



# CODEMONKEY

## CURRICULUM GUIDE

Trial version





Copyright © 2017 by CodeMonkey Studios

All rights reserved. This book or any portion thereof  
may not be reproduced or used in any manner whatsoever  
without the express written permission of the publisher.

195 Montague, 14th floor

Brooklyn, NY 11201

[info@cm-studios.com](mailto:info@cm-studios.com)

[www.playcodemonkey.com](http://www.playcodemonkey.com)

## Table of contents

Introduction	3
Group Management	6
Lesson 1 - Let's Get Started	9
Lesson 2 - Turn Around	13
Lesson 3 - I Have a Plan!	17
Lesson 4 - Turtle Lake	21
Lesson 5 - In the Loop	25
Lesson 6 - Loop On	29
Lesson 7 - 3 Stars Party	33
Reference Card	35
Characters Review	37
Support	38

# Introduction

Dear teacher,

Thank you for choosing to teach coding with CodeMonkey. CodeMonkey allows you, as an educator, to enrich your students' lives. Through fun challenges with beautiful animations and lovable characters, alongside our unique user experience and special online teaching tools, you can introduce basic computer science knowledge to your students and better prepare them for the 21<sup>st</sup> century. We have created this Curriculum Guide to help you teach coding to your students - no matter how much experience or previous knowledge you have in computer science.

This curriculum corresponds to the basics of CodeMonkey. In the first 30 challenges of CodeMonkey we introduce the following subjects: objects, functions, statements, arguments and the simple loop. In our full course, CodeMonkey has 400 challenges available, covering more advanced topics such as for loops, until loops, variables, conditionals and more. If you and your students enjoy our basic course, we encourage you to consider teaching the whole course. For more details please visit [playcodemonkey.com](http://playcodemonkey.com).

In this Curriculum Guide, you will find:

- A short introduction to get you started
- 7 lesson plans - each one designed to run 45 minutes
- A CodeMonkey Reference Card
- Support information

Please take note of the following guidelines for using CodeMonkey in class:

## CoffeeScript

- The programming language used in CodeMonkey is called CoffeeScript. It's a language that compiles to JavaScript, and similarly to JavaScript it is used in the industry primarily for web applications.
- We chose this language for a few reasons, but mainly because of its friendly syntax, which resembles the way we write in English, compared to other programming languages.
- You can read more about CoffeeScript [here](#).

## Prior to Class

- Complete the first few challenges or even all 30 challenges ahead of class so you can help students during class time.
- CodeMonkey runs with HTML5 so it should run on your students' computers with no problems.
- Sound isn't necessary to succeed in CodeMonkey, but it does make the challenges more fun.
- If you don't have enough computers for all students, students can work in pairs. Make sure they discuss their solutions and work collaboratively. Working in pairs can

sometimes be more productive because discussing the process with someone helps us learn better.

- Before each class read through the day's lesson plan and make sure you understand the topic.
- Refer to these introductions as needed.
- If your school is not using one of the supported login methods (Google, Office 365, etc.), create CodeMonkey accounts for your students using this setup guide. This guide includes instructions student registration and managing groups. Make sure to keep a list of student usernames and passwords for future reference.

### **In Class**

- During the first class, you should show the students how to log in to their CodeMonkey accounts.
- Follow the day's lesson plan, but feel free to deviate from it. We encourage you to ask more questions and let your students express themselves.
- While students are working on the activity, walk around the classroom and answer any questions that they might have.
- Give extra attention to students who are far behind their classmates, or student pairs that are not working cooperatively.
- Point your students' attention to our star rating and the meaning of it.
- Once in a while, you can ask for your students' attention and discuss different options for solving challenges.
- In the teacher dashboard, you can see the actual code your students wrote. Use this feature to inspire a discussion in class, but be sure students don't know whose code you are showing.
- Some of the challenges are assessment challenges. This means that the students have to demonstrate what they have learned by writing the entire code from scratch.

### **When Your Students Have Difficulties**

- Most of the issues your students will encounter will result from not reading the instructions or the code itself properly. Encourage them to read the instructions/code carefully. They can see the instructions again by clicking on Gordo (lower-left corner).
- Some of the challenges are "debugging" challenges, where there is a mistake in the code and the students need to fix it. Encourage your students to start each challenge by pressing "run" and watching what the initial code does. This will help clarify where the problem is.
- Encourage students and offer positive reinforcement: "You're doing great, so keep trying."
- It's okay to respond, "I don't know the answer. Let's figure this out together." If you can't figure out a problem, use it as a good learning lesson for the class: "Technology doesn't always work out the way we want. Together, we're a community of learners."
- At any time, you can use the reference solutions in your teacher dashboard to see three-star solutions to all challenges.

### **What Should I do if a Student Finishes Early?**

- Encourage the student to go back and try to get three stars in all the challenges.

- Ask students who finish early to help their classmates who are having trouble with the activity.

**After Class**

Use the teacher dashboard to:

- Monitor your students' progress
- See your students' actual code
- See the scores students received in every challenge
- Determine if there are any topics that need to be repeated
- See statistics on your class's achievements
- Look up a three-star solution for any challenge

Lastly, remember to have fun! It's important to us that our users know that coding can be fun and that it doesn't have to be boring or frightening.

At any time, should you have any questions, you can contact us at: [info@cm-studios.com](mailto:info@cm-studios.com).

Good luck!

The CodeMonkey team

# Classroom Management

This guide will help you get started with creating accounts for your students and managing your classroom.

If you need help or have questions, please contact us at [info@cm-studios.com](mailto:info@cm-studios.com).

[Click here](#) to view extended information about classroom management.

## General information

### 1. Access your classroom

To access your classroom click the menu icon in the top right corner, then select Classrooms from the menu. A list of your classrooms will appear with some statistics about student progress and activity. Click on a classroom to see more information about students.

### 2. Change your classroom's name

From the classrooms page click the classroom you want to edit. When inside the classroom, click on the classroom's name. A text field will appear where you can edit your classroom's name, click on save when you are done.

### 3. Set the language for you classroom

From the classrooms page click on the "Set language" button in the green bar. A pop-up will appear, choose the flag representing your language. Keep in mind that this will set the default language for all the students in your classrooms. You can always access the languages menu by clicking on the menu icon in the upper right corner and then on the language flag.

### 4. Create student accounts

Your dashboard is constructed of 3 tabs. The first tab is the "students" tab in which you can create and manage your students' accounts. Click on the "Add students" button under your student list to create student accounts.

