two triangles back. When they make a 2-D surface at the other end of the roof, clicking will make them disappear.
4. Now, use the rectangle tool to draw steps and a door on the front, and use the Push-Pull tool to make the steps stand out and the door become recessed. To remove extra lines on the steps, use the eraser tool.
5. Now, in order to make the roof pitched in all directions, use the line tool to make triangles on the roof between a point on the top edge of the roof and the roof corners. These are the lines where the roof will fold. Now, choose the Move tool. Make an inference at the corner of the roof and pull the tip of the top edge down to that level.
6. Finally, add windows using rectangles, pushes, and pulls.

Curves and Curved Surfaces – 60 minutes

Now, students will make a few curved and spherical objects.

First, draw a line curve. Give students about 5 minutes and experiment with the different types of curves that can be drawn and think about how they might be useful in different situations. Point out that when drawing curves off of curves, the software will alert you when your new curve follows the old curve continuously.

Next, have students use circles to make straight cylinders and pipes.

1. Go to the rectangle tool and open its options to switch it to a circle tool. Using the circle tool, click once, drag out, and click again to draw a circle (the farther you expand the circle, the more it will show to be a series of line segments that act as one object). Use the push and pull tool to make a cylinder.

BY-NC-SA

2. Now, make a pipe. Draw another circle, and then choose the inset tool, which has two arches connected by an arrow. Clicking your circle, move your cursor to make a second circle within the first, and click again to set it. Switching to the cursor, click on the inside of the circle and hit delete on your keyboard. Now you have a rim surface. Use the push and pull tool to make this a pipe.
    a. You can also use the scale tool, which shows a square with a new perimeter, to change the scales of different surfaces, which can drastically change the look of your object.

Give the students time to practice making cylinders and pipes and scaling their dimensions.

Next, show students how to use the follow me tool to make bowls and curving pipes.

First, the bowl.

1. Make a circle at the origin.
2. Use the arrow keys on your keyboard to change the orientation of the circle maker. Then, use the shift key to hold movement in one direction, and move the cursor up. So long as you see dotted line glowing the color of the cursor, you are going in a single direction. If the circle cursor glows blue, you are moving in the z-direction. If it glows red or green, you are moving in the x- or y-directions, respectively. Toggle direction of focus with the arrow keys.
3. Once you have a circle above your first, perpendicular to your first, make an inner circle using the inset tool. Next draw a line through the diameter horizontally (or close to), and use the eraser tool, clicking, holding, and dragging it across the top half, to delete it. Click and delete the inside of the bottom half. You now have a half rim.
4. Now, with the arrow cursor, click on the perimeter of the bottom circle. Go to the 'Tools' menu and choose 'Follow Me', a click on the face of the half rim. You should now have a bowl. If you do not, you may have clicked things in the wrong order. Call for help here if you do not have a bowl.
5. Order is important here. If you got your bowl the first time, go to 'Edit' → 'Undo Extrusion'. Now drag the arrow cursor across the half rim to select it, and after activating 'Follow Me', click the edge of the circle. Here, you will see how order affects the results of an extrusion.

Now, students will make a curved pipe.

1. Draw two curves that link into a long, winding curve. Use the changing color of the arch to make sure that the two curves are tangent at their meeting point.
2. At the end of the long curve, draw a circle such that the curve seems to draw out of its center, perpendicular to it (this circle should be perpendicular to the plane of the long curve. Make an inner circle with the inset tool, and delete the inner circle.
3. Select the curves by clicking one with the arrow cursor, holding shift, and clicking the other. Go to the 'Follow Me' tool, and click the outer rim surface. You now have a curving pipe.

Give students time to experiment with the available tools to see how they affect and build objects.

(cc) BY-NC-SA

Components and Groups – 60 minutes

Explain to students that they will now look into components and grouping by making a chair. These are useful tools for creating more complex models which are more easily made as a succession of smaller, combined pieces.

1. First, have students make a tall, thin square to act as a leg. Once you have this made, use the arrow cursor to highlight the cylinder and use control, shift, G, to make it a component. Name this component 'Leg'. Now it can be copied and pasted. Copy and Paste 3 more legs, and arrange them in a square, using the arrow keys to lock the movement the legs along precise axes, and using inferences to help form the legs into a square.
2. Now, make a wide, low square for the seat of the chair by making a rectangle at the outside corner of one leg and finishing it at the outside corner of the opposite leg. Use the Push/Pull Tool to make this 3D seat. Draw two more squares on one side to make the chair back, make a rectangle at the top to connect them.
3. Once you have all the parts of the chair together, use the arrow cursor to highlight the whole group, and use control, G, to make it a group. Now you can move it as one piece.
4. Once the chair has been grouped, it cannot be edited. Right-click and explode the chair in order to edit it, if need be.

Help struggling students to build their chairs. If students finish, have them experiment with combining all of the tools they have acquired.

Stage 3: Team Projects

Objectives:

1. Work as team to solve engineering need.
    a. Identify need.
    b. Develop solution.
    c. Test solution.
    d. Make corrections to design based on testing.

Stage Length: 12 hours

Break steps up as necessary over class periods.

Have students sit in groups of four. They will be working together as a group to complete the project.

For this project, groups will brainstorm a problem to fix with a 3-D print, identify necessary criteria and constraints for the print, and develop a print that meets these parameters. Groups will then test their prints against the problem, determine any changes that could be made to improve the print, and redesign their 3D model based on these findings. In the end, each group will present their completed model, describe how it was originally created, and what was changed after testing.

BY-NC-SA

Throughout process, emphasize importance of thinking both creatively about what could be done with 3-D printing and practically about what tools would be most effective for realizing that vision.

Steps:

1) Brainstorm/research ideas.
Give groups 1-½ hours to brainstorm problems that they would like to attempt to address. Have groups make a list of these problems. By the end of time period, each group must decide which problem to base their project off of.

2) Identify criteria and constraints.
Give groups 40 minutes to identify the constraints that their model must adhere to in order to work practically. When groups are finished, have them discuss with the teacher to insure their ideas are reasonable.

3) Develop and test prototype.
Give groups 4-5 hours in which to develop a prototype model in SketchUp, print it, and test its practicality. Students should take notes on how they develop the prototype, including what tools they used in SketchUp and why. They should also take notes on the strengths and weaknesses of their prototype when tested. These details will be necessary for their final presentation. For more complex models, suggest to groups to break their model into multiple pieces to be combined after printing. Also, advise students that are not currently working at a computer to collaborate with their teammates working at computers to help find the best way to use the tools available within the SketchUp software.

