Fun, Innovative Computer Science Activities for the Classroom and Outreach

Dr. Stephany Coffman-Wolph, West Virginia University Institute of Technology

Dr. Stephany Coffman-Wolph is an Assistant Professor in the department of Computer Science and Information Systems at West Virginia University Institute of Technology (WVU Tech). Stephany is actively involved in community outreach with a goal of increasing the number of women in STEM and creating effective methods for introducing young children to CS concepts and topics. She is a founding member and co-Adviser of AWESOME (Association for Women Engineers, Scientists, Or Mathematician Empowerment) a student organization at WVU Tech. Her other research interests include: Artificial Intelligence, Fuzzy Logic, and Software Engineering.

Innovative Computer Science Activities for the Classroom and Outreach Events

Abstract

Teaching a freshman-level introductory course in computer programming can be challenging. Although most college students are familiar with computer science, they seem to be unaware of what being a computer scientist means. These activities help them understand the depth and diversity a computer science undergraduate degree can entail. All the activities outlined below do not require a computer lab, are extremely cost-effective, and require minimum preparation. Additionally, the activities are easy to do with a variety of age groups and various number of students and are specifically designed not to require students to have any prior computer experience. The main goal is to introduce various computer science topics using fun physical activities and everyday experiences that are familiar. The paper will provide the instructions for each of the activities and the learning objectives. The activities included are: (1) Loops with Music, (2) Network Topology and Problem Solving, (3) Linked List with Yarn and Paper Bags, (4) Sorting Algorithms with Paper Bags, and (5) Recursion Introduction: Simple Tower of Hanoi with Colored Paper.

Introduction and Background

Faculty members of West Virginia University Institute of Technology (WVU Tech) are encouraged to try new ideas in the classroom and participate in the ongoing outreach activities on and off campus. As a faculty advisor to a student organization dedicated to recruiting and retaining future generations of females into the STEM fields (Association for Women Engineers, Scientists, or Mathematician Empowerment also known as AWESOME), there have been numerous occasions to participate in outreach events. This led to the development of fun, interactive, easily transportable, and low cost activities to promote computer science to young kids (e.g., Daisy Scouts/Cub Scouts) through college students. These activities are designed to be done without computers as access to a computer lab is often impractical, simply not available, or not easily transportable.

Today with the popularity of smart phones and the increasing use of tablets in classrooms, most kids, even very young kids, have interacted with a computer at some point. The activities described in this paper are designed to go beyond simply teaching someone to program and are focused on the concepts behind programming and those foundational concepts within the field of computer science. The end goal of these activities is to introduce students to the "science" of computer science. Most careers in computer science go beyond simply sitting around programming in a specific language and require many other skills (problem solving, logical reasoning, critical thinking, etc.). All activities outlined below are programming language independent but could be tailored to whatever specific programming language you wish (or could be done using generic pseudo-code).

It has been demonstated[4, 5] that hands-on activities for computer science topics increase both student awareness and interest in the field of computer. The development of these computer-free hands-on activities for computer science topics was inspired by the Computer Science (CS)

Unplugged website[1, 2]. CS Unplugged[1, 2] provides videos, worksheets, and teaching guides to a variety of computer science activities. All the CS Unplugged[1, 2] activities are "universal" computer topics (and, thus, not language specific or even programming specific).

This paper is an expansion and continuation of [3] and includes general computer science concept activities, as well as presenting the details for activities focused on intermediate-level general programming concepts. "Network Topology and Problem Solving" was directly inspired by the CS Unplugged[1, 2] activity entitled "Routing and Deadlock" and specifically their use of "The Orange Game". The "Network Topology and Problem Solving" activity uses "The Orange Game" as a warm-up round before proceeding into the more complex ideas. "Sorting Algorithms with Paper Bags" was also inspired by the CS Unplugged[1, 2] activity for searching – but instead of using cards with numbers and volunteers, this activity uses labelled lunch bags. The "Linked List with Yarn and Paper Bags" was a spin-off activity after development of the "Sorting Algorithms with Paper Bags" activity for demonstrating sorting algorithms with paper bags.