There are 3 ways to create accounts for students. Below you will find explanations on how to use each of these methods. Please note that you can pick either one of the three methods, but you can always change between these methods later.

## Single Student Registration

To create a single user for a student, follow these steps: (for multiple user upload, see next section)

1. Click the menu icon in the top- right corner and select **Classrooms** from the menu.
2. Click the classroom to which you want to upload the student
3. In the "Students" tab click **Add students**
4. Click **"Create single account"**
5. Fill in the username (we recommend using the **first initial.last name** convention) and the Password. Passwords need to be at least six characters long.
6. Click on **Create**. The user will be created and added to the classroom.



## Multiple Student Registration (bulk upload)

For registering your whole class, you will need to create a CSV file that includes all of your students and upload the file directly into CodeMonkey.

1. Using Excel or Google Spreadsheets, create a CSV file with two columns listing all of the students you wish to add:
  - a. Column 1 will contain student usernames using the **first initial.last name** convention.
  - b. Column 2 will contain student passwords. Passwords need to be at least six characters.
  - c. Do not include a header row in the CSV file.
  - d. Make sure you save the file in CSV format
    - i. Excel users: **File, Save As** then **File Type** to select **Comma Separated Values**.
    - ii. Google Spreadsheets users: use **File, Download As** then select **Comma Separated Values**.
2. Go to [www.playcodemonkey.com](http://www.playcodemonkey.com), and click the menu icon in the top- right corner.
3. Select **Classrooms** from the menu. A list of your groups will appear.
4. Click the group to which you want to upload the students.
5. In the “Students” tab click **Add students**
6. Under “Bulk upload” click on **Choose file**.
7. Select the CSV file to upload, and click **Open**.
8. Click **Upload**. The users will be created and added to the classroom.

## Share class code

1. Students can sign up to CodeMonkey on their own and join your classroom by entering your classroom code. The classroom code is a combination of letters and numbers and is unique to your CodeMonkey classroom.
2. Your classroom code appears in a few places:
3. In the classrooms page underneath the name of the classroom
4. In the “students” tab in your classroom page, in the upper right corner.
5. In the “Add students” window -
  1. Click the menu icon in the top- right corner and select classrooms from the menu.
  2. Click the classroom to which you want to upload the student.
  3. From the “Students” tab click on Add students.
  4. Click “Share class code”.
  5. Your unique code will be visible in the box with the green outline. Write the code on the whiteboard or print a sheet to handout to the students.
  6. If you ever want to generate a new code, click on the re-generate icon (two arrows) to the right of the box with your classroom code.

## Dashboard and Challenge Solutions

Your teacher dashboard is where you can manage student’s accounts, see their progress, the code they wrote on each challenge, solutions to all challenges and student’s grades.



Your dashboard is constructed of 3 tabs:

1. Students - for student management
2. Progress - Where you can see information relevant to their progress in the game
3. Grades - Where you can see their grades based on their progress (this feature is only available for teachers with a classroom subscription).

After creating accounts for your students in the “students” tab, go over to the “progress” tab to see the progress table. Once your students start to play, colorful stars will appear to represent their solutions and the star score they got.

## How to find solutions to challenges

To see solutions to different challenges:

1. Click the menu icon in the top right corner.
2. Select **Classrooms** from the menu. A list of your groups will appear, choose your group.
3. In your dashboard, go to the “progress” tab
4. Using the left and right arrows, locate the challenge number you want a solution for.
5. Click on the number.
6. You will now see a **reference solution** (3 stars) to this challenge.
7. To go back to your dashboard, click on the “back” arrow in your browser.

If you want to see a **reference solution** to a skill challenge, first click on the “skill mode” tab, then repeat items 3-5.

## How to see a student solution

To see a **student’s solution** to a challenge, follow these steps:

1. Go to your classroom dashboard
2. Click on the “progress tab”
3. Locate a student’s username in the table.
4. Click on the star that represents the solution to the challenge you want to see
5. Once you click on the star, a pop up window will appear with a time stamp. If the student solved the challenge more than once, a few time stamps will appear, each with it’s own star. Pick which solution you want to see and click on the time stamp.
6. You will now see the student’s solution to this challenge.

If you want to see a **student’s solution** to a Skill Mode challenge, first click on the “Skill Mode” tab, then repeat items 3-6.

**Didn’t find what you were looking for?** [Click here](#) to view extended information about classroom management.

## Lesson 1 - Let's Get Started

### Challenges 0-5

This lesson introduces students to the fascinating world of computers and to the CodeMonkey platform. Some of your students may be familiar with the terms “coding” and “programming” and feel comfortable working with computers. For others, learning to code may be an intimidating experience. As educators, our goal is to help students learn and explore a variety of subjects, including computer science. We want to create an environment where students can learn from their mistakes and build their foundational knowledge to create something new.

### Objectives:

Within this lesson, students will:

- Define *coding* and *computer programming*
- Become familiar with the CodeMonkey platform
- Complete challenges 0-5 on CodeMonkey

### Components:

- Instructions: “step”, “turn”
- Terms: challenge, CoffeeScript

### US Standards Addressed:

- CCSS-Math Standards: PRACTICE.MP1, PRACTICE.MP3, PRACTICE.MP4, PRACTICE.MP5, CONTENT.4.MD.C.5
- CCSS-ELA Standards: Literacy.RST.6-8.3, Literacy.RST.6-8.4

### Part 1: Introduction - 10 minutes

<b>Discussion</b>	<b>2 min.</b>
<ol style="list-style-type: none"><li>1. How many of you have ever used a computer?</li><li>2. Have you ever created something on a computer, like a presentation, a drawing, or maybe even a game?</li><li>3. Let two or three students tell the class what they created.</li></ol>	
<b>Explain</b>	<b>1 min.</b>
In order for all of our favorite applications and games to work on a computer, we have to give instructions to the computer. Computers can't think for themselves, they do whatever we tell them to do. Giving instructions to the computer is called computer programming or coding.	