4) Re-assess prototype and design/print new model.
Taking their original design and their notes from testing, groups should redesign models to improve practical functioning. Give students approximately 3 hours in which to do this. (Note for teachers: Final prints should not be expected to work perfectly in the end, but should show an improvement from the original model.)

5) Present.
Give groups about 40 minutes to put together notes on their work to create presentation. Groups should present their problem, their model, how they designed their model, and what improvements they made to their model after testing. This should include how they used what they learned throughout stages 1 and 2. This time does not include time to present the projects, as that will depend on the number of groups in a class.

(cc) BY-NC-SA

App Development Module – By Seamus Herriman

Objectives:

1) Introduce students to OutSystems
2) Gain familiarity with OutSystems mechanisms and learn process of app development
3) Have students create own apps in pairs

Module Notes – Module developed on Mac using "Limited Web Beta" version. Module's compatibility with OutSystems Windows application is untested.

Lesson Plan

Stage 1: Introduction to OutSystems

Have students create accounts on OutSystems and familiarize them with OutSystems' functionality. Walk students through building a basic app (Tutorial).

Objectives:

4. Create accounts in OutSystems.
5. Walk through OutSystems tutorial.
    a. Familiarize selves with OutSystems' basic operations.

Prerequisites:

3. Each student must have access to a computer with Google Chrome.

Stage Length: 40-60 minutes

Teacher's Instructions:

For this section, the class will be doing an introduction to app development. This will be done using the platform OutSystems.

Have students open an account on OutSystems. When students have successfully opened an account, have them enter the "Limited Web Beta" version. They will encounter the option to enter a tutorial. When everyone is successfully logged in, start the tutorial together. This will give students an introduction to the basic framework of how OutSystems works. As students go through the tutorial, the teacher will emphasize different characteristics of OutSystems and alternative options for tasks given in the tutorial. These are given below.

Tutorial Notes:

i. Make sure to read the orange instruction guides. These give useful information for students (in some cases to be elaborated on) and should not be brushed over.

ii. Pause after "Create App". You are not at a screen that shows the app as the overall project, with modules underneath as elements of this project. Point out to students how this can offer the means to create apps made up of multiple modules, and thus increased complexity, in the future. Continue the tutorial.

(cc) BY-NC-SA

iii. Once the development platform is open, walk the students through its parts. There is the main screen, which shows the user interface, widgets that can be added to the interface on the left, and information on processes, interface elements, logics, and data on the right. At the bottom is the Debugger, where errors and warnings will show when the software of an app is not functional.

iv. After students import the Excel data, have them create two new Entities, one in Database, and one in Local Storage (do so by right clicking on Database or Local Storage, as the tutorial instructions describe). Have students open the Entities to see what Actions are automatically created in association with the Entity. Ask students what they think the difference is? Explanation: Database uses Cloud, so it needs Wifi, while Local Storage requires memory space on device but no Wifi. Have students supply names and features for the Entities.

v. When dragging "Todo" into the screen to make a list, emphasize that this creates an aggregate that allows an app developer to pull information from "Todo". Students may wish to create aggregates to access information in their future apps.

vi. When linking to the new screen, take a moment to look at other options for what can be done and what can be linked to. Ask students how they might use different options.

vii. When double clicking on the button, students are taken to the logic creation window. Here, a graphic interface is offered to help students develop actions without intense coding. Options are offered on the left to add to the frameworks, such as "If" statements, "For Each" loops, and "Assign" statements.

viii. To create more diverse actions, OutSystems offers a JavaScript options that allows developers to expand their programming with JavaScript code. This will be explored in Stage 2.

ix. If a pop-up window comes up when attempting to deploy the app, click on the window to access the app.

If the class finishes early, give students time to experiment with the using the app and with adding more actions to the app.

Stage 2: The Process of App Development

Walk students through breaking idea into individual pieces in order to build it. Use BirthdateToWeekday app as idea to break down.

Objectives:

1. Learn to break app concept down into buildable pieces.
2. Learn about troubleshooting problems.

(cc) BY-NC-SA

3. Create functional app that returns weekday of birthdates that students can download.

Prerequisites:

4. Each student must have access to a computer with Google Chrome.

Stage Length: 3 hours

<u>Teacher's Instructions:</u>

For this section, the class will be going through the process of breaking down an idea for an app into pieces, and how to find solve problems that they encounter in the code.

Students will build an app that takes a birthdate as input and returns a weekday at output.

This stage is structured as a series of questions and suggestions given to the students by the teacher, where the teacher guides the students through the process of creating the app and builds the app over projector, while students follow along at their own computers.

Create new mobile app for phone called "BirthdayToWeekday".

When building the app, ask students questions to drive them to the next step. Questions are provided below. Field responses from student and either lead them to the correct answer, for concept questions, or try their ideas out, for development questions. Work together through any errors that arise. If the class cannot figure out what an error is asking them to change, show students ways to understand errors, such as looking them up in OutSystems Support or on Google. These skills will come in handy when students build their own apps. If the class gets too stuck, a safe option is offered with the given step. Safe options are made assuming the other safe steps were used, so they may not work as given with alternative designs.

Steps to Build:

1) Concept: What are the two basic parts of an application? – Input and Output. Here, date and weekday.
2) Development: How should the date be inputted? – Dropdown for Month, Input for Day and Year. Must create Static Entity for Months, so that users can choose a Month. Within the Month Dropdown, make sure Values is Month.Order, not Month.Id. Create local variables to store responses to each input.
3) Development: How can we tell users what each input it for? – Text boxes.
4) Development: How can we turn the input into an output? – Button and Action.
5) (FOR THIS STEP, WALK STUDENTS THROUGH THE GIVEN PROCESS. CODING ACTIONS CAN BE TEMPERMENTAL) Development: How do we turn a birth date into its corresponding weekday? In the Action window, add a JavaScript block to the pathway and name it "GetDay". At the bottom of GetDay, right-click parameters to add input parameters for Year, Month, and Day, and an output parameter for DayOfWeek. For the code block, use the below:

(cc) BY-NC-SA

```
var birthDate=new Date($parameters.Year + "-" + $parameters.Month + "-" +
$parameters.Day);
var birthDay=birthDate.getDay();
var
weekDays=["Sunday","Monday","Tuesday","Wednesday","Thursday","Friday","Sat
urday"];
$parameters.DayOfWeek=weekDays[birthDay];
```

Have this code available for students to copy and paste it into their work, so they can focus on going over how the code operates.

After the JavaScript block is complete, insert an "Assign" block after it. Create a Local Variable to assign the DayOfBirth, and within the Assign block, set DayOfBirth to be GetDay.DayOfWeek. Now the app has a local variable for the output day of the week.