Loops with Music

Learning Objectives: After this activity, students should be able to:

- Explain, to another student, what a loop is
- Explain, to another student, what a loop condition is
- Define, in their own words, a counter

The primary goal of this activity is to demonstrate various types of loops and loop conditions. This activity was successfully used within two introductory computer science lecture sections. All students in the classroom get to be participants in the activity by either clapping or snapping their fingers along with the music. The details and instructions for this activity are supplied in Appendix A.

Network Topology and Problem Solving

Learning Objectives:

- Define, in their own words, a network
- Define, in their own words, a network packet
- Define, in their own words, a network topology
- Explain the difference between white hats, black hats, and gray hats
- Explain the importance of network topology (high school and up)

This activity is designed to teach students basic networking concepts and terminology: computer network, node, topology, black hats, gray hats, and hacking. All students will have the opportunity to play a node within the network. The activity can be done with almost any size group and generally works better with older students. The materials, required preparation, instructions, and alternatives are provided in Appendix B. As mentioned earlier the first part of this activity was inspired by "The Orange Game" from CS Unplugged [2] and uses it as a warm up exercise. This is a great outreach activity to have students participate in after they have been sitting working on individual tasks and can be easily tailored to various time lengths.

Linked List with Yarn and Paper Bags

Learning Objectives:

- Define, in their own words, a data structure
- Define, in their own words, a linked list
- Explain the steps to add a node to a linked list (front or back)
- Explain the steps to delete a node from a linked list (front or back)

In this activity, students learn about the basic data structure linked list. Many students struggle to develop a mental picture of a linked list which increases the difficulty of them understanding the standard methods associated with this data structure (e.g., add to front of list, add to end of list, delete from front of list, delete from back of list, etc.). This activity is designed to provide that mental image for students and demonstrate the required steps for the essential methods needed to properly implement linked lists. See Appendix C for the details needed to implement this activity during a lecture for students studying an introduction to basic data structures.

Sorting Algorithms with Paper Bags

Learning Objectives:

- Define, in your own words, an algorithm
- Explain, to another student, what it means for a list to be sorted
- Define, in your own words, a sorting algorithm
- Explain the various ways to sort data
- Explain the mechanics of the selection sort
- Explain the mechanics of the bubble sort
- Practice algorithm design

Sorting algorithms are used prolifically throughout computer programming and are often one of the major categories of algorithms taught to introductory students. Additionally, this is often the point in young programmers' development that Big-O and basic algorithm analysis is presented. Therefore, this is an important concept for students to have a thorough understanding of the way the algorithms function. The bags represent each value within the data structure and are easy to move around to mimic the movement described within the algorithm. Appendix D provides the reader with the necessary tools to perform this activity in a classroom. Additionally, this activity is specifically designed to help student' practice developing and designing an algorithm.

Recursion Introduction: Simple Tower of Hanoi with Colored Paper

Learning Objectives:

- Define, in your own words, recursion
- Explain, to another student, the basic process of recursion within programming
- Explain, to another student, the methodology behind solving Tower of Hanoi
- Practice algorithm design

Recursion is a challenging concept for many students and they express frustration when the concept is first introduced to them. Students often find recursion counterintuitive, particularly the part regarding the method being incomplete and another version of the method being pushed onto the program execution stack. This activity is designed to provide students with hands-on experience and a visual reference for the concept. The Tower of Hanoi is a classic example that many textbooks use for a discussion of the topic of recursion for programming. Appendix E provides the materials, prep, and instruction details for implementing this activity. Additionally, this activity is specifically designed to help students' practice essential critical thinking and algorithm design skills.