<b>Activity</b>	<b>3 min.</b>
<p>Play a short game with your students to illustrate instructions. Place an object somewhere visible in the classroom. Ask your students to give you instructions to guide you from where you are standing in class to the object.</p> <ul style="list-style-type: none"> <li>What instructions did the students use, e.g., step, turn right, turn left?</li> </ul>	
<b>Discussion</b>	<b>2 min.</b>
<p>Do computers speak the same language as humans?</p> <p>Computers have their own languages; they can't understand human language as we understand it. HTML, JavaScript, and Python are just a few of the languages computers speak. Each language is different, but they all have something in common: they require a certain way of thinking, clear instructions, and structure. Basically, learning a coding language is just like learning a new language.</p>	
<b>Watch</b>	<b>2 min.</b>
<p>Today you will start learning basic coding principles through a game called CodeMonkey. The language we will learn is called <b>CoffeeScript</b>.</p> <p>Show the <a href="#">CodeMonkey Trailer</a> to the students.</p>	

## Part 2: Let's Go! - 25 minutes

<b>Explain</b>	<b>3 min.</b>
<p>Go to the CodeMonkey website: <a href="http://playcodemonkey.com">playcodemonkey.com</a></p> <p>Instruct the students how to log in to their CodeMonkey accounts.</p> <p>If the students use usernames and passwords to login, make sure they store their usernames and passwords where they can easily access them in the future. Optional: hand out user/password cards.</p> <p>At any time, should a student forget their password, you can reset it by visiting the teacher dashboard, locating the student's username, and clicking on the edit button which will appear if you hover over the username.</p>	
<b>Walkthrough (1)</b>	<b>4 min.</b>
<p>Walk your students through the basic appearance of CodeMonkey:</p> <ol style="list-style-type: none"> <li>Click on the "play now" button on the homepage</li> <li>Watch the short introduction animation</li> <li>Read the instructions out loud</li> <li>The CodeMonkey game is built of levels called challenges. This is what a challenge looks like.</li> <li>The editor on the right is where you'll write your code. You can also use the buttons at the bottom for easy access.</li> <li>On the left is the stage, this is where you'll see your code come to life. Your goal is to complete every challenge by helping the monkey catch the banana</li> </ol>	

7. The monkey on the lower left corner is Gordo, he will give you instructions and sometimes even hints if you're stuck. At any time, you can click on Gordo to see the instructions again.

### Walkthrough (2)

3 min.

8. In every challenge, you will execute the code by clicking on the "run" button to see what the starting code will do.
9. The code on the right says "step 15", so when we'll click on "run" the monkey will step 15 steps forward.
10. Click on run.
11. We completed the first challenge. After every completed challenge, you'll get a star score rating your solution. 3 stars is the highest score and is rewarded for catching all the bananas, implementing new learned topics, and writing short code. If you get less than 3 stars, a hint will help you get them all. You can try to solve a challenge as many times as you want, it will not affect your stars score!
12. Click on replay to see your solution again.
13. Edit the solution to change it from  

```
step 15
to
step 5
step 10
```
14. Click "run" again to execute your solution again. Show the students that this solution only got 2 stars, and draw their attention to the hint that tells them how to get the 3rd star.
15. Click replay again, fix the solution to get 3 stars, and execute it again by clicking "run".

### Walkthrough (3)

3 min.

16. Let's move on to the next challenge, click on "next challenge".
17. Read the instructions out loud
18. The code on the right says "step 10", let's click on run and see what happens.
19. The monkey didn't walk far enough, and the hint told us to try "step 15", let's change the number 10 to 15, and click "run" again.
20. It's always a good idea to use the code that was there. Before we try to change the code, click "run" to see what happens, read the hint, and then try to solve. It doesn't matter that you ran code that doesn't solve the challenge. Besides, the code is there for a reason, so don't delete it.
21. Another good strategy for when you're stuck is to start again from the beginning, in cases like these you can reset your code by clicking on the reset button.
22. Click the replay button, and then click the reset button to show your students how to reset the code to what it was in the beginning.
23. Solve again by editing the code and click "run" to execute the solution.
24. Show your users how to go back to challenge 0 by clicking on the map (top right corner) and clicking on challenge 0. Note that unlike you as a teacher, your students will not be able to skip forward beyond the first challenge that they have not solved yet.

<b>Play time</b>	<b>10 min.</b>
<p>All students should complete challenges 0-5 with at least two stars. (Students from the age of 12 and up should get three stars.) Use the teacher dashboard to keep track of students' achievements.</p> <p>If students are having trouble confusing right and left, draw their attention to the watch on the monkey's left wrist. Tell them that turning in the direction is left.</p>	
<b>Review</b>	<b>2 min.</b>
<p>Open challenge #2 and show the ruler animation. Follow the instructions to measure the distance between the monkey and the banana, and then use that distance to fix the code. Make sure your students understand how to use the ruler.</p>	

### Part 3: Debriefing - 10 minutes

<b>Discussion</b>	<b>5 min.</b>
<ul style="list-style-type: none"> <li>• What instructions did you learn today?</li> <li>• What did you like most about CodeMonkey?</li> <li>• Besides instructions, what else did you learn today?</li> <li>• How do you get 3 stars in a CodeMonkey challenge? Does it matter how many times I tried to solve the challenge? (No, it doesn't!)</li> <li>• What do you do when you are stuck? (the following two questions are related)</li> <li>• In a CodeMonkey challenge, how do you display the instructions again?</li> <li>• In a CodeMonkey challenge, how do you reset the code to what it was in the beginning?</li> </ul>	
<b>Review</b>	<b>3 min.</b>
<p>Open challenge 6 and solve it with your class. They will solve it by themselves in the next lesson.</p>	
<b>Assignment</b>	<b>2 min.</b>
<p>Due next lesson, create a map with your route to school by writing the directions as computer instructions, just like you learned today. You can also route the way from your room to other places in your house, or even from your homeroom at school to the playground. Be sure to use the basic instructions used in CodeMonkey. Show an example of such a sequence of instructions on the whiteboard.</p>	

## Lesson 2 - Turn Around

Challenges 6-10

In this lesson, students will continue to explore the CodeMonkey platform by completing five more challenges. Prior to class, use the teacher dashboard to make sure all of your students have completed the first five challenges with three stars. It is important that all students are on the same level and no one is behind.

### Objectives:

Within this lesson, students will:

- Review what they learned in the previous lesson
- Identify the different ways to use “turn” instructions
- Complete challenges 6-10 on CodeMonkey

### Components:

- Instructions: “turn” using degrees, “step” backwards
- Terms: program, function, argument, statement, object

### US Standards Addressed:

- CCSS-Math Standards: PRACTICE.MP1, PRACTICE.MP3, PRACTICE.MP4, PRACTICE.MP5, CONTENT.3.MD.B.4, CONTENT.4.MD.C.5
- CCSS-ELA Standards: Literacy.RST.6-8.3, Literacy.RST.6-8.4

### Part 1: Introduction - 25 minutes

<b>Review</b>	<b>5 min.</b>
Collect the home assignments. Start with a brief discussion with the class of what was learned in the previous lesson: <ul style="list-style-type: none"><li>• What is coding?</li><li>• What instructions have we used so far? (step, turn)</li><li>• What is a programming language and which one do we use in CodeMonkey? (CoffeeScript)</li></ul>	
<b>Activity</b>	<b>5 min.</b>

Ask for three volunteers, giving each of them a role: one is the “Programmer,” one is the “Computer,” and the third is the “Character.” Now ask the “Programmer” to instruct the “Computer” to lead the “Character” to an object you placed in the classroom. Make sure the students use instructions properly (“step” with a number, “turn” left/right). As they go, write the instructions on the board to remind the other students of what they have learned. Repeat this activity with another group of three volunteers.