6) Development: How do we show the output? – On the Homescreen, insert an Expression, and set its value equal to DayOfBirth. Set an Example day if you wish.
7) Congrats, launch the app!

Stage 3: Team Projects

Have students build apps of their own in pairs, offering guidance as necessary.

Objectives:

2. Work as team to solve need.
    a. Identify need.
    b. Develop solution.
    c. Test solution.
    d. Make corrections to design based on testing.

Stage Length: 9-11 hours

Break steps up as necessary over class periods.

<u>Teacher's Instructions</u>

Have students sit in pairs. They will be working together as a group to complete the project.

For this project, groups will brainstorm a problem to fix with a mobile app, identify necessary criteria and constraints for the app, and develop an app that meets these parameters. Groups will then test their apps, determine any changes that could be made to improve the app, and redesign their app based on these findings. In the end, each

BY-NC-SA

group will present their completed app, describe how it was originally created, and what was changed after testing.

Throughout process, emphasize importance of thinking both creatively about what could be done with apps and practically about what tools would be most effective for realizing that vision.

Steps:

Brainstorm/research ideas.
Give groups 1-½ hours to brainstorm problems that they would like to attempt to address. Have groups make a list of these problems. By the end of this day, each group must decide which problem to base their project off of.

Identify criteria and constraints.
Give groups 40 minutes to identify the constraints that their model must adhere to in order to work practically. Key information for groups to identify are what input the app will take, what output the app will offer, and what features in OutSystems might be used to produce this. When groups are finished, have them discuss with the teacher to insure their ideas are reasonable.

Develop and test app.
Give groups 5-6 hours in which to develop a prototype app in OutSystems and test its practicality. Students should take notes on how they develop the app, including what elements they used in OutSystems and why. They should also take notes on the strengths and weaknesses of their app when tested. These details will be necessary for their final presentation. Also, advise students that are not currently able to type, because their partner is at the computer, to collaborate with their partner to help find the best way to use the tools available within the SketchUp software.

Re-assess app and develop new model.
Taking their original design and their notes from testing, groups should redesign apps to improve practical functioning. Give groups 2-3 hours to do this.
(Note for teachers: Final apps should not be expected to solve all possible issues in the end, but should show an improvement from the original model.)

Present.
Give groups 40 minutes to put together notes on their work to create presentation. Groups should present their problem, their app, how they designed their app, and what improvements they made to their app after testing. This time does not include time to present the projects, as that will depend on the number of groups in a class.

CC BY-NC-SA

BirthdayToWeekday Code

var birthDate=new Date($parameters.Year + “-“ + $parameters.Month + “-“ + $parameters.Day); \\ creates Date object in JavaScript, using information given by user
var birthDay=birthDate.getDay(); \\ uses Date object’s inherent actions to get the day of the week that the date occurred on (day is number 0 to 6 corresponding to weekday
var weekDays=
[“Sunday”,”Monday”,”Tuesday”,”Wednesday”,”Thursday”,”Friday”,”Saturday”]; \\ makes array of weekday names
$parameters.DayOfWeek=weekDays[birthDay]; \\ searches array for correct day and assigns text day name to an output parameter, to be used to show the user the day name

(cc) BY-NC-SA

Arduino Module – By Seamus Herriman[1]

Objectives:

1) Introduce students to Arduino Board, Arduino Integrated Development Environment (IDE), and various Arduino components
2) Teach students rules and limits to safely work with Arduino electronics and begin practicing basic circuits.
3) Have students develop their own Arduino circuits with personalized coding.

Module Notes –

1) Arduino IDE installation required.
2) School will need to purchase various Arduino components for students to experiment with in school.

Stage 1: Teach students about Arduino Board, Arduino IDE, and component data sheets.

Objectives:

1) Learn basic functionality of Arduino Uno Rev 3.
2) Learn set up of Arduino IDE.
3) Teach students to read component data sheets for relevant information.

Prerequisites: All students must have computer, breadboard, and Arduino Uno Rev 3 board.

Stage Length: 1 hour

Recommended:

1) Have overhead to show students parts of Arduino board projected on board.
2) If students are using school computers, make sure the Arduino IDE is downloaded and installed. It can be found at https://www.arduino.cc/en/Main/Software. Otherwise, have students download and install the software prior to class if possible, or between examining the board and examining the IDE itself, if not.

Teacher Instructions:

In this section, students will be introduced to the basics of working with Arduino.

Explain that students will be going over the Arduino Uno Rev 3 board, the Arduino IDE (Integrated Development Environment), and how to learn about different components.

First, pass out the Arduino boards to students, one to each. Keep one to walk students through its structure. Using an overhead, take students through the important parts of the board to know to use it properly. These parts include:

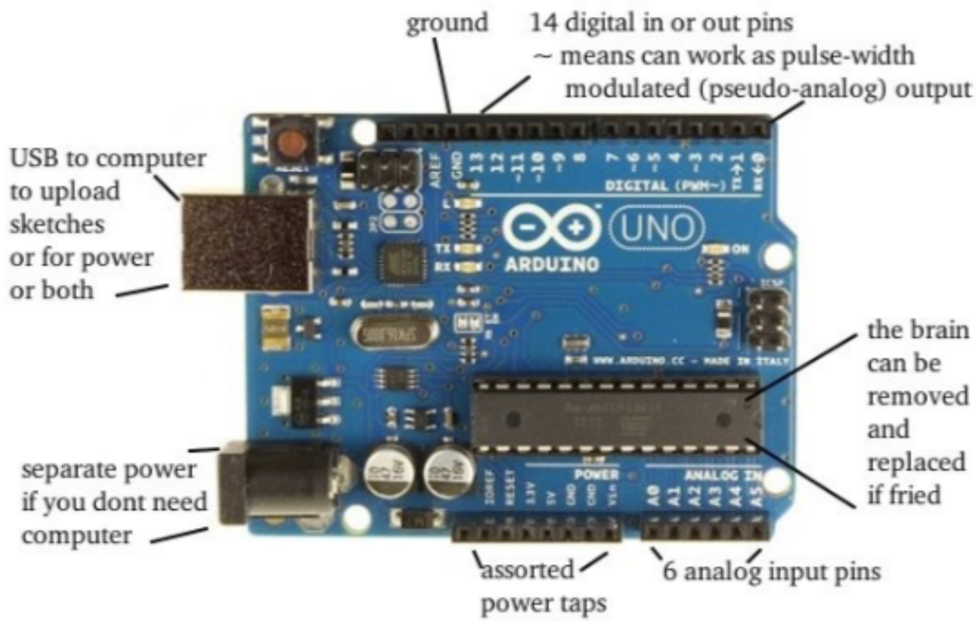
---

[1] The contents of Stages 1 and 2 of this module are an adaptation of the work of Linda Barton, received via private communication.

