CodeMonkey
LESSON PLANS

Beaver Achiever

Chapter 2

Conditional Loops

LESSON PLANS

2345 Yale St., 1st floor
Palo Alto, CA 94306
info@codemonkey.com
www.codemonkey.com

Welcome to back to Beaver Achiever! In this course your students will use block-based coding to help the beaver building a house and roof.
In this course the students will learn about conditional loops that repeat based on a certain condition. Following this chapter your students will master the use of loops - combination between loops, nested loops and more.
At the end of the course, you can assign a quiz that includes 5 challenges to test your students' knowledge.

To learn how to set up a class, please read A Beginner's Guide to CodeMonkey. The guide can also be found in the Teacher's Resources Menu on your homepage.
Please email us at info@codemonkey.com for any questions you may have along the way.

Have fun!
The CodeMonkey Team

LESSON PLANS

Table of Content

Lesson 1 – Introduction and Warm Up Challenges

In the first lesson of this chapter, we will help the Beaver build its house by chopping planks to fit the wall.

This chapter will introduce new blocks and will use an until loop.

To remind your students how to log in and the game mechanism, please go to lesson 2 on Beaver Achiever - Sequencing and Simple Loops.

Objectives

In this lesson, students will:

- Learn new blocks - pick, chop
- Understand the power of conditional loop
- Work with “until” loops
- Complete challenges 1-6

U.S. Standards Addressed

CSTA-K12 Computer Science Standards	
★ 1A-NI-04	★ 1B-AP-08
★ 1A-AP-08	★ 1B-AP-10
★ 1A-AP-10	★ 1B-AP-11
★ 1A-AP-11	★ 1B-AP-13
★ 1A-AP-12	★ 1B-AP-15
★ 1A-AP-14	★ 1B-AP-16
★ 1A-AP-15	★ 1B-AP-17
★ 1A-IC-18	

Part 1: 25 Minutes

Introduction

Game	**6 mins.**
7 Boom Arrange the class to stand in a circle and choose the first player. The students will count one by one from 1. Every time that the player hits a number that can be divided by 7 or 7 is a part of the number, the current player must shout "BOOM!". If the player misses a “BOOM” he/she will go back to their seat.	
Playing the game 1. Explain only the “BOOM” mechanism, do not explain the end-point of the game or how to win. 2. Guide the students to start playing, let the class play a full round.	

Part 1: 25 Minutes

Introduction cont.

Discussion	**6 mins.**

1. Ask the students:
   a. Can they tell in advance how long will it take to play? how many turns?
   b. How can they explain the game to someone that is not familiar with the game?
   c. How can you define when to stop the game? which terms are used?
2. This game is about conditional loop
   a. A players keeps playing **until** he misses a “BOOM”.
   b. The game continues **until** there is a single player left.
3. Can they think of other repeated activities that we know when to stop but we can not declare a certain number of repetitions?
   a. Discuss activities like - Eating (until you are full or finish your dish), Riding Bike (until reaching your destination) - these are repeated activities that repeats until a certain condition is met.

Define Conditional statement - conditional statement can be either **true** or **false** and is a key part of the decision-making process for humans and computers.
For example - play until there is a winner: as long as the answer to the statement - “there is a winner” is **false** you keep on playing.

Part 1: 25 Minutes

Introduction cont.