**Discussion**
**3 min.**

Ask the students, “Why do you need both a computer and a character? Why can’t one person be both?”

If we compare programming to the human body, then the programmer is the brain that sends instructions to the different parts of the body. The computer is responsible for making sure that the different parts of the body (“characters”) execute the instructions exactly as instructed.

**Explain**
**2 min.**

Introduce your students to the term *statement*: an element which expresses some action to be carried out. A computer program is a set of instructions which are simple tasks provided to the computer. These instructions are called statements. The instructions the “Programmer” gave earlier to the “Computer” are statements. Statements can be anything from a simple line of code to a complex set of conditions and formulas.

**Explain**
**3 min.**

This lesson is about turning and walking backwards. There are three ways to make a character turn; the first is to use “turn right/left” like we learned in the first lesson. In this lesson, we are introducing another way to turn.

Instead of turning right/left, we can turn by degrees. If your students have basic knowledge of degrees, such as a 360 degree turn or a 90 degree turn, then make a quick review of that knowledge. Otherwise, provide a short introduction to degrees. Optional: use a protractor.

**Explain**
**3 min.**

Objects are everything in the scene we can interact with, like the bush, bridge, banana, and turtle.

Each object has a set of actions it can do, like “step”, “turn”, or “turnTo” (we’ll learn about turnTo in the next lesson) for the monkey. These actions are called functions, and the input we add to them is called an argument. For example in “turn 10”, the argument is 10.

**Discussion**
**2 min.**



<ul style="list-style-type: none"> <li>• Ask the class to give you an example for a statement and write it on the whiteboard (possible results: step 10, step 15, turn right, turn left)</li> <li>• Ask what is the function in this statement (step or turn)</li> <li>• Ask what is the argument (10, 15, right or left)</li> </ul>	
<b>Explain</b>	<b>2 min.</b>
<p>Understanding the concept of walking backwards is pretty easy. If we want to go forward 15 steps, we type “step 15”, and if we want to go backwards, we type “step -15”. -15 will be read by the computer in this context just like “15 steps backwards”. If your students are older (6th grade and above), this is a good opportunity to talk about negative numbers on the number line.</p>	

## Part 2: Let's Go! - 15 minutes

<b>Login</b>	<b>1 min.</b>
<p>Ask the students to go to the CodeMonkey website (<a href="http://playcodemonkey.com">playcodemonkey.com</a>) and log in to their accounts. If a student is having trouble remembering his or her login information, use your list or the cards to remind the student of the assigned username and password.</p>	
<b>Play time</b>	<b>2 min.</b>
<p>All students should complete challenges 6-9 with at least two stars. (Students from the age of 12 and up should get three stars.) Use the teacher dashboard to keep track of students' achievements.</p> <p>Keep in mind that students might find turning with degrees difficult. You may need to provide extra help in levels 7 and 8. Use the Walkthrough below.</p>	
<b>Walk through</b>	<b>2 min.</b>
<p>Open challenge #7 and show the animation about angles. Use the ruler to measure the distance between the monkey and the banana, and show that ruler is also a protractor: it shows the number 45 which is the angle the monkey has to turn in order to face the banana. Show that this is the same number in the code. Make sure your students understand how to use the ruler as a protractor.</p>	
<b>Explain</b>	<b>2 min.</b>
<p>Open the challenge map and show your students the skill mode tab. Explain that in skill mode students can play through more challenges to perfect their CodeMonkey skills. These extra challenges are great practice and they only unlock after we complete certain</p>	

challenges. Hover over a locked challenge to show the unlocking tip. The first skill challenges will open for your students after they complete challenge 6.

Let students know that if they finish early, they can go to skill mode and complete unlocked challenges.

**Play time**

**5 min.**

The students continue working on levels 6-9.

In level 8, the students can use either “Turn left” or “Turn 90” to get three stars. Some of your students will probably use “turn left”. Make sure to emphasize that they can also use “turn 90” for the same result.

**Assessment**

**3 min.**

Challenge #10 is an assessment challenge that covers everything your students recently learned on CodeMonkey.

### Part 3: Debriefing - 5 minutes

**Review**

**2 minutes**

Check your students’ understanding of turning with degrees: Ask all of them to stand up and instruct them to “turn 90”, “turn 120”, and “turn 360”.

Repeat the explanation of turning by degrees: turn followed by a number turns the monkey by that number of degrees. For example, turn 90 turns the monkey the same as turn left.

**Review**

**2 minutes**

Check your students’ understanding of walking backwards: Stand with your back to the door and ask, “If I were the monkey what would be the right instruction to get me to the door?”. Emphasize that it should be one instruction, and not involve turning. Make sure their answer includes “step **minus** X”.

Repeat the explanation of stepping backwards: to step backwards a number of steps, add the minus sign (-) before the number. For example: step -10. The computer reads -10 in this context just like “10 steps backwards”.

**Assignment**

**1 minute**

Due next lesson, ask your students to include degrees in the navigation instructions from their homes to the school.

## Lesson 3 - I Have a Plan!

Challenges 11-15

This lesson revolves around planning. Everything we do in the physical world has to be planned, even if we sometimes do things automatically. We can cross the road without checking if it's clear, but that may result in a very dangerous outcome. Computers are the same; if we want to create a game or a program, we have to plan ahead and organize our instructions in the correct order.

