Introduction

Welcome!

This guide book will give you everything you need to prepare your students for the Firia Labs Jumpstart/Explorer and Python with Robots coding kits. If your students are new to computational thinking, start here!

For many students and teachers, this is their very first exposure to computer science. If that's your situation, don't worry! We've designed the introduction to give students a good foundation in computer science terms and concepts.

Use these lessons the first weeks of school to get to know your students better and help them see the relevance and real-world applications of computer science. It's important that students a) don't feel intimidated and b) see relevance in what they're learning. These lessons are meant to help you establish a good rapport and instill a growth mindset with your students.

Lesson: What is a Computer Scientist?

Intro and Discussion Points:

Students need to be confronted with preconceived notions and stereotypes about what a computer scientist is. This is a remix of the classic “Draw a Scientist” lesson.

Preparation and Materials:

- Chart paper or post-its and markers
- Person outline (below)
- Drawing supplies, such as colored pencils or crayons

Timeframe:

1 class period

Student Learning Targets:

- I can describe many diverse careers in Computer Science.

Project Goals:

- ➢ Understand that Computer Science careers encompass many industries.
- ➢ Understand that there is no one “type” of person that makes a computer scientist.
- ➢ Understand that as CS students, THEY are computer scientists, too!

Lesson Sequence:

- ➔ Pre-assessment: give students the blank outline (or plain paper) and ask them to draw whatever comes to mind when they think of a computer scientist. Some students may prefer to write adjectives instead of drawing.
- ➔ Discuss their ideas, and compile them on chart paper or post-it notes on your board.
- ➔ Show videos from the Get Inspired section of the Code.org YouTube channel, such as “Code Stars” or “Computer Science is Changing Everything.”
- ➔ Post-assessment: ask students to repeat the intro activity or give them time to discuss in groups how their ideas have changed. Come back together as a whole group and discuss how their ideas changed.
- ➔ If time, share a picture book such as *Grace Hopper: Queen of Computer Code (People Who Shaped Our World)* or *Ada Byron Lovelace and the Thinking Machine* by Laurie Wallmark or *Margaret and the Moon* by Dean Robbins (or any story that challenges traditional stereotypes.

Lesson: PB&J Algorithm

Intro and Discussion Points:

Computer Scientists use specific language, just like any industry. It's important for students to be able to use correct terminology, but also relate it to things they already know. Computers are not intelligent! They can only follow instructions literally, step-by-step, in order.

Preparation and Materials:

- Instructions outline
- Loaf of bread (may need 1 to 1.5 per class)
- Peanut Butter (Have a backup in case of allergies - plain butter works!)
- Jelly
- Paper towels, baby wipes, etc. to clean up.

Timeframe:

1 class period

Student Learning Targets:

- I can write step-by-step instructions.
- I can describe what an algorithm is.
- I can relate an algorithm to a recipe.

Project Goals:

- ➢ Understand and use the term algorithm.
- ➢ Understand that a computer is not "intelligent" and will only do EXACTLY what you tell it to do.
- ➢ Write repeatable instructions that another user can understand and execute.

Lesson Sequence:

- ➔ Students write out step-by-step instructions, and may include diagrams if necessary.
- ➔ Teacher executes instructions EXACTLY. (The more literal, the better.)
- ➔ Give students instructions back to revise (debug).
- ➔ Show Literal Instructions Challenge video
- ➔ Review the technical terms algorithm and debug.

Lesson: Hide & Seek

Intro and Discussion Points:

As discussed in the PB&J challenge, computers can only follow instructions EXACTLY. In this lesson, students will work with a partner. One will act as the programmer and the other as the computer.

Preparation and Materials:

- Flowchart poster
- A classroom, hallway, or gymnasium with tiles (grid)
- Objects such as Happy Meal toys, matchbox cars, etc. that are small enough to easily place on the floor.

Timeframe:

1 class period

Student Learning Targets:

- I can write an algorithm in steps and/or as a flowchart.
- I can follow an algorithm written in steps and/or as a flowchart.

Project Goals:

- ➢ Understand and use the term algorithm.
- ➢ Understand that a computer is not “intelligent” and will only do EXACTLY what you tell it to do.
- ➢ Write repeatable instructions that another user can understand and execute.
- ➢ Use flowcharting to write an algorithm.

Lesson Sequence:

- ➔ Pair students up, and assign one to be the programmer and one to be the computer.
- ➔ The programmer will choose a toy and place it somewhere in the room.
- ➔ From an assigned starting point, the programmer will use the commands move forward, turn left, and turn right to lead the computer to their toy. (Note- turn means turn IN PLACE, not turn and take a step left or right.)
- ➔ The programmer will hand their program to the computer, who will follow the steps. Together, the programmer and computer can debug and reset.
- ➔ Swap roles.
- ➔ Introduce the term “flowchart” as a documentation and planning process. (Clip: The Friendship Algorithm from *The Big Bang Theory*). Discuss and have students copy the basic flowcharting symbols from the poster, or give them a copy for their notebooks. (Note- there are more symbols for more advanced programs, but these are a good place to start.)
- ➔ Have students repeat the process, but use a flowchart instead of list.
- ➔ Review the terms algorithm, debug, and flowcharting symbols.

Lesson: Coding Relay Race

Intro and Discussion Points:

Computer Scientists work as a team, and must rely on each other to find and point out one another's mistakes. Debugging is not a solitary activity!

Preparation and Materials:

- Large space
- Grid paper program diagrams, laminated
- Small dry erase boards
- Expo markers

Timeframe:

1 class period

Student Learning Targets:

- I can work on a team to follow and debug a program.

Project Goals:

➢ Understand that precision and communication are important aspects of computer science.