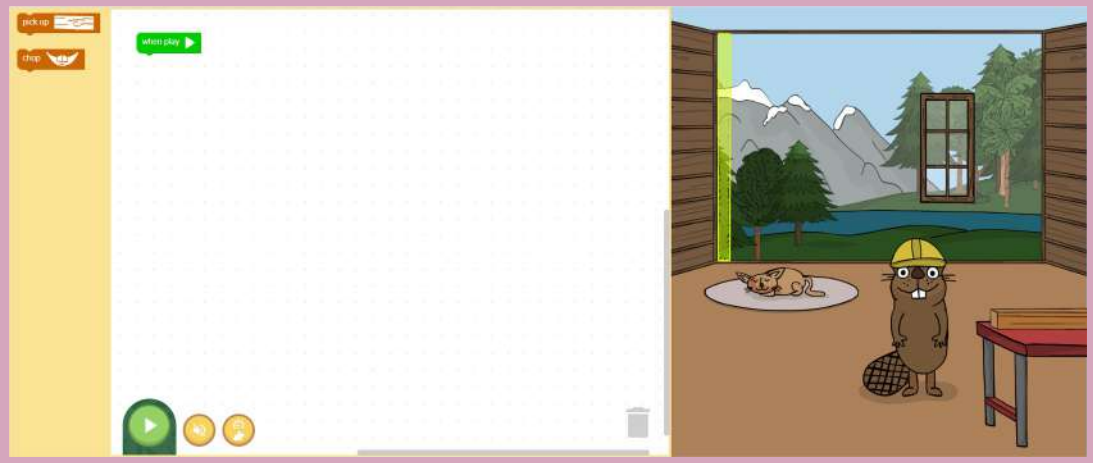
Back to the Beaver - walk through	**3 mins.**

Open challenge #1 - Present the new scene and describe the challenges in this chapter - helping the Beaver building its house by chopping planks.

In this lesson, we will learn how the Beaver uses planks to build its house.
1. The Beaver needs to **pick-up** a plank
2. The Beaver needs to **chop** the plank to fit the desirable size
There are 3 different sizes - Large (L, yellow), Medium (M, blue) and Small (S, pink).

Challenge #1 is a guided challenge that introduces the pick-up and chop blocks with a single **L** plank to fill.
We can see the highlighted missing plank on the wall.

Ask the students how many times the Beaver will chop the plank?
We do not know the exact number of times to reach a L-size plank, this will be part of the guided solution.
The students will follow the guided instructions and will complete the challenge.



Part 1

For the Teacher

Until Loop - The mechanism

As humans we know when to stop a repeated action like eating, riding bike, and more.

In programming we define very specific instructions so the computer can perform the task.

When the repeated instruction was numbered it was easy to describe and understand, but when we face a situation where the repeated execution is not defined with specific number of times, we need to translate the end point to a conditional statement.

This is where conditional loops come in handy.

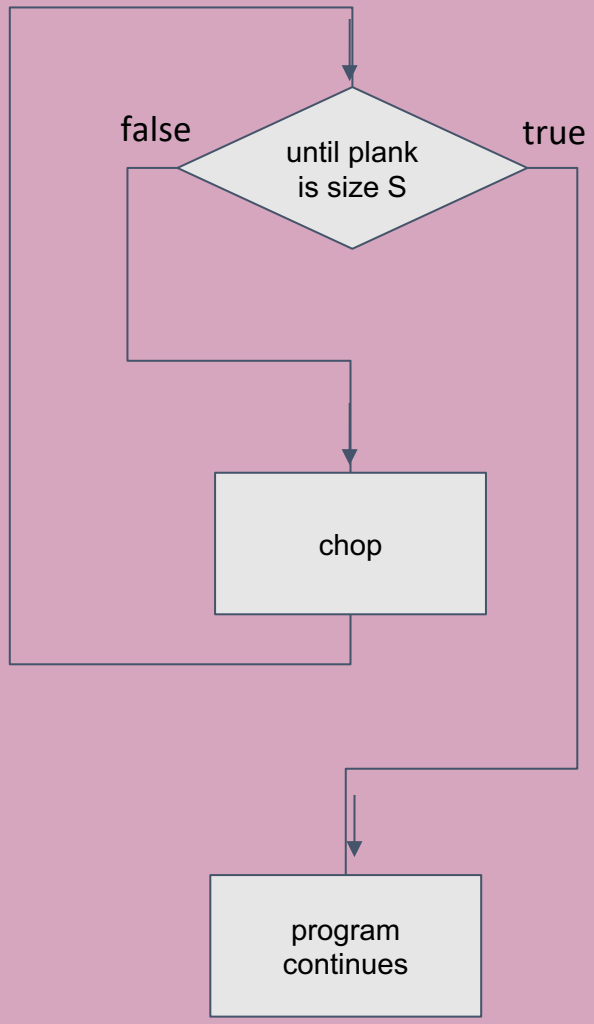
The “until” loop contains a block of code that will repeat *until* “something happens”, or more accurately, until a specific condition is met. This condition is called a *control expression* or *loop condition*. The computer checks the condition at every repetition.
If the answer is false, the loop will keep going. It will only stop once the answer is true.