### Objectives:

Within this lesson, students will:

- Review what they learned in the previous lesson
- Discuss the concept of planning and its importance in coding
- Complete challenges 11-15 on CodeMonkey

### Components:

- Instructions: `turnTo`, `turtle.step`
- Terms: planning

### US Standards Addressed:

- CCSS-Math Standards: PRACTICE.MP1, PRACTICE.MP3, PRACTICE.MP4, PRACTICE.MP5, CONTENT.3.MD.B.4, CONTENT.4.MD.C.5, CONTENT.6.NS.C.5
- CCSS-ELA Standards: Literacy.RST.6-8.3, Literacy.RST.6-8.4, Literacy.RST.9-10.3

### Part 1: Introduction - 20 minutes

Review	5 min.
<p>Collect the home assignments.</p> <p>Recall that we have learned two ways to turn, and ask the students to say some examples to both of them. The first way is turn right and turn left, the other one is turn 45, turn 30, etc.</p> <p>Introduce the third way to turn: by using “turnTo”. When using “turnTo” the computer identifies that there is another object present, besides our beloved monkey, and by calling its name, it knows which way to turn.</p> <p>Review the first two ways to turn by asking for a two volunteers and instruct each of them to explain and demonstrate one of the ways to turn we learned in the previous lesson:</p>	

- Direction (e.g. turn right)
- Degree (e.g. turn 180)

Write their answers with the examples on the board so your other students remember as well.

**Activity**
**5 min.**

To check your students' understanding of "turnTo" Play a short game in the spirit of Simon Says. Give instructions to your students to "turnTo" a specific place or a specific student. They should only turn when you say "turnTo", and not when you say "turn".

**Activity**
**5 min.**

In this lesson, your students will learn about planning.

Ask your students, "What do you do in the morning to get ready for school?"

Write their answers scattered on the board (not in a list).

Next, create a list out of the actions on the board, and put the tasks out of order, for example:

1. Get dressed
2. Take a shower
3. Wake up
4. Brush my teeth
5. Eat breakfast, etc.

**Discussion**
**5 min.**

Ask your students, "Is this the order of actions you will take to get ready in the morning?" When they say no, ask them to explain why not.

The point of this activity is to show the students the importance of planning. We plan our day and the order in which we do things; sometimes we do this without thinking and sometimes we plan every step.

Explain to your students that when we write code, we have to consider that computers read the code from TOP to BOTTOM, and we have to think ahead about the order of instructions. When we have just one object, this isn't a big problem (in our case, the monkey is the object). But what happens when we want to control another object? How do we know who should be instructed to go first?

In this lesson's challenges, your students will meet our trusty turtle and will have to use his help to get more bananas. In order to do so, they will have to think ahead and plan how to write the code.

**Part 2: Let's Go! - 20 minutes**

<b>Login</b>	<b>1 min.</b>
Ask the students to go to the CodeMonkey website ( <a href="https://playcodemonkey.com">playcodemonkey.com</a> ) and log in to their accounts. If a student is having trouble remembering his or her login information, use your list or the cards to remind the student of the assigned username and password.	
<b>Play time</b>	<b>2 min.</b>
All students should complete challenges 11-15 with at least two stars. (Students from the age of 12 and up should get three stars.) Use the teacher dashboard to keep track of students' achievements.  After 2 minutes, use the following walk through:	
<b>Walkthrough</b>	<b>6 min.</b>
<p>Open challenge #12 and show the animation. It explains how to use objects on the screen. After the animation, walk your students through the following steps:</p> <ol style="list-style-type: none"><li>1. Hover over the bridge, show that the word bridge appears on the screen</li><li>2. "bridge" is the name of that object.</li><li>3. Highlight the word banana in the editor</li><li>4. Click on the bridge and show how the word banana is replaced by bridge</li><li>5. Move the cursor by clicking on row 3 after the word turnTo</li><li>6. Click the banana and show how the word banana is entered into the code</li><li>7. Move the cursor to line 4 and write "step 10"</li><li>8. Run the solution</li><li>9. Click replay to go back to your solution</li><li>10. Delete all the code to start from blank.</li><li>11. Now you will demonstrate how to use even more clicking instead of typing.</li><li>12. Hover over the block "step" at the bottom of the editor, show how a description shows up</li><li>13. Show the descriptions that show up when hovering over every block</li><li>14. By clicking step, turnTo, bridge, and banana, reach the following solution:</li></ol> <pre>turnTo bridge step 10 turnTo banana step 10</pre> <p>15. Make sure you have only used the keyboard for typing the number and jumping to the next line.</p>	

16. Make sure your students understand how to use <u>clicking and hovering</u> for object on the stage (banana, bridge) and for blocks at bottom (turnTo, step).	
<b>Play time</b>	<b>6 min.</b>
The students continue working on challenges 11-15.	
<b>Assessment</b>	<b>5 min.</b>
Challenge #15 is an assessment challenge that covers everything your students recently learned on CodeMonkey.	
<b>Practice</b>	
Encourage students who finish early to open skill mode on the map and complete unlocked challenges.	

### Part 3: Debriefing - 5 minutes

<b>Discussion</b>	<b>4 min.</b>
Open level 14 and ask your students, "How did you plan what to write in your code?" Make sure to lead them to the correct answer, explaining the right train of thought needed when planning the code. We should first think about what steps should be taken to achieve our goal (in this case, get the banana), and then break the steps into separate statements, while deciding what should come first (should the turtle or monkey go first?). If we tell the monkey to move before the turtle is in the right place, he is going to fall in the water, and monkeys don't like water.	
<b>Review</b>	<b>1 min.</b>
Use this opportunity to remind your students that a program is a set of instructions, or simple tasks provided to a computer. These instructions are called statements. Statements can be anything from a single line of code to a complex mathematical equation.	

## Lesson 4 - Turtle Lake

Challenges 16 - 20

In the previous 3 lessons your students have learned how to move around using code. They have actually mastered the foundation to programming, as they are now able to write a block of code that will carry out the instructions they intend to give the computer. We will take the current lesson to practice and reinforce this knowledge, and to deepen their understanding of what is actually going on.

### Objectives:

Within this lesson, students will:

- Practice using functions with different objects (monkey, turtle)
- Complete challenges 16-20 on CodeMonkey

### Components:

- Instructions: `turtle.turnTo`
- Terms: syntax

### US Standards Addressed:

- CCSS-Math Standards: PRACTICE.MP1, PRACTICE.MP3, PRACTICE.MP4, PRACTICE.MP5, CONTENT.3.MD.B.4, CONTENT.4.MD.C.5
- CCSS-ELA Standards: Literacy.RST.6-8.3, Literacy.RST.6-8.4, Literacy.RST.9-10.3, Literacy.RST.9-10.4, Literacy.RST.9-10.7

### Part 1: Introduction - 10 minutes

Explain	5 min
<p>Recall with your students that in CodeMonkey we are writing code in a programming language called CoffeeScript. As they experienced in the previous lessons, the code has to be written in a particular way in order for the computer to do what we are trying to achieve.</p> <p>Explain that this is because a programming language, just like any language, has its own rules on how things can be said or written. In programming this is called the <i>syntax</i> of the language.</p> <p>There might be more than one correct way to say or write a certain thing in CoffeeScript, just like in English or any language. An important difference between programming languages and other languages is this:</p>	



In a spoken language, sometimes we can say something incorrectly but still be understood. However, with the computer - even the slightest mistake will definitely cause our code to fail. So we always have to pay attention to syntax and be very accurate.