(cc) BY-NC-SA

1) USB for connecting to computer to upload sketches, acquire power, or both
2) Simple power connector when computer is not necessary
3) Ground
4) 14 digital in or out pins ~ allows for pulse-width modulated (pseudo-analog) output
5) Assorted power taps
6) 6 analog input pins
7) Central processor – brain that can be removed and replaced if fried

Picture for reference:



Notes:

1) Even if you use a computer to program the board, once the sketch has been uploaded, the computer is no longer necessary.
2) Digital pins have power limits of ~30 mA at 5 V and any more current will damage the board. WATCH THIS CLOSELY.
3) Power tap pins also have very sensitive current limits
   a. < 150 mA for 3.3 V tap
   b. < 400 mA for 5 V tap

Next, turn students to components. All components have data sheets with relevant safety information that it is important for students to be familiar with. Explain to students that they need to be familiar with the parameters of a given component before they use it, so

(cc) BY-NC-SA

that they do not damage it or the board accidentally. Use small signal NPN transistor and Sparkfun Relay as examples.

For the transistor, the first important parameters are the ‘Absolute Maximum Ratings’, in particular the collector emitter maximum voltage and the collector current. These specify the conditions to stay within to avoid damaging or ruining the transistor. Under ‘Electrical Characteristics’, students should pay attention to the DC current gain, which will tell them how the transistor effects the output of the transistor to the rest of the circuit.

For the relay, important information include the features, such as the power consumption, and maximum and minimum limits for current, voltage, power, and switching.

Emphasize to students the importance of taking such information into account when building Arduino circuits.

Next, show the students the breadboard. Explain its structure (metal strips along rows of holes that allow users to connect components together into circuits).

Now turn the class to the Arduino IDE. The IDE, short for Integrated Development Environment, is where sketches are made. “Sketches” are pieces of software that developers write to control Arduino circuits.

Once the application is opened, point out to the students the ‘Compile’ button, represented by a check, and the ‘Upload’ button, represented by an arrow. For more advanced future use, students can use the magnifying-glass button to use serial communication to pass variables or data within your sketch. An image reference is attached below.

(cc) BY-NC-SA

Have students write up the ‘blinky1’ sketch (code is given below if blurry in picture) and walk them through its main structural elements. These include:

1) Preliminaries – set up constants
   a. Note: this is used to define various pins

```
const int LED = 13; // connect LED to pin 13
```

2) setup function – prepares pin to operate

```
void setup ()
{
   pinMode(LED, OUTPUT); // sets the digital pin as an output

}
```

3) loop function – executes pattern in pins until turned off

```
{
  digitalWrite(LED, HIGH);
  delay(1000);
      digitalWrite(LED, LOW);
    delay(1000);
}
```

(cc) BY-NC-SA

Code in full:

```
const int LED = 13; // connect LED to pin 13

void setup ()
{
   pinMode(LED, OUTPUT); // sets the digital pin as an output

}

void loop ()
{
  digitalWrite(LED, HIGH);
  delay(1000);
  digitalWrite(LED, LOW);
  delay(1000);
}
```

End of Code

Students will use the sketch in Stage 2.

Stage 2: Components and Their Uses

Walk student through construction of various Arduino electronic circuits. **E**mphasize safe building with attention to rules and limits of electronics use.

Objectives:

1) Learn to safely put together circuits.
2) Build knowledge of different Arduino components, their uses, and Arduino IDE commands to make use of them.
3) Have students make connections between changes to code and changes to circuit or circuit function, and vice versa.

Prerequisites: All students must have:

1) Computer with Arduino IDE
2) Arduino Uno Rev 3 board
3) At least 2 LEDs
4) Switch
5) Piezo buzzer
6) 2N3904 Transistor
7) CdS Photoresistor
8) PN2222 Transistor
9) Relay

BY-NC-SA

10) 12 V Fan

Stage Length: 10 hours

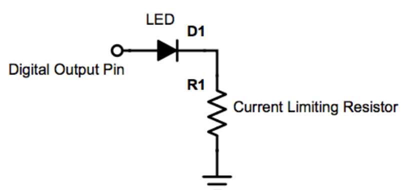
<u>Teacher's Instructions</u>:

In this section, students will build a series of circuits with accompanying sketches.

<u>LEDs (Light-Emitting Diodes)</u>

- produce light
- LEDs are polarized, which has two effects
  - essentially zero resistance when connected in direction of current
  - essentially open circuit (no current flow) when connected opposite direction of current, unless current is strong enough to fry the LED

Start with the sketch given at the end of Stage 1. Have students build a circuit that matches the circuit drawn below, where an LED connects to digital output pin 13, then to a current limiting resistor, then to ground. When plugging in the LED, the short leg (cathode), should be on the LED's negative side (closer to the ground). Some LEDs mark this with a flat side. Conversely, the LED's long leg (anode), should be on its positive side, farther from ground. Ask students why a current limiting resistor might be necessary (The LED offers no resistance itself, so to keep within safe current limits, a resistor is needed). Have students use the V=IR equation to calculate the necessary resistance for the current of the circuit to be 20 mA.



Once the circuit is ready, plug it into the computer via the USB cable, and upload the compiled sketch.

Once this circuit is working properly, move only the next step.

For the next step, have students make the circuit slightly more complicated by adding more digital output pins for more LEDs. Make sure to take precautions to control current and update the sketch as necessary. Give students free reign to re-design the light patterns.

Ask students: What are applications of this kind of circuit?

(cc) BY-NC-SA

Switches

- allows manual opening and closing of circuit

For the next circuit, have students add an input line as well as an output line. This input line will be a switch that controls whether the LEDs are on or off. The LEDs will be on when the line is HIGH, and off when the line is LOW. The code for this is given below.

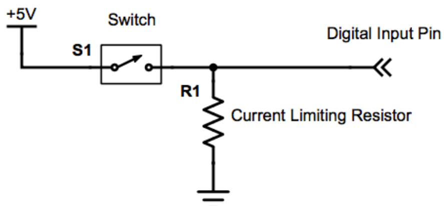
```
// two blinky LEDs with a switch to control them

  const int LED =13 ; // connect LED to pin 13
  const int LED2=5;   // connect second LED to pin 5
  const int SWITCH=2;
  int val=0;

void setup () {

   pinMode(LED, OUTPUT);  //sets this digital pin as an output
   pinMode(LED2,OUTPUT); //sets the digital pin as an output
   pinMode(SWITCH,INPUT); //sets pin 2 as a switch digital input

}

void loop()

  {
    val=digitalRead(SWITCH); //decide if switch pin 2 is hi or lo
    if (val==HIGH) {       //   if its hi, blinky pattern
   ... insert your code for whatever blink pattern you like ...

}

   else {                //if its lo, shut the LEDs off
   digitalWrite(LED,LOW);
   digitalWrite(LED2,LOW);}

}
```

Note that the code for the LEDs actions when the switch is HIGH must be put within the curly brackets.