Part 1

For the Teacher

Until Loop - The mechanism

The flowchart on the right illustrates the until loop mechanism - emphasize the loop continues as long as the answer is false, once the answer is true the loop ends, and the program continues.



Part 1: 25 Minutes

Introduction cont.

Explain & Walk through	**10 mins.**

Open challenge #2 - This time the Beaver needs to chop the plank to reach a **S**-size plank.
Ask the student

- How many chops are needed?
- Is there a different way to guide the Beaver to stop chopping?
- Present one of the examples you discussed before, for example, ask how do we know when to stop eating? can we do the same with chopping?

Introduce the until loop, a conditional loop, that is used for repeated execution when we ca not define the number of iterations in advance, but we do know when to stop iterating.

In a repeat loop we defined the number of times. In an **until loop,** we will define a conditional statement that marks the end of the process.

For example - bike until you reach your destination - **until** (as long as) the answer to the statement - "**reached the destination?**" is **false** you keep on biking.

Until loops will run as long as the condition is false and only end when the condition becomes true.

Until when the Beaver will chop the plank? Until the plank reaches the desirable size.
How does it work? After each chop we check if the plank has reached the defined size (conditional statement) if false (no) or true (yes). If False - continue, otherwise - stop.

Part 2: 15 Minutes

Playtime

Log-in	**3 mins.**
Go to app.codemonkey.com. Instruct your class on how to log in to their CodeMonkey account. If your 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 log-in cards. If a student forgets their password, you can reset it by visiting the classroom dashboard, locating the student's username, and clicking on the edit button which will appear if you hover over the username.	
Playtime	**12 mins.**
All students should complete challenges 1-6 with three stars. Challenges 1-2 are the guided challenges you presented to the class. Challenges 3-6 are simple challenges with a single & simple use of the until loop, each challenge for a different plank size with a different location in the wall layout. Use the classroom dashboard to keep track of students' achievements. Use this time to walk around the class and help students who are struggling.	

Part 3: 5 Minutes

Debriefing

Summary	5 mins.

This was a lesson with many new terms, ask the students:

1. Which new blocks did we use? pick-up, chop, until
2. What is the order between pick-up and chop?
3. Ask your students to identify the types of loops they have learned? Have a brief discussion about repeat loops (a sequence of instructions that repeats a specified number of times) and until loops (performs a set of actions until a condition is met)
4. Write on the board “The until loop is performed until the condition is True/False” - ask the students to select between true or false

Lesson 2 – Let's Build our House

This lesson is designed to deepen the understanding of the until loop with more complex sequences that include several consecutive loops (separate loops one after the other).

Objectives

In this lesson, students will:

- Practice until loop
- Learn how to use until loops in consecutive order
- Complete challenges 7-10

U.S. Standards Addressed

CSTA-K12 Computer Science Standards	
1A-NI-04	1B-AP-08
1A-AP-08	1B-AP-10
1A-AP-10	1B-AP-11
1A-AP-11	1B-AP-13
1A-AP-12	1B-AP-15
1A-AP-14	1B-AP-16
1A-AP-15	1B-AP-17
1A-IC-18	

Part 1: 20 Minutes

Introduction

PREPARE IN ADVANCE:

For this activity, please prepare a fun playlist you can play in class for the freeze dance game.

