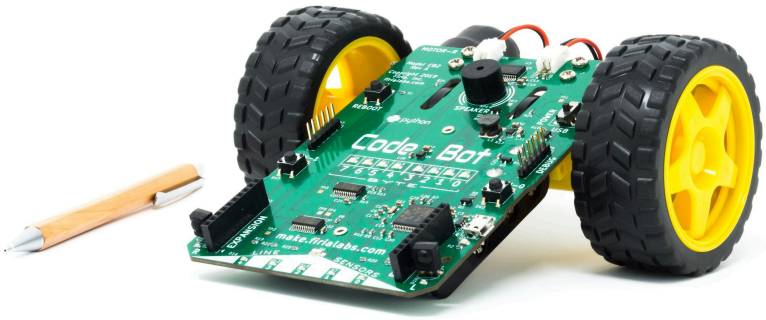
Python with Robots

Middle School Curriculum Guide



Unit 1

Introduction to Python with Robots

Unit 1: Introduction to Python with Robots (4-10 hours)

Students will learn about the programming environment, the CodeBot, and basic commands for programming the CodeBot using Python. Students create their own program to move the 'bot in a simple shape, like a square and use button presses for input.

Summary of Getting Started:

Students create an account and join the class to access the curriculum.

Summary of Mission 1:

Students become familiar with CodeSpace by learning its parts and features. It ends with an activity to label the parts of CodeSpace.

Summary of Mission 2: This mission is divided into two lessons, each approximately one class period.

Lesson 1: Students learn about CodeBot, its peripherals, and proper care.

Lesson 2: Students create a simple program by importing a library (module) and turning on an LED.

Summary of Mission 3: This mission is divided into four lessons, each approximately one class period. Students follow along with the slides and instructions in CodeSpace for the first lesson. The last lessons add to and change up some instructions in CodeSpace, and the slides can be used instead of the instructions in CodeSpace but still cover the same material.

Lesson 1: Students learn about the LEDs and turning them on and off. There are two sets of LEDs on the'bot that can be controlled. Students also learn about variables and the debugger.

Lesson 2: Binary is introduced and how to use it in code to turn on/off LEDS.

Lesson 3: Students learn how to turn on the motors and move the wheels both forwards and backwards and rotationally. Two robot labs are added in the assignment.

Lesson 4: Students learn about algorithms and putting together several steps to complete a task. They learn about push buttons and if statements to add decisions to code.

Unit 1 Remix: The remix project can be an extension of a current program or a completely new project. Students can work individually with a partner, or in teams.

Unit 1 Classroom Materials that are provided for each lesson:

- **Lesson Plan:** A detailed lesson plan is provided for each lesson. It includes learning targets, success criteria, vocabulary and new code. It also has teaching tips for each objective.
- **Mission Slidedeck:** Each lesson includes PowerPoint slides for teacher-led instructions. You can use them to guide students through the material. It is a supplement and alternative to reading a lot of instructions in CodeSpace. The slides mirror the instructions, with simplified language that is chunked into smaller sections at a time. The slides also provide added information, examples and instructions.
- **Mission Log:** Each lesson has an assignment, called a mission log, for students to complete as they go through the lesson. It includes warm-up and wrap-up questions as well as questions along the way for guided notes. An answer key for each mission log is provided.
- **Kahoot! Review:** Each lesson has a Kahoot! that reviews the concepts and codes.

Middle School: Python with Robots

Unit 1 Preparation:

- Create a class on the teacher dashboard.
- Have a computer / laptop with the Chrome web browser for each student.
- Make sure students can log in and create an account at http://make.firialabs.com, They will need an email address.
- Be able to give students the class join code.
- Have a CodeBot and USB cable for each student or programming pair.
- Have a ruler for measuring the distance traveled by the 'bot.
- Have a protractor for measuring angles (one is provided that can be printed).

Assessment:

Mission 1 Kahoot Review	Mission 2 Obj 1-5 Kahoot	Mission 2 Obj 6-10 Kahoot	Mission 3 Obj 1-5 Kahoot
Mission 3 Obj 6 Kahoot	Mission 3 Obj 7-8 Kahoot	Mission 3 Obj 9-11 Kahoot	

Coming soon: Unit 1 reviews and multiple choice exams are provided. They are divided into two parts: vocabulary and coding. The reviews are available as Kahoots, and the exams are provided as MS Forms.