To test the circuit, have students use a wire to connect the digital input pin to either ground or the 5 V source. Take notice what the LED does when connected to ground, 5 V, or neither.

Once the circuit is working, add a push button switch, either SPDT (single pole, double through) or SPST (single pole, single throw). Use a DVM ohmmeter to figure out how the push button switch works (only makes contact when pushed). A diagram for this circuit is below

(cc) BY-NC-SA



Think about how the resistor effects the circuit (keeps current low when switch is on) and pin is HIGH. Discuss with students to insure they understand the circuitry. (Because the LED assumes the HIGH setting unless connected to ground, it must be physically connected to ground in order to go off. With switch connects it to the 5 V source, and the resistor controls the current of the circuit at these times.)

Ask students: What are applications of this kind of circuit?

Dimming LED with Pulse Width Modulation (PWM) and analogWrite()

For this sketch, students will to use the Arduino Board to simulate analog output. This is done by using train of pulses where the pulse widths are modulated (lending to the name). In our board, the analog output has an 8-bit conversion, which means that the maximum achievable resolution is $2^8 = 256$, which from 5 V is 20 mV steps, roughly. Digital output pins that have this capacity have an ~ before the number printed on the board.

To make use of PWM analog output, students will use the command analogWrite(pin, N), where 'pin' is the pin number, and N is an integer between 0 and 255 (this gives the width modulation).

For the necessary circuit, have students put their LED into one of the PWM pins and (of course) add a current limiting resistor (200-300 Ohms should be fine).

To make the LEDs alternatively grow brighter and dimmer, for loops are necessary to change the value of N. Code for this is given below.

```
const int LED =3 ; // connect LED to pin 3 int i=0;

void setup () {

    pinMode(LED, OUTPUT);  //sets the digital pin as an output
   }
void loop() {

    for (i=20; i<255; i++)
```

(cc) BY-NC-SA

```
    {  analogWrite(LED, i);
delay(15); }

    for (i=255; i>20; i--)
    {  analogWrite(LED,i);
      delay (15);}
  }
```

Compile, upload, and see how it works. Discuss with students to insure they understand the circuitry. (The LED is turned on and off rapidly for variable intervals. Human eyes only see the average, resulting in a variable brightness for the non-analog LED light.)

Oscilloscope – rigol $300 to see what is happening with the pulse
Analog is made.

Ask students: What are applications of this kind of circuit?

<u>Piezo Buzzer</u>

For the next circuit, students will use a piezo buzzer. Piezo buzzers make noises, and Arduino can control the pitch, volume, and length of the sound. The piezo is not polarized, unlike the LED, which means it can be attached directly from a digital pin to ground. The buzzers are not very loud and are softer at lower frequencies. They operate best for sound frequencies above 2 kHz.

In the sketch, set the PZ as a constant, and as an output. For the main loop, use something along the lines of the following. Feel free to adjust numbers to see their effects.

```
 {
    tone(PZ,1760); // replace PZ with the number of the pin it is in.  The other
number is
    delay(1000);    // the frequency
    tone(PZ,880);
    delay(1000);
    noTone(8);    // the PZ remains on until it is turned off.
    delay(1000);
}
```

Try to make the PZ play a tune. Note pitches are found here https://www.arduino.cc/en/Tutorial/toneMelody.

To use the PZ without a specific frequency, use the following.

(cc) BY-NC-SA

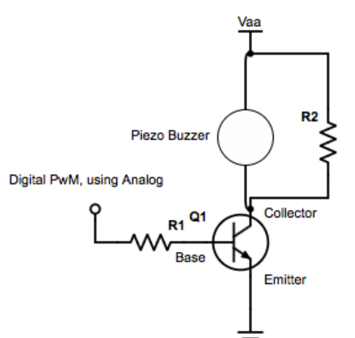
```
void loop() {

    analogWrite(PZ,128);
    delay(time);
    digitalWrite(PZ,LOW);
    delay(time);
}
```

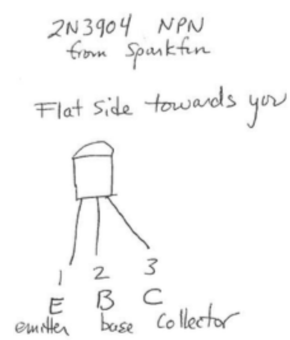
Time is a set constant.

Ask students: What are applications of this kind of circuit?

Piezo Buzzer with Transistor – Louder Piezo

Next, students will use a transistor to make the piezo louder. This works because the piezo's volume is currently limited by the power supply it is attached to, 5 V and 20 mA. Transistors function as current amplifiers, so that while the base current is small, the current from the collector to the emitter is significant. This greatly amplifies the piezo's volume. The transistor also function as a switch to the extent that when no current flows through the base of the transistor, no current in able to flow from the collector to the base. A diagram of the necessary circuit is below. Use a 2N3904 transistor, or any transistor that is rated for 200 mA collector current.





Have students use the same sketch as the last circuit, gradually increasing the power supply and seeing if they can hear the difference. Take time to analyze the circuit to see what is going on. When doing so, consider the C-E path as a short when the transistor is conducting. Also, try to determine the use of the resistors.

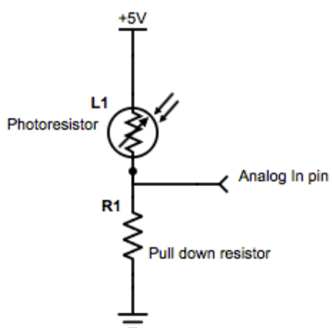
Ask students: What are applications of this kind of circuit?

(cc) BY-NC-SA

Light Sensor

For the next circuit, students will be using a light sensor, specifically, a CdS photoresistor. This sensor operates by providing resistance based on the intensity of incident light received, providing several hundred kOhms of resistance when in a dark environment and only a couple kOhms in a light environment.