For example, if we forget a dot or a space in `turtle.step 10` we will get `turtle step 10` or `turtle.step10`, and the code will not do the right thing.

### Walkthrough

5 min.

Open challenge #15 and click “reset” to reset the code to what it was initially (blank).

Use typing only, no clicking, and enter the following code:

```
turtle step 10
```

Click run to execute the code. Read out loud the error message that appears. Explain that the dot is important. In this example the computer was able to guess what we meant, but this is not always the case.

Edit the code to this code:

```
turtle.step 10  
step15
```

Execute it and read the error message with the students.

Repeat the same with the following modification to the 2nd line (capital S):

```
turtle.step 10  
Step 15
```

And with the following (without breaking between lines):

```
turtle.step 10 step 15
```

Conclude that spelling, punctuation, capitals and new lines are part of the syntax and are essential for our code to do what we want.

Finally, run a 3-star solution:

```
turtle.step 10  
step 15
```

When it completes, click replay and edit it to the following:

```
turtle.step 10  
monkey.step 15
```

Conclude with your students that `step` and `monkey.step` can be used interchangeably, because the computer assumes we are referring to the monkey. When we refer to the turtle or any other object, we must use its name.

## Part 2: Let's Go! - 30 minutes

<b>Login</b>	<b>1 min.</b>
Ask the students to go to the CodeMonkey website ( <a href="https://playcodemonkey.com">playcodemonkey.com</a> ) and log in to their accounts. If a student is having trouble remembering his or her login information, use your list or the cards to remind the student of the assigned username and password.	
<b>Play time</b>	<b>25 min.</b>
<p>All students should complete challenges 16-20 with at least two stars. (Students from the age of 12 and up should get three stars.) Use the teacher dashboard to keep track of students' achievements.</p> <p>Note that challenge #16 is a tricky one to achieve three stars. Make sure your students do not stay on this challenge for too long and encourage them to keep going and come back to it if they have time left. At the end of the lesson, you can open a discussion regarding this challenge and try to solve it together with your students in order to get those sneaky three stars.</p> <p>In challenge #19 there are different ways to make the monkey turn the right way after catching 3 bananas. One way is by using the island</p> <pre>turtle.turnTo island</pre> <p>and one way is by using any of the bananas along that path e.g:</p> <pre>turtle.turnTo bananas[3]</pre> <p>In both cases, hovering and/or clicking will do the trick. Remind your students that hovering over an object shows its name, and clicking enters that name into the editor.</p> <p>If your students ask you about the meaning of something like bananas[3] just tell them that it's the way to access a particular banana and we will get back to it later on.</p>	
<b>Assessment</b>	<b>4 min.</b>
Note that challenge #20 is an assessment challenge that covers everything your students recently learned on CodeMonkey.	
<b>Practice</b>	
Encourage students who finish early to open skill mode on the map and complete unlocked challenges.	

### Part 3: Debriefing - 5 minutes

<b>Walk through</b>	<b>5 min.</b>
<p>Open challenge 2-7 in skill mode and solve it with your class. Ask them to explain how they <b>plan</b> the solution for this challenge. You can even invite a student to solve this challenge in front of the class.</p> <p>The trick in this challenge is similar to the one in challenge 16 - tell the monkey to walk backwards in order to have less lines of code, and to get the third star.</p> <p>If at first try your students can't get the third star, ask them if this challenge seems similar to one they've solved before. Explain that It's fairly common to use references from old projects when programming, or even full blocks of code, and in CodeMonkey they are encouraged to go back to old challenges to get inspiration or help.</p>	

## Lesson 5 - In the Loop

Challenges 21-25

Congratulations! You have passed the introductory part of CodeMonkey. You and your students now hold basic programming skills. This lesson will focus on loops. There are different kinds of loops, like “for loops” and “until loops”, but first we will learn how to use a simple loop.

### Objectives:

Within this lesson, students will:

- Define *loop* as a programming term
- Understand why using loops in programming is more efficient
- Complete challenges 21-25 on CodeMonkey

### Components:

- Instructions: “x.times ->”
- Terms: loop
- Feature: tab (indentation)

### US Standards Addressed:

- CCSS-Math Standards: PRACTICE.MP1, PRACTICE.MP3, PRACTICE.MP4, PRACTICE.MP5, CONTENT.3.MD.B.4, CONTENT.4.MD.C.5
- CCSS-ELA Standards: Literacy.RST.6-8.3, Literacy.RST.6-8.4, Literacy.RST.9-10.3, Literacy.RST.9-10.4, Literacy.RST.9-10.7

### Part 1: Introduction - 25 minutes

Discussion	5 min.
<p>Programming is not only about writing the correct statements in the right order; it is also about knowing how to write clear and short code.</p> <p>Let's imagine that we have to write a simple program to make the monkey climb up a long staircase of 100 steps, and we can only use the function “stepUp” that makes the monkey climb up one step at a time.</p> <p>Ask your students, “Do you think that the programmer wrote a line of code for every stairstep?” Just imagine how LONG this code would be - 100 lines of code!</p>	

So, instead of code that looks like this (100 times):

```
stepUp  
stepUp  
stepUp  
stepUp  
stepUp  
...
```

Wouldn't it be great to write something shorter? Ask the students to suggest a shorter way.

How about something like this?

```
stepUp 100 times
```

Luckily, this is possible. Not exactly the way we just wrote it now, but quite similar. Code that is written in such a way is called a loop.

### Explain

5 min

Explain to your students that a “simple loop” is a sequence of instructions that repeats a specified number of times. There are also other kinds of loops (for loops, until loops) that last until a particular condition is met, but we will learn about those later on.

Back to the stairs example, the way to write that in CodeMonkey would be:

```
100.times ->  
    stepUp
```

The number represents the number of times that we want the code inside the loop to run.

Note the special syntax: the dot between the number and the word times, the space before the ->, and the *indentation* of the code inside the loop (stepUp is the code inside the loop).

Make sure the students know how to use the “Tab” key on their keyboard in order to get indentation into the code. Another alternative is to press the spacebar four times.

Remember that you can click the “times” button at the bottom in order to get a loop into the code without having to worry about the syntax.

### Activity

10 min.