Lesson Sequence:

➔ Establish or review the symbols you will need to draw the grid images (left, right, forward, backward, color in)
➔ Divide students into groups of 3-4, determine who will be team captain, and line up relay-race style.
➔ Place the image on the other side of the room from each team.
➔ The first student will run over to the image and write down the first symbol in the program needed to create that image.
➔ The first student then runs back and passes the expo marker to the next person in line.
➔ The next person in line runs to the image, checks the program that has already been written, then either debugs the program by crossing out an incorrect symbol, **or** adds a new one. Each student can only do ONE action each time, so they cannot cross out AND correct the bug.
➔ That student runs back to the next person in line and tells them how to fix the bug, or to move on to the next step.
➔ When all steps are complete, the captain checks the program one last time and declares it done.
➔ Points are awarded as follows: 2 points for being done first AND everything is correct, 1 point for having everything correct, but not completed first, 0 points if there are bugs.
➔ Repeat for each image.

Lesson: LEGO Maze Algorithms

Intro and Discussion Points:

As discussed in the PB&J challenge, computers can only follow instructions EXACTLY. In this lesson, students will work with a partner to find their way through a maze and check each others' work.

Preparation and Materials:

- Flowchart poster
- LEGO mazes
- LEGO mini-figs
- LEGOs (optional)

Timeframe:

1 class period

Student Learning Targets:

- I can write an algorithm in steps and/or as a flowchart.
- I can follow an algorithm written in steps and/or as a flowchart.

Project Goals:

- ➢ Understand and use the term algorithm.
- ➢ Understand that a computer is not "intelligent" and will only do EXACTLY what you tell it to do.
- ➢ Write repeatable instructions that another user can understand and execute.
- ➢ Use flowcharting to write an algorithm.

Lesson Sequence:

- ➔ Optional set-up, using pre-made or a blank maze, students will use LEGOs to make the wall of the maze. (Otherwise, dark gray boxes represent the maze "walls.")
- ➔ Place mini-fig on Start, facing North. It is important everyone starts facing the same direction.
- ➔ From the assigned starting point, use the commands move forward, turn left, and turn right to lead the mini-fig to the End space. (Note- turn means turn IN PLACE, not turn and take a step left or right.) Students may write in "pseudocode" or in flowchart depending on your preference.
- ➔ As they finish, partner up to check each other's work and debug.
- ➔ Extension - Can you find the most efficient path? The least efficient path without retracing steps?
- ➔ Review the terms algorithm, debug, and flowcharting symbols.

Lesson: Binary Beads

Intro and Discussion Points:

Computers send electrical information as "off" and "on," just like a light switch (point out the 1 and 0 on light switches, power buttons, etc.). When computers represent information using only two options, it's called "Binary." ASCII (American Standard Code for Information Interchange) is the most common format for text files in computers and on the Internet. In an ASCII file, each alphabetic, numeric, or special character is stored as an 8-bit binary number (a string of eight 0s or 1s).

Preparation and Materials:

- ASCII binary chart (found online, or use the Code.org one here.)
- Plain paper
- Keyring or bracelet making supplies, such as perler or pony beads and string, pipe cleaners, or key rings.

Timeframe:

1 class period

Student Learning Targets:

- I can explain how information is sent using 1s and 0s.

Project Goals:

➢ Understand how computers send electrical signals as 1s and 0s, which are converted into the letters, numbers, and symbols we read.

Lesson Sequence:

➔ Introduce binary with Binary Numbers - Math Bites with Danica Kellar.
➔ Show students a binary conversion chart for letters and have them write out their name in binary lengthwise on a sheet of paper.
➔ Using two colors of beads, students will lay out the beads that correspond to the binary code.
➔ Cut crafting string to the proper length. If looping on a keyring, double the length.
➔ Thread beads onto string and knot tightly. Add keychain to bookbag or wear the new bracelet!

Lesson: Digital Escape Room

Intro and Discussion Points:

"Who Let the Bugs In? Oh no! Our computer programs are buzzing, and beeping but not like they should. Bugs, real bugs, have gotten in and messed everything up! Please help rewrite the programs, beat the bugs and Break Out!"

Appropriate for CS students of all ages, this is a fun way to review computational thinking skills covered in the Unplugged unit.

Preparation and Materials:

- Teacher link: https://platform.breakoutedu.com/game/going-buggy
- Student link: https://platform.breakoutedu.com/game/play/going-buggy-78

Timeframe:

1 class period (45 min)

Student Learning Targets:

- I can work on a team to decipher the combination locks.

Project Goals:

- ➢ Communicate with precision.
- ➢ Divide up tasks between team members.
- ➢ Work together to unlock the combinations.

Lesson Sequence:

- ➔ Introduce the tasks and set up ground rules. (Will you allow them to look up answers on the internet? Are they allowed to refer back to their notes? What will the assigned roles be?)
- ➔ Establish how teams will record their answers. (Note - they should write them down as well as type them in, just in case the page freezes or crashes. They won't want to lose all their work.)
- ➔ Each team will have an internal clock, but set an overall timer for the class just in case.
- ➔ Answer Key - Lock combinations:
  - ◆ WORD LOCK = E-R-R-O-R (*Hint - use the binary chart from the "binary beads" activity)
  - ◆ DIRECTION LOCK =R-L-L-R-R
  - ◆ NUMBER LOCK = 6-4-5-9-0
  - ◆ SHAPE LOCK =STAR-CIRCLE-TRIANGLE-TRIANGLE-CIRCLE (*Hint - Start in the center square each time. Ignore ladybug with 0 dots. Start with 1 dot, then 2, etc.)
  - ◆ COLOR LOCK = RED-YELLOW-BLUE-GREEN-PURPLE (*Hint - start at the butterfly, and navigate the butterfly to the flowers.)