All review and test questions are provided in the Unit 1 Question Bank (learning portal).

Unit 1 Vocab Kahoot Review	Unit 1 Coding Kahoot Review	Unit 1 Vocab Test (MS Form)	Unit 1 Coding Test (MS Form)

Standards addressed in this unit:

CSTA Standards Grades 6-8	CSTA Standards Grades 9-10	CSTA Standards Grades 11-12
• 2-CS-03 • 2-DA-07 • 2-AP-10 • 2-AP-11 • 2-AP-12 • 2-AP-13 • 2-AP-15 • 2-AP-16 • 2-AP-18 • 2-AP-19	• 3A-CS-03 • 3A-DA-09 • 3A-AP-13 • 3A-AP-16 • 3A-AP-19 • 3A-AP-21 • 3A-IC-26	• 3B-AP-10 • 3B-AP-16 • 3B-AP-17 • 3B-AP-22

Basic Lesson Plans for each lesson included here.

For complete lesson plans, see each mission in the Learning Portal.

Middle School: Python with Robots

MISSION 1: Welcome to CodeSpace	**Time Frame: 30-40 minutes**
Project Goal: Students will learn about the CodeSpace learning environment. **Learning Targets** ● I can navigate CodeSpace. ● Identify major parts of the Codespace interface: Mission Bar, Objective Panel, text editor, CodeTrek, Toolbox, and Lesson Navigation Controls	**Key Concepts** ● Follow instructions in the Lesson Panel carefully. There is a lot of important reading! ● Look for "tool icons" to collect tools in your Toolbox as you go. ● Some objectives include Hints. It is always a good idea to click on the hints.
Assessment Opportunities ● Quiz after Objective 4. ● Mission 1 Log (print or digital) ● Mission 1 Kahoot! Review	**Success Criteria** ☐ Navigate CodeSpace ☐ Identify major features of the CodeSpace interface: Editor panel, Lesson panel, Toolbox, CodeTrek, Hints
Teacher Materials in Learning Portal ● Mission 1 Slides ● Mission 1 Log (print or digital) ● Mission 1 Log Answer Key	**Additional Resources** ● Short CodeBot video (on Youtube and Firia website) ● Getting Started with CodeBot on youtube ● Mission 1 Kahoot! Review
Vocabulary ● **Objective:** The steps in the mission; has a goal to accomplish ● **Text editor:** Where you type the code ● **Code:** Instructions to the computer ● **Toolbox:** A place in CodeSpace to keep information you learn about programming concepts so you can use it later when you need the information ● **Simulation:** A 3D environment that lets you see the robot move and interact in a virtual world ● **Debugging**: Fixing your code	
New Python Code ● No code or programming in this mission	
Real World Applications Programmers need to use some type of text editor to create their code. CodeSpace is an IDE, or integrated development environment. It is patterned after other popular IDEs.	
Teacher Notes: ● This mission is also the first mission in the Python with CodeX curriculum. ● If students have visual problems, like color blindness, switching to the "light" preference can be helpful. ● The Mission Log is provided in a digital format or a print format. You can choose the one that is best for your students. ● The lesson is extended beyond the four objectives in CodeSpace by adding information on more of the parts of CodeSpace.	**Extensions / Cross-Curricular:** ● Supports **language arts** through reading instructions and reflection writing.

MISSION 2: Introducing CodeBot Lesson 1 (Objectives 1-5)	**Time Frame: 30-40 minutes**
Project Goal: Students will learn about the peripherals of CodeBot and identify them as input or output. **Learning Targets** • I can identify the main components of the CodeBot. • I can identify CodeBot inputs and outputs.	**Key Concepts** • There are a lot of hardware peripherals on the CodeBot, including sensors, LEDs, motors, buttons, and a speaker. • Some of the peripherals are used for input, and others are used for output.
Assessment Opportunities • Quiz after Objective 5. • Mission 2 Lesson 1 Log (digital) • Mission 2 Obj 1-5 Kahoot! Review	**Success Criteria** ☐ Identify the parts of the CodeBot ☐ Identify a peripheral as input or output
Teacher Materials in Learning Portal • Mission 2 Lesson 1 Slides • Mission 2 Lesson 1 Log • Mission 2 Lesson 1 Answer Key	**Additional Resources** • Mission 2 Obj 1-5 Kahoot! Review • Code.org video on how computers are changing everything