Students will build a circuit that takes advantage of the photoresistor to change the blink rate of an LED based on the amount of light received by the photoresistor. A diagram and the accompanying code are given below.




```
const int LED =13 ; // connect LED to pin 13
int lightlevel=0;  // lightlevel is the variable to store analog in
int loopindex=0;
void setup () {

  pinMode(LED, OUTPUT);  //sets the digital pin as an output
                      // note analog pins are automatically inputs
Serial.begin(9600); // set up serial comms with computer at 9600 baud }

void loop() {

  lightlevel=analogRead(0);   //sensor on analog pin 0
Serial.println(lightlevel); // print out lightlevel to user, via serial

  for (loopindex=0;loopindex<5;loopindex++) {
   digitalWrite(LED, HIGH);
   delay(lightlevel);
   digitalWrite(LED, LOW);
   delay(lightlevel);
  }                     // 5 blinks then checks light level again
                       // this is so it doesn't spew too much at ya
```

BY-NC-SA

}

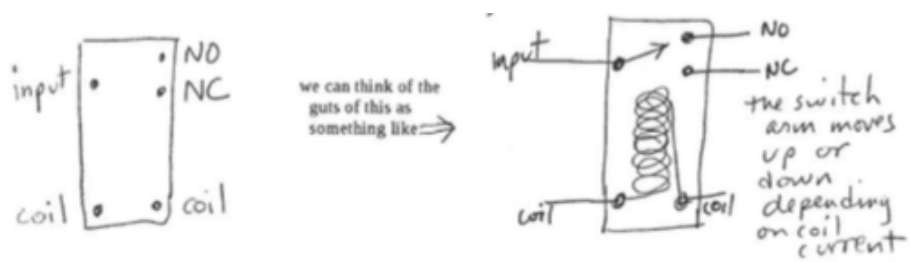
Note the use of the analogRead commands to find the intensity of ambient light.

Also note the use of ‘Serial’. This accesses the Serial Monitor, allowing the circuit to send back to the IDE the actual value of the analog input read. This can be accessed by students by clicking on the magnifying glass in the upper right-hand corner of the IDE, and shows the actual value of the light level to the user.

Ask students: What are applications of this kind of circuit?

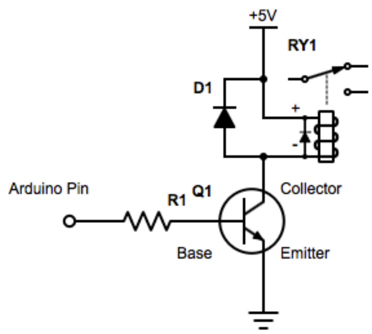
<u>Relays</u>

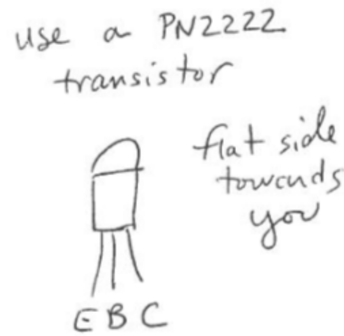
For the next circuit, students will be working with relays. Relays are a mechanical switch that uses a change in current through a metal coil to open and close other connections. Below is a picture, of a relay, showing the five pins and their uses. There are two pins for the coil, in order to put current through it, an input pin for the circuit that is being switched, and two output pins that the relay switches between, based on the coil current.



While the relay is capable of switching a lot of power, it requires a not-insignificant amount of current to produce a strong enough magnetic field with the coil to move the relay arm. This is more current that the Arduino can produce on its own. In order to effectively use the relay, students must first use a transistor to produce enough current to operate the relay. The necessary circuit is shown below. Notice the inclusion of the power rectifier diode. This is put in place in order to stop the voltage spike that is produced by turning off the current, an artifact of Faraday’s Law. For this circuit, students should use a PN2222 general purpose NPN transistor, capable of delivering 1 A of current.

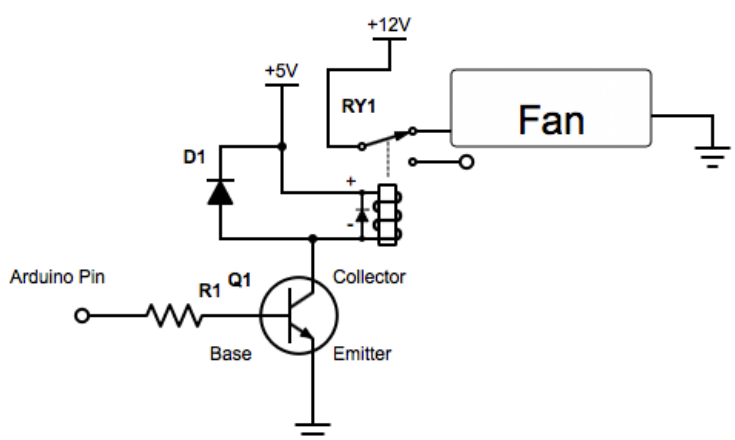
BY-NC-SA





Once students have made the circuit, instruct them to write a sketch that turns the relay on and off every few seconds. Because the relay is a mechanical switch, students should hear a click with each switch. To verify that the relay is working as expected, increase the delay to between 20 and 30 seconds, and use a DVM to check the resistance between the input pin and each output pin. (If two pins are not connected, they will have a very large resistance. If they are connected, they will have a very small resistance.)

Finally, allow students to use the relay to turn something else on. Using a bench power supply, connect the input pin to 12 V (current value is not important). Connect a 12 V fan to one of the output pins and run the sketch. Students should see the fan turn on and off every 20 to 30 seconds, depending on what time delay they wrote into their sketch. The complete circuit is shown below.



Ask students: What are applications of this kind of circuit?

(cc) BY-NC-SA

Stage 3: Team Projects

Have students work in pairs to build an Arduino circuit for a specific purpose, offering guidance as necessary.

Objectives:

3. Work as team to solve need.
    a. Identify need.
    b. Develop solution.
    c. Test solution.
    d. Make corrections to design based on testing.

Stage Length: 8-11 hours

Break steps up as necessary over class periods.

Teacher's Instructions

Have students sit in pairs. They will be working together as a group to complete the project.

For this project, groups will brainstorm a problem to fix with an Arduino circuit, identify necessary criteria and constraints for the circuit, and build an appropriate circuit. Groups will then test their circuits, determine any changes that could be made to improve the app, and redesign their circuit based on these findings. In the end, each group will present their completed circuit, describe how it was originally created, and what was changed after testing.

Throughout process, emphasize importance of thinking both creatively about what could be done with circuitry and practically about what tools would be most effective for realizing that vision.

Steps:

Brainstorm/research ideas.
Give groups 1-½ hours to brainstorm problems that they would like to attempt to address. Have groups make a list of these problems. By the end of this day, each group must decide which problem to base their project off of.