Let's show another example to better clarify the use of a simple loop. Write the following on the left hand side of the board:

```
step 10
turn left
step 10
turn left
step 10
turn left
step 10
turn left
```

Ask your students to identify a **repeating pattern** in the code. The pattern they identify should be:

```
step 10
turn left
```

Now, next to that code, on the right hand side, write:

```
4.times->
  step 10
  turn left
```

Ask your students what they think each block of code does.

Explain that each code is the same; only the right hand side of the code is written as a loop. Once we found the pattern on the left hand side, all we had to do is just write it once, and add 4.times->. The resulting code does the same, but is shorter and more readable.

The meaning of the code on the right is that “step 10, turn left” would repeat four times, and then the loop would be over. Once the loop is over, the computer moves to the next statement.

## Part 2: Let's Go! - 15 minutes

<b>Login</b>	<b>1 min.</b>
Ask the students to go to the CodeMonkey website ( <a href="https://playcodemonkey.com">playcodemonkey.com</a> ) and log in to their accounts. If a student is having trouble remembering his or her login information, use your list or the cards to remind the student of the assigned username and password.	
<b>Play time</b>	<b>10 min.</b>
All students should complete challenges 21-25 with at least two stars. (Students from the age of 12 and up should get three stars.) Use the teacher dashboard to keep track of students' achievements.	

<b>Assessment</b>	<b>4 min.</b>
Challenge #24 is an assessment challenge that covers everything your students recently learned on CodeMonkey.	
<b>Practice</b>	
Encourage students who finish early to open skill mode on the map and complete unlocked challenges.	

### Part 3: Debriefing - 5 minutes

<b>Walkthrough</b>	<b>3 min.</b>
<p>Open challenge #25 and click the reset button to reset the code. Go over the code with your students. Read the statements aloud, slowly and clearly. This Walkthrough is intended to show your students how to read code correctly.</p> <p>Walk your students through the process of identifying the <b>pattern</b> of bananas arranged in a <b>L</b> shape, and translate that into the sequence of statements:</p> <pre>turn left step 5 turn right step 5</pre> <p>Then fix the inside of the loop to match the sequence of statements and hit the run button. This will not solve the challenge as the loop runs 3 times instead of 4.</p> <p>Ask your students how many times does the L pattern repeat itself? The solution is the last help needed in order to solve this challenge correctly, i.e. replacing the 3 by 4.</p>	
<b>Explain</b>	<b>2 min.</b>
<p>Imagine you had to give instructions to somebody to find a place that is 5 blocks down the street. Do you say: walk a block, then walk another block, then another, then another, and then one more. No. You simply say: walk 5 blocks down the street. That is because the <u>same action</u> has to be done more than once.</p> <p>Remind your students that it's the same in coding. When there is a repeating pattern of things to do, then a loop is a good way to keep the program short and easy to understand. Just find the pattern, write it once and add the line of code that tells the computer how many times to repeat.</p>	





## Lesson 6 - Loop on

Challenges 26-30

Today your students will continue using simple loops, and will deepen their understanding on why it is important to use loops.

### Objectives:

Within this lesson, students will:

- Understand why using loops in programming is more efficient
- Understand the importance of using indentation right
- Complete challenges 26-30 on CodeMonkey

### Components:

- Orange highlight over running code

### US Standards Addressed:

- CCSS-Math Standards: PRACTICE.MP1, PRACTICE.MP3, PRACTICE.MP4, PRACTICE.MP5, CONTENT.3.MD.B.4, CONTENT.4.MD.C.5
- CCSS-ELA Standards: Literacy.RST.6-8.3, Literacy.RST.6-8.4, Literacy.RST.9-10.3, Literacy.RST.9-10.4, Literacy.RST.9-10.7

### Part 1: Introduction - 10 minutes

Review	5 min.
<p>We begin this lesson with a review of the concept of loops and the way to use them.</p> <p>Ask the class:</p> <ul style="list-style-type: none"><li>• What is a loop?</li><li>• When do we use loops in our code?</li><li>• What is a syntax of a loop that makes the monkey walk in the shape of a square?</li><li>• How do we tell between code that's inside the loop and code that is after the loop?</li></ul>	

### Part 2: Part 2: Let's Go! - 30 minutes

Login	1 min.
-------	--------

Ask the students to go to the CodeMonkey website ([playcodemonkey.com](https://playcodemonkey.com)) and log in to their accounts. If a student is having trouble remembering his or her login information, use your list or the cards to remind the student of the assigned username and password.

**Play time****4 min.**

All students should complete challenges 26-30 with at least two stars. (Students from the age of 12 and up should get three stars.) Use the teacher dashboard to keep track of students' achievements.

The main thing to watch in this play time session is the use of loops. Most of the levels that involve a repeating pattern can be solved by repeating the same or similar code. However, this misses the point, so you should make sure the students are actually using loops and getting at least 2 stars.

4 minutes into the play time session, use the following walk through:

**Walkthrough****5 min.**

Open challenge #27.

1. Draw this challenge on the whiteboard. You can use a shape (smiley, etc.) to represent the monkey.
2. Ask one of the students to draw on the whiteboard and guess the path that the monkey has to go to get all the bananas. The answer should be a + shaped path.
3. Ask another student to translate the path into a sequence of steps and turns without loops. The answer is something similar to the following (the direction right or left can be different).

```
step 10
step -10
turn right
step 10
step -10
turn right
step 10
step -10
turn right
step 10
step -10
turn right
```

4. The answer may be longer, like the following:

```
step 10
turn 180
step 10
```

```

turn right
step 10
turn 180
step 10
turn right
step 10
turn 180
step 10
turn right
step 10
step 180
step 10
turn right

```

5. If so, ask the student to improve it by making it shorter. A hint on how to do this would be to walk backwards.
6. Ask another student to identify the recurring pattern. The recurring pattern in the example above is:

```

step 10
step -10
turn right

```

7. Ask the student to make the code shorter by using a loop, expect an answer like this:

```

4.times ->
  step 10
  step -10
  turn right

```

**Play time**

**5 min.**

The students continue their work on challenges 26-30.

**Walkthrough**

**5 min.**

Open challenge #28 and click “reset” to reset the code.

Observe with your students that the existing code will go through the half-circle and collect all the bananas except one. Conclude that the missing line of code to complete the challenge is “step 10” (use ruler if necessary).

Try to type “step 10” and click “run” to see what happens. This does not solve the challenge (hit “stop” if this goes on too long).

Observe that the problem in this solution is that the computer takes the “step 10” as if it should be executed 10 times in the loop, not 1 time after the loop.