Vocabulary

- **CodeBot:** A computer on wheels with lots of sensors and controls built-in
- **Peripherals:** Devices that give input or output to the CodeBot; they include LED lights, speaker, motors, line sensors, proximity sensors, an accelerometer and push buttons
- **Motors:** Programmable electric engines; powers the wheels
- **LEDs:** Light emitting diodes; tiny and efficient electronic components that produce light
- **Wheel encoders:** Discs that rotate, counting the invisible IR light beam pulses through its slots
- **Static electricity:** A charge that can build up and causes a jolt and spark when grounded

New Python Code

- No code or programming in this lesson

Real World Applications

Make sure each student takes the time to personally inspect their ‘bot. Discuss the fact that all the electronic devices they use have similar circuit boards inside. The tools and techniques they’re learning apply to all the electronic devices they use every day! Challenge students to name a few devices they use every day that might contain computer chips or “microcontrollers” such as the one on the ‘bot. How many of the following do they think of? There are so many more!

Microwave oven	Cell phone	Automobile	Watch or fitness tracker
Video game controller	Refrigerator	Home thermostat	Coffee maker
Bread machine	Alarm system	Automatic garage doors	Electronic locks

Challenge students to describe how our lives are impacted by the above technology, and to compare how related tasks were done before computer technology was invented.

Teacher Notes:	**Extensions / Cross-Curricular:**
• Students do not need the CodeBot for this lesson, but it could be useful to look at during the lesson or the wrap-up. • The first pre-mission warm-up question asks	• Make a list of common input and output devices. • **LANGUAGE ARTS:** Students write about technology today and its impact. • **SCIENCE:** Students research a microcontroller or

students about peripherals. You might want to review what a peripheral is and have students become familiar with the term. • The mission log assignment is created as a digital document. You can print it for students as an alternative by giving more space for answering questions by hand.	another every day technology device. • Supports **language arts** through reading instructions and reflection writing.

MISSION 2: Introducing CodeBot
Lesson 2 Your First Program (Obj. 6-10)

Time Frame: 30-40 minutes

Project Goal: Students will connect CodeBot to the computer and learn the basics of Python.

Learning Targets

- I can safely connect and disconnect the CodeBot using the USB cable.
- I can create a new file.
- I can write code using the conventions of comments and correct punctuation.

Key Concepts

- Python requires all objects – variables, peripherals, etc. – to be spelled exactly the same; capitalization matters!
- Adding comments and blank lines in your code makes it easier to read.
- Python programmers can import a library, or module, to access pre-defined functions.

Assessment Opportunities

- Mission 2 Lesson 2 Log assignment
- Submit completed program ***LightsOn***
- Submit the program with challenges
- Mission 2 Obj 6-10 Review Kahoot!

Success Criteria

- ☐ Create a new file
- ☐ Import the botcore library
- ☐ Use code to turn on an LED
- ☐ Use descriptive comments

Teacher Materials in Learning Portal

- Mission 2 Lesson 2 Slides
- Mission 2 Lesson 2 Log
- Mission 2 Lesson 2 Answer Key

Additional Resources

- Mission 2 Obj 6-10 Review Kahoot!
- LightsOn_challenge sample code (in learning portal)

Vocabulary

- **Comment:** Notes in a program code that don't get executed (more information in Mission 3)
- **Import:** Provides access to a module, or library, of pre-defined Python objects and functions to use in your code
- **Boolean:** Something that has two possible values: True or False

New Python Code

`From botcore import leds`	Import from botcore only the leds object and its functions
`leds.user_num(0, True)` `leds.user_num(7, False)`	Turn on one user LED (arguments are LED number 0-7, True=on, False=off) Turn off user LED #7
`leds.ls_num(0, True)`	Turn on a line sensor LED (arguments are LED number 0-4, True/False)

Real World Applications

- Many electrical devices have single-color LEDs. Discuss devices that have them and what they are used for.

Teacher Notes:

- Students will need a CodeBot and USB cable for this lesson.
- This is a good time to discuss common programming practices, such as descriptive variable names, adding comments and blank lines, use of capital letters, etc.
- Students will type their first lines of code. You may want to create a poster of Python commands to refer back to during the Mission Pack. Or you can encourage students to create their own "library" of Python codes.