Table 1: Activities, Group Sizes, Group Level, and Settings

Activity	Student Group Sizes and Level	Setting (class, outreach, etc.)
Loops with Music (Appendix A)	45 college students 36 college students	Class, Individual Desks Class, Tables that seat 4
Network Topology and Problem Solving (Appendix B)	6 college students 18 high school students 40+ high school students	Class, Moved chairs into a circle STEM Camp, Students stood STEM Camp, Large empty room
Linked List with Yarn and Paper Bags (Appendix C)	34 college students 22 college students	Class, Tables that seat 2 Class, Tables that seat 4
Sorting Algorithms with Paper Bags (Appendix D)	40 college students 45 college students 36 college students	Class, Individual Desks Class, Individual Desks Class, Tables that seat 4
Recursion Introduction: Simple Tower of Hanoi with Colored Paper (Appendix E)	34 college students 22 college students	Class, Tables that seat 2 Class, Tables that seat 4

Implementations of the Activities

The above activities have been successfully used during various classroom and outreach events as summarized in Table 1. The "Loops with Music" is a fun end of the lecture activity with introductory computer programming classes and a nice way to wrap up the introduction of the topic. The "Linked List with Yarn and Paper Bags", "Sorting Algorithms with Paper Bags", and "Recursion Introduction: Simple Tower of Hanoi with Color Paper" activities have been successfully used during the second introductory computer programming courses to teach the concepts of algorithm design, sorting, linked lists, and recursion. In addition to the computer science knowledge, students have gained valuable teamwork skills. Most importantly, students have expressed enjoyment of these activities. College students have particularly commented on how much these activities have assisted in their understanding and ability to remember the concepts. This parallels the findings in publications [4, 5] regarding both elementary and middle school students.

The "Network Topology and Problem Solving" activity was a successful and popular activity during the first night of a weeklong summer STEM camp and during an introductory Information

Systems course. It was not only used as an ice breaker on the first evening but also allowed them to practice teamwork, problem solving, and team based problem solving skills. Given the large group of students, the discussion of how changing the network topology to speed up the process was very effective as there was time to do multiple rounds of slowly combining groups (most groups start at 3 or 4 and double in size until there are 9-10 per group).

Student Feedback

The author has not collected quantitative data regarding the activities. Informally, students have provided positive feedback for the activities. This is particularly true regarding remembering a given topic later. Several students have found it difficult to connect the time in lab with the time in a lecture setting. The specific topic name might not be familiar to them, but when reminded of the activity in question they instantly remember and can continue working on their assignments. Additionally, faculty evaluations for courses that use these and similar fun activities get enthusiastic student comments (e.g., "Great course with lots of interactive activities").

Concluding Remarks and Future Work

As the author continues to teach introductory computer science courses, participate in outreach events, and work during STEM based science camps, these activities will be repeated, updated, expanded, and improved. Additionally, new activities or add-ons will be designed to cover even more topics from computer science. Each experience with these activities has allowed the author to continually improve and expand the activities for a wider age and experience range. Additionally, the author plans to extend several of these activities to cover more advance computer science topics. For example, with the "Network Topology and Problem Solving" activity, have multiple types of white hats each labeled to demonstrate that there are different types of nodes within a network and discuss the role of each. Another example is expanding the "Sorting Algorithms with Paper Bags" to cover more complex sorting algorithms and more complex data structures. The "Linked List with Yarn and Paper Bags" could be easily be extended to cover not only doubly linked lists but circular linked lists as well. The author chose to focused early iterations of using these activities on creating, implementing, and improving the activities. In the future, the author plans to collect data on student opinions of these activities and analyze grades from exams/homework/labs to improve the student experience and maximize their learning.

Bibliography

[1] T. Bell, et al., "Computer Science Unplugged: School Students Doing Real Computing Without Computers," Computing and Information Technology Research and Education, New Zealand (CITRENZ), vol. 13, no. 1, pp. 20-29, 2009.
[2] T. Bell, et al., CS Unplugged: Computer Science without a Computer. www.csunplugged.org., 2015.
[3] S. Coffman-Wolph, "Innovative Activities to Teach Computer Science Concepts Inside the Classroom and at Outreach Events", Proceedings of the 123rd Annual ASEE Annual Conference & Exposition, June 26-29, 2016, New Orleans, LA, pp. 25715.
[4] L. Lambert and H. Guiffre, "Computer Science Outreach in an Elementary School," Journal of Computing Sciences in Colleges, vol. 24, no. 3, pp. 118-124, 2009.