Go back to your code and remove the indentation before the step 10. Now run your code again. Show your students that this is how to run code after the loop. Remind them that this is called *indentation*.

**Play time**

**10 min.**

The students continue their work on challenges 26-30.

**Practice**

Encourage students who finish early to open skill mode on the map and complete unlocked challenges.

### Part 3: Debriefing - 5 minutes

**Walkthrough**

**5 min.**

Open challenge #29 and click the reset button to reset the code. Go over the code with your students. Read the statements aloud, slowly and clearly. This Walkthrough is intended to show your students how to read code correctly.

Click run to run the code, and direct your students' attention to the orange highlight which highlights the line of code that is being processed by the computer at that very moment.

Solve challenge #29 with your students. You can also open one of your student's solutions anonymously using the teacher dashboard.

## Lesson 7 - Three stars party!

Challenges 1-30

In this lesson, your students will revisit challenges they have already solved but received only one or two stars for their solution. By the end of this lesson, all of your students should have perfect three-star scores on the first 30 challenges of CodeMonkey.

### Objectives:

Within this lesson, students will:

- Revisit challenges where they received one or two stars
- Solve all challenges with three stars

### US Standards Addressed:

- CCSS-Math Standards: PRACTICE.MP1, PRACTICE.MP3, PRACTICE.MP4, PRACTICE.MP5, CONTENT.3.MD.B.4, CONTENT.4.MD.C.5
- CCSS-ELA Standards: Literacy.RST.6-8.3, Literacy.RST.6-8.4, Literacy.RST.9-10.3, Literacy.RST.9-10.4, Literacy.RST.9-10.7

### Part 1: Introduction - 15 minutes

**Prior to class:** Check your teacher dashboard for challenges where a high number of students struggled to get three stars. Use the “challenges by difficulty” metrics in the statistics bar in your dashboard to see which challenges were the hardest for your students, bigger circle means more students struggled.

<b>Walk through</b>	<b>15 min.</b>
Choose two or three challenges that got a relatively high number of blue or red stars, and solve them together in class with your students.	

### Part 2: Let's Go! - 25 minutes

<b>Play time</b>	<b>25 min.</b>
Ask the students to go to the CodeMonkey website ( <a href="https://playcodemonkey.com">playcodemonkey.com</a> ) and log in to their accounts.	

Once they are logged in, instruct them to click the map (upper-right corner) and find challenges where they got one or two stars.

At the end of class, all students should aim at having three stars in all the first 30 CodeMonkey challenges.






If any of your students finished all 30 challenges with 3 stars, ask them to help their fellow classmates.

### Part 3: Debriefing - 5 minutes

Explain	5 min.
<p>Discuss briefly with your students the importance of writing short code.</p> <p>In CodeMonkey, when we get two stars, it means there is a shorter way to reach the same end result. Either we have lines of code that are unnecessary for reaching the end result, or there is a shorter solution, like using a loop.</p> <p>Imagine that every time you wanted to go to your next class, you had to first go home and then come back and go to the classroom. It doesn't make any sense to do that. Writing long code is the same. If there is a shorter way to do the same thing, it doesn't make any sense to do it the long way.</p>	



## Reference Card

Keyword / Button	Description
	To make the monkey “step” to a certain distance, we have to write “step X” using the number of steps we want him to take, for example, “step 10”. Pressing the <b>step</b> button will write the word “step” in your code.
	<p>“Turn” should be accompanied by a direction (left/right) or degrees (45, 90, 180).</p> <p><b>Examples:</b> “turn right”, “turn 90”</p> <p>Pressing the <b>turn</b> button will write the word “turn” in your code.</p>
	<p>“Left” and “right” are used after the statement “turn” to make the monkey turn in the desired direction.</p> <p>Pressing the <b>left</b> or <b>right</b> buttons will write the word “left” or “right” in your code accordingly.</p>
	<p>“turnTo” is another way of turning. Instead of using direction or degrees, we are asking the monkey to turn to a specific object, for example, “turnTo banana”. Pressing the <b>turnTo</b> button will write the word “turnTo” in your code.</p>
	<p>A simple loop is a sequence of instructions that repeats a specified number of times.</p> <p><b>Example:</b></p> <pre>3.times -&gt; ... step 5 ... turn left</pre> <p>In this example, the monkey will repeat “step 5, turn left” three times. The instructions we write in the loop should be written underneath it with an indentation (...). You can do that by pressing the <b>Tab</b> key on the keyboard.</p> <p>Pressing the <b>times</b> button will write the beginning of a simple loop in your code: “3.times - &gt;”.</p>



Pressing the **run** button will make the code on the right run. You can see the outcome by looking at the scene on the left.



The reset button will erase everything you wrote in the code on the right and will reset the code to how it was at the beginning of the challenge.





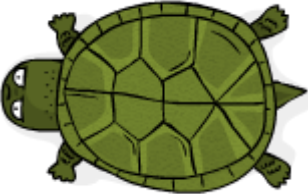
The ruler is a tool to help you measure the distance between different objects in the game, for example, the distance between the monkey and the banana. The ruler can also help you measure angles that the monkey or turtle has to turn in order to face another object on the screen, like a banana. To use the ruler, click it once, and then use your mouse to move the ruler to the point you want it to start measuring from. Click the mouse again, and then drag it to the end point. A number will appear at the end point to indicate the distance. Use this number with the “step” statement. Another number will appear near the starting point, this number indicates the angle from the first object to the second one, use it with the “turn” statement.



After each challenge, you’ll receive a star-rating for your solution. The stars are distributed as so:

- First star is given if you got all bananas
- Second star is given if you used what you learned
- Third star is given if your code is short and to the point

## Characters Review

Character	Description
	Gordo, named after the first ape in space, is the guide who will help you and give you instructions along the way. His remarks are both funny and helpful. You can always click him to re-read the instructions.
	The monkey is the main character. You will have to help him collect bananas by writing lines of code. Just so you know, monkeys don't like to get wet, and they are very friendly.
	<p>In challenge #13, you will meet our trusty turtle. The turtle will help you get those sneaky bananas. In order to instruct the turtle to "turn" or "step", we have to click it first. This will write its name in the code, and then separate it from the action we want it to take using a dot (.).</p> <p><b>Example:</b></p> <pre>turtle.step 10</pre>

## Support

Need extra help?

You can contact us via

email: [info@cm-studios.com](mailto:info@cm-studios.com).

twitter: [www.twitter.com/codemonkeystu](https://www.twitter.com/codemonkeystu)

facebook: [www.facebook.com/codemonkeystu](https://www.facebook.com/codemonkeystu)