Extensions / Cross-Curricular:

- Make a poster or chart of Python commands.
- Turn on the line sensor LEDs.
- Combine user and line sensor LEDs to create a pattern, like turning on the even LEDs and leaving the odd LEDs off.
- **LANGUAGE ARTS:** Students write about technology today and its impact.
- **SCIENCE:** Have a lesson on LEDs and light.

MISSION 3: Time and Motion
Lesson 1 (Objectives 1-5)

Time Frame: 30-40 minutes

Project Goal: Students will control CodeBot LEDs with specific timing and sequencing.

Learning Targets
- I can import a specific built-in function from a library.
- I can use the "Step" feature of the CodeSpace debugger.
- I can use a variable to make code more efficient.
- I can assign a value to a variable.
- I can use a variable as an argument in a function call.

Key Concepts
- Computers execute code in sequential steps.
- The CodeSpace debugger lets you *step* through the code one line at a time to understand what the computer is doing.
- Built-in functions come from libraries, like botcore or time.
- A function call can pass a value, or argument, to the function.
- Variables can be defined to hold changing values. Variables make code more efficient and reduce duplication.

Assessment Opportunities
- Mission 3 Lesson 1 Log (digital)
- Submit completed program ***SequenceLEDs***
- Submit the program with extensions
- Mission 3 Obj. 1-5 Review Kahoot!

Success Criteria
- ☐ Light up four user LEDs in sequence
- ☐ Use the "Step In" feature of the CodeSpace debugger
- ☐ Import the sleep function
- ☐ Use the sleep function to slow down code
- ☐ Define a variable
- ☐ Use a variable in a function call
- ☐ Create a light show with CodeBot's LEDs

Teacher Materials in Learning Portal
- Mission 3 Lesson 1 Slides
- Mission 3 Lesson 1 Log
- Mission 3 Lesson 1 Answer Key

Additional Resources
- Mission 3 Obj. 1-5 Review Kahoot!
- SequenceLEDs sample code (learning portal)
- SequenceLEDs_extensions sample code (learning portal)

Vocabulary
- **Physical computing:** Writing code (instructions) for a physical device, like CodeBot or cars
- **Editor shortcuts:** Keyboard hotkeys to write code faster; combinations of keys which complete a task
- **CPU:** The "brain" of the computer that executes your code; the Central Processing Unit
- **Debugging:** The process of understanding what the computer is actually doing and then changing the code to do what you want it to do
- **Argument:** A value that is passed to a function
- **Literal:** An actual value, like 1 or "hello" or True
- **Variable:** A name to which you assign some data, like a number; must be defined before it is used

New Python Code

`from time import sleep`	Import the ability to delay, or pause, the code
`sleep(1.0)`	Use sleep() – will sleep the amount of time in seconds
`delay = 1.0`	Define a variable (define variables near the top just under the imports)
`sleep(delay)`	Use a variable with sleep()
`leds.user_num(2, False)`	Turn off an LED

Middle School: Python with Robots

Real World Applications

You've used some fundamental computer science and robotics principles:

- Controlling LEDs with specific timing and sequencing

This code is used in cars, stage lights, espresso machines, music sequencers, and more!

Teacher Notes:

- Although you cannot copy and paste from CodeTrek, you can copy and paste in the text editor. The program for this lesson has a lot of repetition. Show students editor shortcuts, as needed, to save time.
- This lesson is a good time to discuss common programming practices, such as descriptive variable names and blank lines.
- The extensions for this lesson give students an opportunity to be creative and express themselves through a light show. If you have time for students to do an extension, it is motivational and engaging.

Extensions / Cross-Curricular:

- Make a poster or chart of Python commands.
- Use the line sensor LEDs in the light show.
- Combine user and line sensor LEDs to create a pattern, like turning on the even LEDs and leaving the odd LEDs off.
- **MATH:** Students design a timed light show and use math to determine the specific timings for each LED segment to meet the specified time.
- Supports **language arts** through reading instructions and reflection writing.