Game & Discussion	**5 mins.**
Freeze dance Write on the board: stand up until the music stops dance freeze Play the game by asking the students to follow the instructions on the board. Play the music once, give the student time to dance and then stop, all the students should freeze.	
This game is a fun reminder of the until loop we learned in the previous lesson. Ask the students to define how an until loop is different from a repeat loop.	

Part 1: 20 Minutes

Introduction cont.

Analyzing code	**10 mins.**

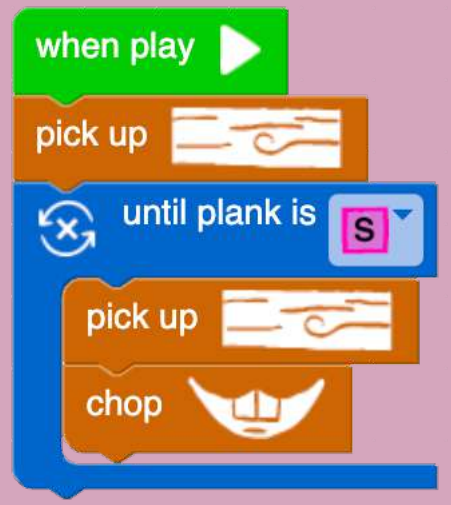
Present code #1 and ask the students what they think will happen when the code is executed?
The loop will not start because no planks were picked. There is no plank to check if its size is **S**-size.

Present code #2 and ask the students what they think will happen when the code is executed?
The Beaver will pick up the plank → the loop will start executing [the Beaver will chop the plank once and will try to pick-up another plank] → since there is only one plank the code will stop.

Discuss the importance of defining the correct order of instructions before and within the loop.

1

2



Part 1: 20 Minutes

Introduction cont.

Explain & walk through	**5 mins.**

Open challenge #7 - This time the Beaver needs to chop 2 different planks.
Ask the students

- How is this challenge different from challenge #6?
- How can we solve the problem?
  - Can we chop 2 different planks simultaneously? why not?
    - The different plank size requires different conditional loops
  - What is the difference between simultaneously and sequentially?
    - Simultaneously - at the same time (together) in a single loop
    - Sequentially - one after the other, 2 separate loops, one for each plank
  - How do the Beaver pick up the planks? can we pick 2 planks together?
- What is the correct order of instructions?
  - pick-up
  - until
    - chop
  - Pick-up
  - until
    - chop

Part 2: 20 Minutes

Playtime

Playtime	**20 mins.**
All students should complete challenges 7-10 with three stars. Use the classroom dashboard to keep track of students' achievements. Use this time to walk around the class and help students who are struggling.	

Part 3: 5 Minutes

Debriefing

Summary	5 mins.

Encourage the students who completed the challenges to go back and change the order of the instructions to test different solutions.

Testing is very important for programmers – it is an important phase of software developing since it assure correctness of the code and help us discover defects and bugs. Also, it allows the programmer to analyze the quality of the code - in terms of efficiency and clarity.

Lesson 3 – There is Plenty of Work

What happens if the Beaver has multiple planks of the same size?

This lesson is focused on more advanced loop mechanism. Nested loops - combining a repeat loop with an until loop.

The topic of nested loops was first presented towards the end of the previous chapter. Repeating a topic in a spiral manner allows us to deepen the understanding and gain more practice time.

Objectives

In this lesson, students will:

- Practice new mechanisms to implement nested loop
- Define the role of the outer loop and the inner loop
- Complete challenges 11-15

U.S. Standards Addressed

CSTA-K12 Computer Science Standards	
1A-NI-04	1B-AP-08
1A-AP-08	1B-AP-10
1A-AP-10	1B-AP-11
1A-AP-11	1B-AP-13
1A-AP-12	1B-AP-15
1A-AP-14	1B-AP-16
1A-AP-15	1B-AP-17
1A-IC-18	

Part 1: 20 Minutes

Introduction

What is the difference	5 mins.