Identify criteria and constraints.
Give groups 40 minutes to identify the constraints that their circuit must adhere to in order to work practically. Key information for groups to identify are what the user will do to make the circuit work (input), how the circuit will respond (output), and what components will be necessary for the circuit. When groups are finished, have them discuss with the teacher to insure their ideas are reasonable.

BY-NC-SA

Develop and test Arduino circuit.
Give groups between 4 and 5 hours in which to develop a prototype Arduino circuit and test its practicality. Students should take notes on how they create the circuit, including what components they used, why they used them, and what steps they took to insure safe operations. They should also take notes on the strengths and weaknesses of their circuit when tested. These details will be necessary for their final presentation.

Re-assess circuit and develop new model.
Give groups between 1 and 2 hours to take their original design and their notes from testing, groups should build circuits to improve practical functioning.

Create owner's manual.
Once groups have finished creating their Arduino circuits, give students between 1 and 2 hours to put together an owner's manual. This manual should be designed to help users understand the purpose of the circuit and how to get the circuit to carry out its purpose. Especially include any limitations to the circuit's ability to perform as expected.

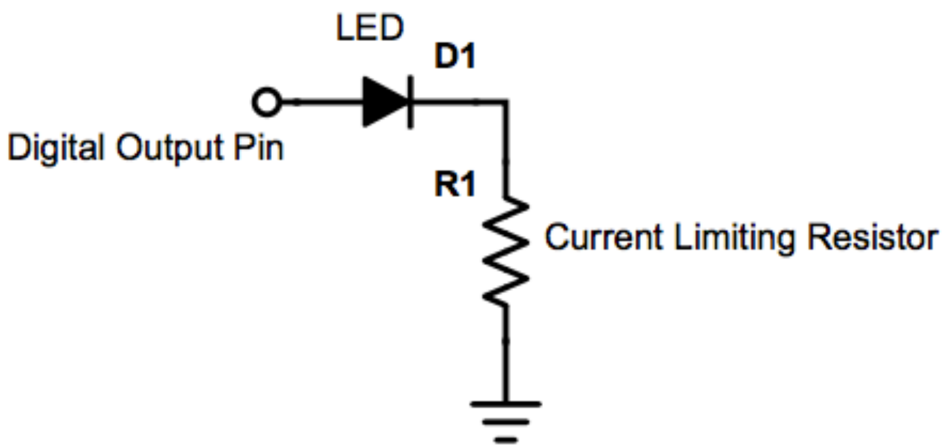
Present.
Give groups 40 minutes to put together notes on their work to create presentation. Groups should present their problem, their Arduino circuit, how they designed their circuit, and what improvements they made to their circuit after testing. This should include how they used what they learned throughout stages 1 and 2. This time does not include time to present the projects, as that will depend on the number of groups in a class.

(cc) BY-NC-SA

Arduino Circuit Diagrams and Code Sketches

1st Circuit: Basic LED

Diagram:
Remember - LED has two legs of different lengths. The shorter one must be closer to ground or it won't work!



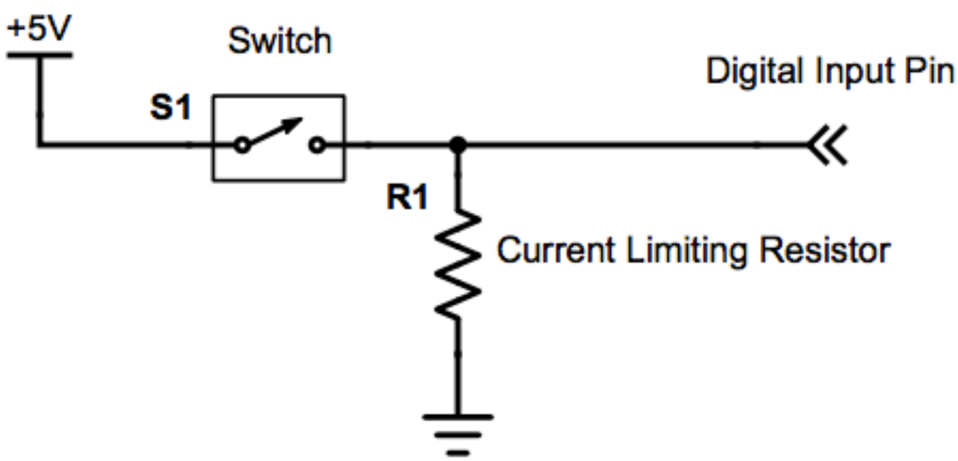
Code:
```
const int LED = 13; // connect LED to pin 13

void setup ()
{
    pinMode(LED, OUTPUT); // sets the digital pin as an output

}

void loop ()
{
   digitalWrite(LED, HIGH);
   delay(1000);
   digitalWrite(LED, LOW);
```

(cc) BY-NC-SA

```
  delay(1000);
}
```

2nd Circuit: Switch

Diagram:



https://www.youtube.com/watch?v=kqYRZG3FfiQ

Code:

```
// two blinky LEDs with a switch to control them
        const int LED =13 ; // connect LED to pin 13
        const int LED2=5;   // connect second LED to pin 5
        const int SWITCH=2;
        int val=0;
void setup () {
        pinMode(LED, OUTPUT);  //sets this digital pin as an output
        pinMode(LED2,OUTPUT); //sets the digital pin as an output
        pinMode(SWITCH,INPUT); //sets pin 2 as a switch digital input
}
void loop()
        {
        val=digitalRead(SWITCH); //decide if switch pin 2 is hi or lo
        if (val==HIGH) {       //   if its hi, blinky pattern
        ... insert your code for whatever blink pattern you like ...
}
        else {                  //if its lo, shut the LEDs off
        digitalWrite(LED,LOW);
        digitalWrite(LED2,LOW);}
```

BY-NC-SA

}

3rd Circuit: LED with Pulse Modulation

Code:

```
const int LED =3 ; // connect LED to pin 3
int i=0;
void setup () {

        pinMode(LED, OUTPUT);  //sets the digital pin as an output
        }
void loop() {

        for (i=20; i<255; i++)
        {  analogWrite(LED, i);
delay(15); }

        for (i=255; i>20; i--)
        {  analogWrite(LED,i);
        delay (15);}
        }
```

4th Circuit: Buzzer

Code:

```
const int PZ = 8 ; // connect PZ buzzer to pin 8

void setup () {

        pinMode(PZ, OUTPUT);  //sets the digital pin as an output
        }
void loop()

  {
        tone(PZ,1760); // replace PZ with the number of the pin it is in.  The other
number is
        delay(1000);          // the frequency
        tone(PZ,880);
        delay(1000);
        noTone(8);   // the PZ remains on until it is turned off.
        delay(1000);
}
```

Try to make the PZ play a tune. Note pitches are found here https://www.arduino.cc/en/Tutorial/toneMelody.