MISSION 3: Bright Byte Lights **Lesson 2 (Objective 6)**	**Time Frame: 40-60 minutes**
Project Goal: Students will convert between decimal and binary numbers and use them to control LEDs. **Learning Targets** • I can define “bit” and “byte”. • I can convert a decimal number to binary. • I can convert a binary number to decimal. • I can use a binary number to turn on/off user LEDs. • I can use a binary number to turn on/off line sensor LEDs.	**Key Concepts** • Computers use binary numbers to represent all information. • You can control (turn on and off) LEDs using binary numbers. This is much faster, and less lines of code, than turning on and off each LED individually. • Binary numbers have only two digits: 0 and 1, and it uses the base 2 number system.
Assessment Opportunities • Mission 3 Lesson 2 Log • Submit completed program ***Binary LEDs*** • Submit the program with extensions • Mission 3 Obj. 6 Review Kahoot!	**Success Criteria** ☐ Convert a number between 0 and 15 to binary ☐ Convert a binary number to decimal ☐ Convert a number between 0 and 255 to binary ☐ Write code using binary to turn on/off user LEDs ☐ Write code using binary to turn on/of ls LEDs
Teacher Materials in Learning Portal • Mission 3 Lesson 2 Slides • Mission 3 Lesson 2 Log • Mission 3 Lesson 2 Answer Key	**Additional Resources** • Mission 3 Obj. 6 Review Kahoot! • Secret Coders video (introduces binary numbers and Activity #1) • BinaryLEDs sample code (learning portal)

Vocabulary

- **Binary:** A number system, or computer language, that uses only 0s and 1s
- **Bit:** A single binary digit (on/off or 1/0)
- **Byte:** A set of 8 bits of binary data
- **Bit banging:** Controlling hardware with binary digits

New Python Code

`leds.user(16)`	Use a decimal number to turn on/off all user LEDs
`leds.user(0b10101010)`	Use a binary number to turn on/off all user LEDs (0b for binary, then 0=off, 1=on for each LED)
`leds.ls(0b11111)`	Use a binary number to turn on all line sensor LEDs
`leds.ls(0b00100)`	Use a binary number to turn on only the middle line sensor LED

Real World Applications

Computers combine binary digits (bits) into binary codes that enable them to represent any type of data, from numbers and text to images and sounds.

- Text: a unique numerical code is assigned to each character, which is stored and processed as binary sequences.
- Images: An image is represented using a grid of pixels, with each pixel assigned a number for its color.
- Sound: Computers convert continuous analog sound waves into a series of digital values using binary code.

Teacher Notes:	**Extensions / Cross-Curricular:**
• This lesson uses Mission 3 Objective 6 at the beginning and end, but adds three activities to help students understand binary numbers. The	• Make a poster or chart of Python commands. • Use the line sensor LEDs and user LEDs in a light show.

best way to present this lesson is to start with Obj. 6 for the intro, and then use the slides all the way until the end. They can use CodeSpace instructions and CodeTrek if needed for the program. • If you don't have much time, or you don't want the extra activities on binary, you can cut them out of the lesson and just do Objective 6. If so, use the Mission 3 Obj. 1-6 Kahoot review instead of the one in the additional resources.	• Do an unplugged activity with binary, like binary bracelets. • **MATH:** Binary uses base 2. Discuss number systems with other bases, like the octal system with base 8 or hexadecimal with base 16. Other bases are fun, too, like base 3. • Supports **language arts** through reading instructions and reflection writing.

MISSION 3: Get Moving! **Lesson 3 (Objectives 7-8)**	**Time Frame: 40-60 minutes**
Project Goal: Students will move the CodeBot by powering the wheels. **Learning Targets** • I can import everything in the botcore library. • I can enable the wheels. • I can move CodeBot forward. • I can rotate CodeBot clockwise and counterclockwise.	**Key Concepts** • The botcore library contains many functions and objects, not just leds. You can import the entire library using the wildcard *. • The wheels must be enabled before running. If you forget this line of code, you will not get an error, the wheels just won't turn. • A positive left wheel and negative right wheel will rotate clockwise. A negative left wheel and positive right wheel will rotate counterclockwise.
Assessment Opportunities • Mission 3 Lesson 3 Log • Submit completed program ***MoveOut*** • Mission 3 Obj. 7-8 Review Kahoot!	**Success Criteria** ☐ Import the entire botcore library ☐ Move the CodeBot forward ☐ Rotate the CodeBot clockwise ☐ Rotate the CodeBot counterclockwise
Teacher Materials in Learning Portal • Mission 3 Lesson 3 Slides • Mission 3 Lesson 3 Log • Mission 3 Lesson 3 Answer Key	**Additional Resources** • Paper protractors (learning portal) • Mission 3 Obj. 7-8 Review Kahoot! • MoveOut sample code (learning portal)

Vocabulary

- **Wildcard:** The * character; shorthand for "everything."