Start the lesson by reminding the students about the previous lesson, open challenge #9 and ask them how they would solve it. Discuss the use of two until loops, in a consecutive order.
Then, present the image on the right taken from challenge #11 (do not open the challenge as it leads to the solution).
Ask the students to characterize the difference between the two challenges.

Challenge #9

Challenge #11

Part 1: 20 Minutes

Introduction cont.

What is the difference	**10 mins.**

These two challenges present two different problems -

- Challenge #9 - 2 different planks means two different conditional checks in the until-loop block.
- Challenge #11 - 2 identical planks → the same conditional checks in the until-loop block.
- Can we use a single until-loop block to manage both?
- Can the Beaver pick-up two planks together?

To solve challenge #11, we need to repeat the <pick-up-chop-until-small> process twice.

Ask the students - How can write a code that repeats for a defined number of times?

- Duplicating the code? It is possible, it is easy for 2 planks but what about 100 planks?
- When the planks are identical, we can use a repeat loop to execute the <pick-up-chop-until-correct-size> process
- How can we determine the number to use in the repeat loop? according to the number of identical planks

Challenge #9

Challenge #11

Part 1: 20 Minutes

Introduction cont.

Explain & Walk through	5 mins.

Open challenge #11 - and solve it together with the class.

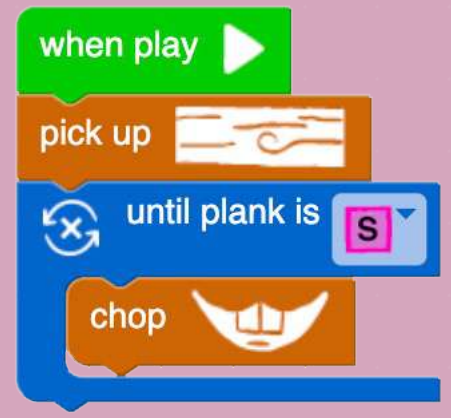
We recommend analyzing the solution bottom up - this will be a good example on how to approach complex problems that require division into simpler or known sub-problems. This is a common practice in programming and computer science.

We already know how to write the code to pick-up-chop a single plank.

How can we reiterate it twice? place the code within a repeat loop - This is called nested loops

- The outer loop is performed twice (according to the number of planks), in each iteration the entire code will be fully executed.
- The inner loop is responsible for chopping a single plank on each iteration of the outer loop
- For every plank of the outer loop, the inner loop is executed from start to end, and then that whole process repeats for the next plank in the outer loop, and so on
- The solution for a single plank is nested within the repeat loop as a whole.

Chopping a single plank



Chopping two planks

Part 2: 20 Minutes

Playtime

Playtime	**20 mins.**

If you see that the students understand the topic, then ask them to complete challenges 11-15 with three stars.

If not, we recommend discussing challenge #14 with the class - this challenge requires a combination of the solutions we wrote for challenges #11 & #12.

Ask the students to define the sub-problems -

- 2 large planks
- 4 small planks

We know how to solve each one them but how to combine the two solutions into one? Are they consecutive loops or nested loops?

Following the discussion on challenge #14 ask the students to complete challenges 13 and 15.

Use the classroom dashboard to keep track of students' achievements.
Use this time to walk around the class and help students who are struggling.

Part 3: 5 Minutes
Debriefing

Summary	5 mins.

Nested loops is a very powerful programming mechanism.

Ask students if they remember nested loops in the previous chapter, you can present challenge #31 where we learned how to nest a repeat loop within a repeat loop, or challenge #38 with a repeat loop within a counter loop (both challenges presented on the right of this slide).

The combination of nested loops is unlimited - you can nest any kind of loop inside any other kind of loop.