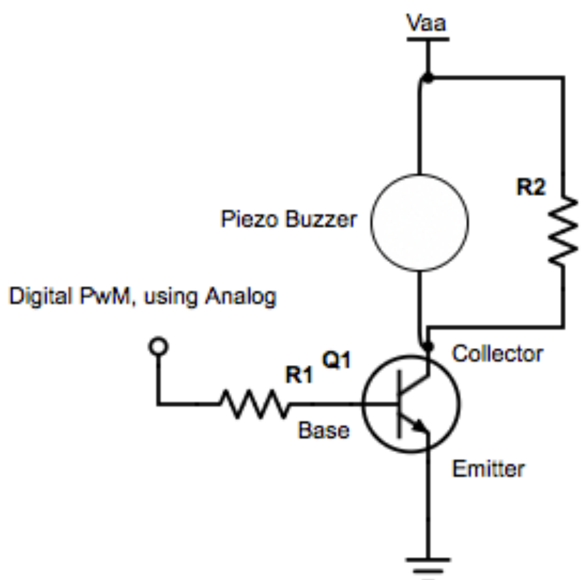
To use the PZ without a specific frequency, use the following.

BY-NC-SA

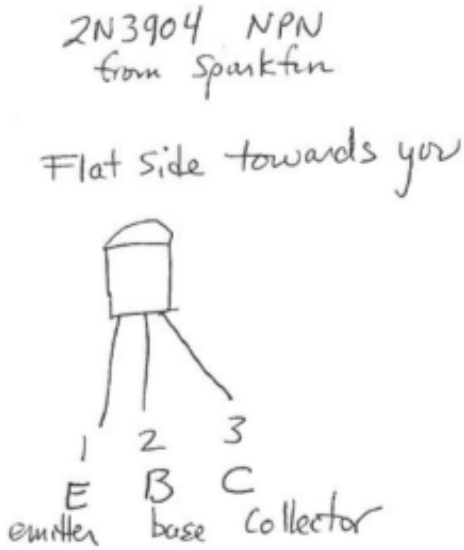
```
void loop() {

        analogWrite(PZ,128);
        delay(time);
        digitalWrite(PZ,LOW);
        delay(time);
}
```

5th Circuit: Buzzer with Transistor

Diagram:



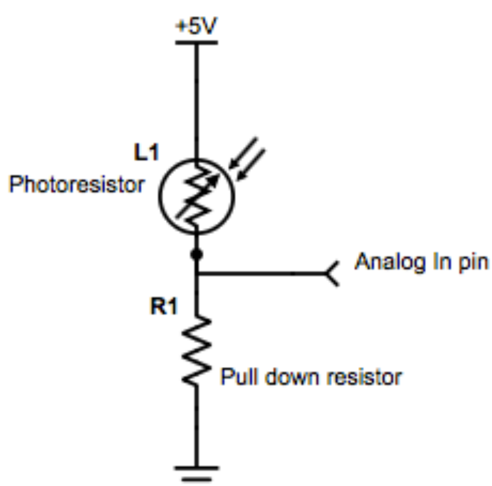
(cc) BY-NC-SA



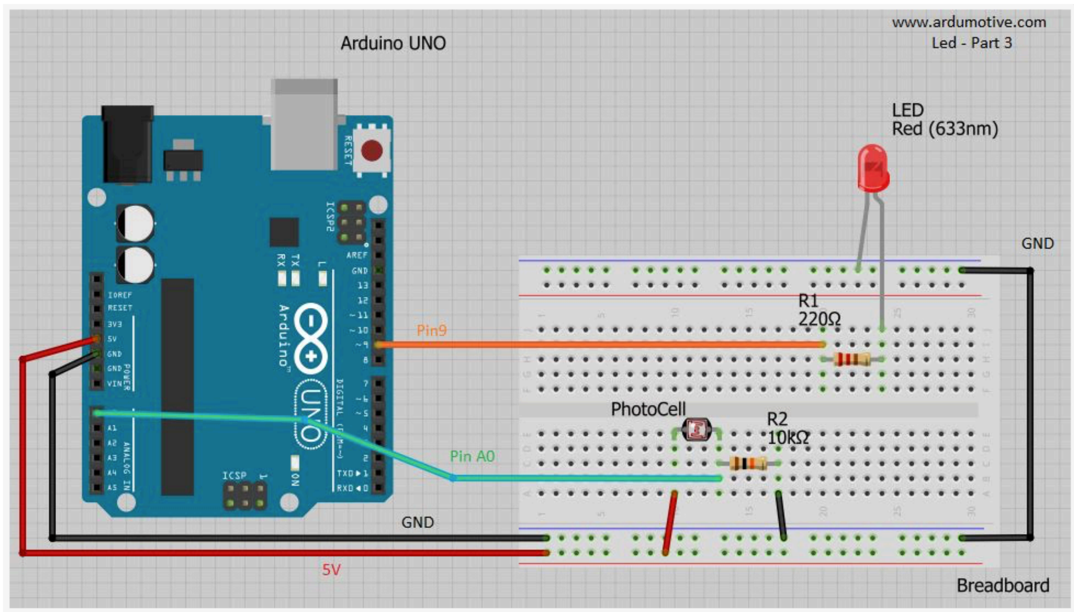
Code: Same as previous Buzzer code

6th Circuit: Light Sensor (Photoresistor)

Diagram:



(cc) BY-NC-SA



http://www.ardumotive.com/how-to-use-a-photoresistor-en.html

Code:

```
const int LED =13 ; // connect LED to pin 13
int lightlevel=0;  // lightlevel is the variable to store analog in
int loopindex=0;
void setup () {

  pinMode(LED, OUTPUT);  //sets the digital pin as an output
                     // note analog pins are automatically inputs
Serial.begin(9600); // set up serial comms with computer at 9600 baud(unit of measurement)

 }

void loop() {

  lightlevel=analogRead(0);   //sensor on analog pin 0
Serial.println(lightlevel); // print out lightlevel to user, via serial

  for (loopindex=0;loopindex<5;loopindex++) {
        digitalWrite(LED, HIGH);
        delay(lightlevel);
        digitalWrite(LED, LOW);
        delay(lightlevel);
  }                   // 5 blinks then checks light level again
                     // this is so it doesn't spew too much at ya
```

BY-NC-SA

```
}
```

CC BY-NC-SA