New Python Code

`from botcore import *`	Import an entire library (* is a wildcard, which means everything)
`motors.enable(True)`	Turn on motors, must be done before motors will turn and wheels move
`motors.run(LEFT, 50)`	Turn left wheel forward at 50% power (use -50 for backward movement)
`motors.enable(False)`	Turn off motors
`motors.run(LEFT, 30)` `motors.run(RIGHT, 30)`	Move forward (straight line)
`motors.run(LEFT, 30)` `motors.run(RIGHT, -30)`	Rotate (+ left, - right is clockwise; -left, +right is counterclockwise)

Real World Applications

You've used some fundamental computer science and robotics principles:

- Controlling motors with specific timing and sequencing

This code is used in cars, robots, electric toothbrushes, and more!

Teacher Notes:	**Extensions / Cross-Curricular:**
• All CodeBots will need batteries for this lesson. The 'bots will be moving around the room. Rechargeable batteries work fine. • CodeBots will need space to move. Dedicate some	• Make a poster or chart of Python commands. • **SCIENCE:** Use the labs as science experiments. Talk about cause and effect. You can add an extra column to the lab reports and have students

floor space in your room for students to test their code. • This lesson covers objectives 7 and 8 in Mission 3. You do not need to use the instructions in CodeSpace if you don't want to. All material is covered in the slides, including CodeTrek and meeting the goals. • This lesson includes two "robot labs" that aren't in the CodeSpace instructions. They give students a chance to try things out and experiment. ***The labs prepare students for the next lesson***, which is a navigation challenge.	predict what will happen before running the code. • **MATH:** Make a chart or graph from the data of the lab reports. • Supports **language arts** through reading instructions and reflection writing.

MISSION 3: Navigation Challenge **Lesson 4 (Objectives 9-11)**	**Time Frame: 40-60 minutes**
Project Goal: Students will use button presses to control robot movement and follow an algorithm to move the 'bot in a square pattern. **Learning Targets** • I can check for a button press. • I can use an if statement for branching. • I can indent code inside a control flow. • I can implement a safety feature in code when the CodeBot moves. • I can write and follow an algorithm. • I can increase code readability by using comments and blank lines. • I can make the 'bot move in a square pattern. • I can understand the control flow of an if statement.	**Key Concepts** • The push buttons can be used to control program flow. One way to do this is by enacting a safety feature, so code is only run after a button is pressed. This will keep the 'bot from moving right when the program is run. • Branching with if:elif:else statements controls the flow of the program. • The colon (:) at the end of an if statement introduces a new block of code. Everything inside the block should be indented at the same level. • An algorithm is a useful tool for planning a program. There are many ways to create an algorithm, such as pseudocode and flowcharts. In this lesson, only pseudocode is used.
Assessment Opportunities • Mission 3 Lesson 4 Log • Submit completed program ***NavSquare*** • Submit completed program ***WhatIf*** • Mission 3 Obj. 9-11 Review Kahoot!	**Success Criteria** ☐ Use buttons.was_pressed() in an if statement ☐ Implement a safety feature in code ☐ Plan a program using an algorithm ☐ Write code to drive CodeBot in a specific pattern ☐ Use buttons.was_pressed() in an if/elif/else statement
Teacher Materials in Learning Portal • Mission 3 Lesson 4 Slides • Mission 3 Lesson 4 Log • Mission 3 Lesson 4 Answer Key	**Additional Resources** • MoveOut_safety sample code (learning portal) • NavSquare sample code (learning portal) • WhatIf sample code (learning portal) • Mission 3 Obj. 9-11 Review Kahoot!

Vocabulary

- **Algorithm:** A list of instructions, in order, that the computer can follow to complete a task. (A precise sequence of instructions that the computer can follow exactly, one step at a time, to complete a task or solve a problem)
- **Comments:** Notes in the code about what you are doing; ignored by the computer
- **Whitespace:** Adding blank lines and space around symbols to make the code more readable
- **Control flow (branching):** Decision points in code; code takes a different branch depending on a condition
- **Condition:** A Boolean value (True or False), often the result of a comparison operator like <, >, or ==. Use an if statement, optionally followed by an elif or else, for branching
- **Indenting:** A way to structure blocks of code by offsetting a block of code four spaces; blocks of code are indented following a defining statement with a colon (:)

New Python Code

`buttons.was_pressed(0)`	Checks to see if BTN-0 was pressed; returns True or False

`leds.user(0b00011000)`
`sleep(10)`
`leds.user(0)`
`if buttons.was_pressed(0):`
`  {indented code}`
	Safety feature in code that only executes indented code if Button-0 was pressed.