Lesson 4 – Do not Fall Off the Roof

To build a house we need walls and a roof. In this lesson we will learn new instructions that help the Beaver place and fix tiles on the roof.

Objectives

In this lesson, students will:

- Learn new blocks - place tile, move forward, turn left
- Learn how to position the beaver in the right direction with only a turn left block
- Complete challenges 16-18

U.S. Standards Addressed

CSTA-K12 Computer Science Standards	
1A-NI-04	1B-AP-08
1A-AP-08	1B-AP-10
1A-AP-10	1B-AP-11
1A-AP-11	1B-AP-13
1A-AP-12	1B-AP-15
1A-AP-14	1B-AP-16
1A-AP-15	1B-AP-17
1A-IC-18	

LESSON FOUR

Part 1: 25 Minutes

Introduction

PREPARE IN ADVANCE:

Set of smiley/stars/colored round stickers - around 5 stickers per student.

Play a game	**8 mins.**

The objective of the game is to make sure all the students can differ between their right and left hand and can tell between the right and left of a friend standing facing them.

As part of the game students will practice direction definition - left and right - for themselves and for a friend in order to make sure they understand that directions are perspective-based.

Set up the class

1. Divide the class into couples. The students in each couple will be numbered 1 and 2.
2. Arrange the class in two lines based on their numbers in a way that each couple will stand facing one another

Playing the game

1. On each round the teacher gives the instruction the couples need follow
2. On each round you will say a number - **1 or 2** and a body part like - **left hand**, **right leg**, **right ear**, **left shoe**, **right elbow**, etc.
3. All the students with the number you declared will put a sticker on their partner's body part you declared
4. The partners should keep quiet and not help or guide the player where to put the sticker
5. To run the game faster, you can start a timer (5-10 seconds) for each round
6. After each round check the couples, the couples who put the sticker in a wrong location will leave the game
7. The winners are the last couple to stay in the game

Part 1: 25 Minutes

Introduction cont.

Back to the Beaver - walk through	**7 mins.**

Open challenge #16 - Present the new scene and ask the students to describe the challenge - we need to write the code to help Beaver place roof tiles on the highlighted blocks.

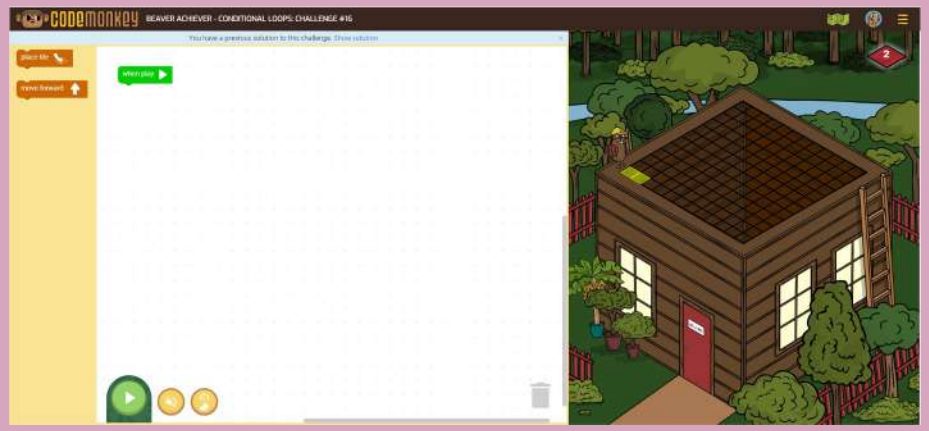
Components:

- Highlighted blocks on the roof - tiles to place
- Red numbered block on the top right corner - number of available tiles
- Initial direction of Beaver - we need to make sure the beaver is facing the right direction

To place a tile on a highlighted block the Beaver needs to stand on the previous tile, facing the highlighted block.