[5] R. Taub, et al., "The Effect of CS Unplugged on Middle-School Students' Views of CS," Proceedings of the 14th annual ACM SIGCSE Conference on Innovation and Technology in Computer Science Education, July 06-09, 2009, Paris, France, pp. 99-103.

Appendix A: Instructions for Looping with Music

Materials:

- A device that will play music
- (Optional) A speaker if the room is large

Prep:

- Select several songs that have a strong beat (this will help students clap or snap along with the music and help them count during the for loops)

Activity Instructions:

1. Divide the students in the room into two halves. Clappers and Snappers.
2. Start the music. Tell the students that when your right hand is up the clappers clap and when your left hand is up the snappers snap. Do various combinations (including having both hands up).
3. Stop the music. Explain how this relates to conditions on a while loop.
4. Start the music and have the students immediately start clapping and snapping. Tell the students when your right hand is up they should stop clapping and when you left hand is up they should stop snapping.
5. Stop the music. Explain how this relates to a do while loop.
6. Start the music. Count out the beats. Tell the students that they are going to clap for X number of beats, then stop. (Repeat with various number of beats).
7. Stop the music. Explain how this relates to loops that have a counter – for loops.

Alternatives:

- If snapping is too difficult for the students, you can also have them tap their feet or do another movement.
- After explaining various loops, if there is time, you can show the programming code or pseudo-code. If you are working with older students, you can have them try to create the pseudo-code for each loop.
- For older students or if time, you can show how you can write equivalent for, do while, and while loops.

Appendix B: Network Topology and Problem Solving

Materials:

- Pairs of matching shaped erasers, pencil toppers, small toys, etc. (Anything that easily fits within the palm of your hand)
- Random other erasers, pencil toppers, or small toys
- Baskets, buckets, or plastic sandwich bags
- One or two black hats, one or two gray hats, Numerous white hats

Prep:

- Pre-sort the matching pairs of erasers, pencil toppers, small toys, etc.
- Create the Packets: Divide these items into groups of 3-5 and place into the baskets, buckets, or plastic sandwich bags

Activity Instructions:

1. Divide the students into small groups (3-5).
2. Using a packet that is of the corresponding group size, randomly distribute two items to each participant in the group except the final person who only gets one item. (Note: You should have one left over in the bag).
3. Begin "The Orange Game": The goal is for each participant to have two matching items. The challenge, you can only ever hold two items and you can only pass to the person immediately on your left or right. Pass and switch around items until everyone has two matching items (except the person who has an open hand).
4. Combine two groups. Repeat "The Orange Game".
5. Discuss how much more difficult it is with a larger group. Ask groups to figure out possible solutions to make it easier. Discuss these ideas.
6. Pass out one white hat to each group. Have this person act as a router or switch for the group. Repeat "The Orange Game" but allow the users to also pass to the router/switch. (The router/switch can pass to anyone within the group).
7. Discuss how to change the network topology to improve the speed of the process.
8. Gather all the group packets, but have the participants stay in with their groups. Distribute two random items to each participant. The router/switch function as they did last time, but they can also move to speak with other router/switches. Repeat "The Orange Game" with these modifications.
9. Repeat the last experience, but change or add one or two "black hat routers/switches" and "gray hat router/switches". (The black hats make things harder by stealing packets or giving out the extra packets. The gray hats randomly do things correct or evilly).