`if buttons.was_pressed(0):`
`  # turn right`
`elif buttons.was_pressed(1):`
`  # turn left`
`else:`
`  # stop`
	Example of control flow, or branching, using if/elif/else.

Real World Applications

You've used some fundamental computer science and robotics principles: controlling motors with specific timing and sequencing, incorporating safety features, and programming different options, depending on conditions. This code is used in cars, robots, electric toothbrushes, and more!

Algorithms are part of our lives. We use algorithms all the time, for many of the tasks we complete each day. Have students think of a daily task, like getting ready for school or making a jump shot in basketball. Then have them break down the task into an algorithm.

Teacher Notes:

- This lesson covers objectives 9-11 in Mission 3. You do not need to use the instructions in CodeSpace. Some of the concepts are switched around and introduced in a different order, but all material is covered, and all goals for each objective will be met You should choose to use either the CodeSpace instructions or the slides, but not both because even the code used in the slides is a little different than CodeTrek.
- Algorithms are used throughout this lesson. There are many unplugged activities for practicing algorithms, and they can be found in the learning portal under "CS Unplugged."

Extensions / Cross-Curricular:

- Suggested unplugged algorithm activities you can choose from:
  Algorithms lesson with drawings
  Algorithms lesson with LEGO
  PB&J Algorithm
  Hide and Seek
- **MATH:** Have a discussion about the algorithms used in math, and have students write algorithms for common math problems.
- Supports **language arts** through reading instructions and reflection writing.

Unit 1 Remix Project	**Time Frame: 2-5 hours**
Project Goal: Students will use the skills and concepts they learned in the unit to create their own project. **Learning Targets** • I can summarize the programming concepts from Missions 2 & 3. • I can plan an original program. • I can create an original program using concepts and code from previous programs. • I can get feedback on my project.	**Key Concepts** • Code segments from previous programs can be reused and repurposed in a new project. • The program development in the planning guide follows the software design process. • Creating a new project from the beginning, without CodeTrek or starter code, is an excellent way for students to master their learning and gives them an opportunity to express themselves and work on something that interests them.
Assessment Opportunities • Unit 1 Remix Planning Guide • Unit 1 Remix Project	**Success Criteria** ☐ Plan an original program ☐ Create an original program ☐ Incorporate feedback in a program
Teacher Materials in Learning Portal • Unit 1 Remix Project slides • Unit 1 Remix Planning Guide	**Additional Resources** • Students can use their previous programs as a guide throughout this project.

Teacher Notes:

- A remix for this unit is optional. Unit 1 is a very long unit, with several lessons. If you feel like students have had enough practice through the lessons, you can skip the remix. However, it is an excellent opportunity for students to create their own original program by doing something that interests them. And the remix project gives students a chance to practice and apply what they have learned.
- Students can work with a partner for this project. Collaboration is an important skill. Students will be programming both CodeBot buttons. As one option, if working with a partner, each student could write code for one of the buttons, and they can combine the code into a finished project.
- A set of slides is prepared to explain the project and give step by step guidance. The slides also give some suggestions for the project. The suggestions are meant to help students think of their own ideas and should not be required. They can be used for students who are drawing a complete blank, or as inspiration.
- A planning guide is provided to help students know where to start, and to guide them throughout the process. You can modify the planning guide as needed by changing or adding to the questions.
- Consider how you want to end the remix project. You can have students present them to the class, have a “gallery walk” of projects, have students create a slide show about the project, etc.
- A checklist for the remix project is below.

Remix Project Checklist:

- ☐ Filename is descriptive
- ☐ Uses one or more variables, each with a descriptive name
- ☐ Moves the CodeBot forward and/or backward at least once
- ☐ Rotates the CodeBot at least once
- ☐ Uses a sleep delay at least once
- ☐ Turns on at least one LED light
- ☐ Uses two buttons as input
- ☐ Modify program based on user feedback
- ☐ Includes comments and blank lines for readability
- ☐ Code runs without errors