Alternatives:

- For the black and gray hat routers/switches, use either people you prep or tell them their roles in secret. Thus, it is a surprise to all the other participants.
- Switch around routers/switches between rounds
- For extremely large groups or if you have time, you can have the students try out other network topologies

Appendix C: Linked Lists with Yarn and Paper Bags

Materials:

- Paper lunch bags
- Yarn or string
- Scissors
- Hole puncher

Prep:

- Pre-cut the yarn into lengths of 12-15 inches
- Use the hole puncher to create holes on each side of the lunch bags

Activity Instructions:

1. Begin by describing what a data structure is
2. Describe what a node is (i.e., data and a link to the next node)
3. Explain that a bag represents the data and the yarn represents the link
4. Start with an empty linked list, added a node to the list. Point out that when the list is empty, adding to front or back is the same.
5. Add another node to the front of the list. Tie the "next" link to the node already in the linked list. (Repeat if needed).
6. Add a node to the back of the list. Tie the last node in the linked list to the new node. (Repeat if needed).
7. Delete a node from the front of the list: cut the yarn to remove the first node. (Repeat if needed)
8. Delete a node from the back of the list: cut the yarn to detach the linked list from the node being deleted. (Repeat if needed)

Alternatives:

- Add a node card labeled null to the end of the yarn. Remove when the link becomes attached to another node
- You can create paper arrows and attach to the yarn so students understand the one direction of a linked list
- You could expand this activity to include adding or deleting a node from the middle of the linked list

Appendix D: Sorting Algorithms with Paper Bags

Materials:

- Paper lunch bags
- A dark marker
- Chalkboard, Whiteboard, Computer connected to Overhead Projector, or Large Paper Flip-pad

Prep:

- Label each bag with a "random" number

Activity Instructions:

1. Set up the paper lunch bags in front of the students so that everyone can see. (Make sure to randomize the order)
2. Divide the students into groups
3. Ask the students to develop an algorithm to sort the numbers in ascending order. (Possible hint: Decide a scheme or a sequence of repeatable steps to sort the numbers. You want something that is a pattern)
4. Have each group share their scheme and walkthrough the scheme with the paper bags. Discuss which known sorting algorithm their scheme resembles or what the group did not include
5. Demonstrate about 50% of the selection sort using the bags. Write the algorithm for selection sort. Demonstrate the remaining 50% of the sort while comparing to the written algorithm

6. Demonstrate about 50% of the bubble sort using the bags. Write the algorithm for bubble sort. Demonstrate the remaining 50% of the sort while comparing to the written algorithm

Alternatives

- This activity could be expanded to demonstrate any of the many sort algorithms
- For older students, after completing the demonstration, this would be an ideal time to discuss the Big-O analysis for each algorithm

Appendix E: Recursion Introduction: Simple Tower of Hanoi with Colored Paper

Materials:

- Plain white copy paper
- Plain white legal-size copy paper (optional)
- 4-5 different colors of copy paper (dependent on age and length of activity)
- Scissors
- A dark color pin or marker

Prep:

- On a piece of white copy paper, draw three equal size squares (or use a computer to create this). Copy so there is enough for each group to have one. (The legal-size paper might be best). This will be used to represent the pegs
- Take one of the colors of the copy paper, cut out circles that fit inside the squares. Cut enough for each group to have one. This will be used to represent the largest disks
- Take a different color of copy paper, cut out slightly smaller circles than previously. Cut enough for each group to have one. This will be used to represent the next size of disks
- Repeat with remaining colors, each time getting slightly smaller circles. These will be used to represent all the other disks

Activity Instructions:

1. Divide the students into groups and hand out the sheet for the pegs and one of three different color disks
2. Instruct the students to stack the color disks in order with the largest on the bottom and smallest on top. Place the ordered disks onto the square/peg furthest to the left
3. Explain the general concept behind Tower of Hanoi, the end goal (i.e., move all the disks to peg 3), and the general rules (e.g., cannot place a larger disk onto a smaller disk).
4. Have the students experiment and see if they can solve the puzzle
5. Once most groups have “solved” the Tower of Hanoi, have them discuss it algorithmically. They should try to write down the steps (and work towards writing a concise and general algorithm)
6. Hand out a fourth disk. Ask the students to verify their algorithm is correct or make changes. (If time, also hand out a fifth disk and repeat the process)