The Beaver can move on the roof after placing a tile - otherwise the Beaver will fall.

Challenge #16 is a guided challenge that introduces the place tile and move forward blocks.
Ask the students to follow the guided instructions and to complete the challenge.

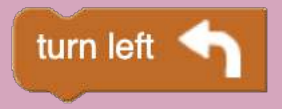


Part 1: 25 Minutes

Introduction cont.

Explain & walk through	**10 mins.**

Open challenge #17 - this challenge introduces a new block - turn left.



Ask the student to look at the Beaver in challenge 17, and ask:

1. From the starting direction, what will be the direction of the Beaver after executing a single turn left instruction?
2. From the starting direction, what will be the direction of the Beaver after executing two turn left instructions?
3. From the starting direction, what will be the direction of the Beaver after executing three turn left instructions?
4. From the starting direction, what will be the direction of the Beaver after executing four turn left instructions?

Let's emphasize:

- Q3 - three turn left equals turn right
- Q4 - four turn left equals zero turn left

Since in this game we only have a turn left block, the students should understand that they can turn to any direction using this block.

Part 2: 10 Minutes

Playtime

Playtime	**10 mins.**
All students should complete challenges 17-18 without falling of the roof and with three stars. Make sure the students pay attention to the Beaver's direction and the turns required to place all tiles. Use the classroom dashboard to keep track of students' achievements. Use this time to walk around the class and help students who are struggling.	

Part 3: 10 Minutes

Debriefing

Summary	**5 mins.**

This was a lesson with many new terms, ask the students:

1. Which new blocks have you used? move forward, place tile, turn left
2. How can we turn right if we can only turn left? Turn left three times

Sneak peek to next lesson	**5 mins**

1. Ask the students what is missing in terms of programming blocks? What would make it easier to write the code for challenge #18?
2. If they do not have an idea, ask them what would they do if they had 10 or 100 tiles to place?

There is a need for a loop to write better code - ask the students if they can use any of loops they have already used - repeat loop or until loop.

In the next lesson we will learn a new type of loop – a while loop. It is a conditional loop that is very similar to an until loop we used in previous challenges.

If you have time left in the lesson you can continue with the introduction part of the next lesson.

Lesson 5 – Make Sure to Cover it All

This lesson is focused on introducing the while loop and solving complex challenges with new combinations of consecutive and nested loops.

Objectives

In this lesson, students will:

- Understand the mechanism of a while loop
- Analyze the difference between until and while loops
- Use more combinations of consecutive loops
- Practice bottom-up problems solving
- Complete challenges 19-25

U.S. Standards Addressed

CSTA-K12 Computer Science Standards	
★ 1A-NI-04	★ 1B-AP-08
★ 1A-AP-08	★ 1B-AP-10
★ 1A-AP-10	★ 1B-AP-11
★ 1A-AP-11	★ 1B-AP-13
★ 1A-AP-12	★ 1B-AP-15
★ 1A-AP-14	★ 1B-AP-16
★ 1A-AP-15	★ 1B-AP-17
★ 1A-IC-18	

Part 1: 10 Minutes

Introduction

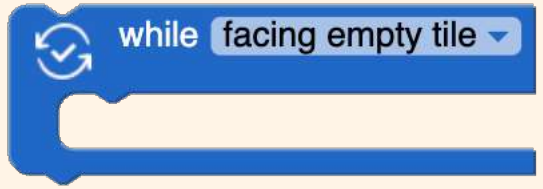
Reminder	**10 mins.**

Remind the students of the closing discussion from the previous lesson - where they raised the need for a loop.

Ask the students what is the loop condition in and until loop and how it affects the loop execution.

Reminder - The “until” loop contains a block of code that will repeat *until* “something happens”, or more accurately, until a specific condition is met. This condition is called a *loop condition*. The computer checks the condition at every repetition.
If the answer is false, the loop will keep going. It will stop once the answer is true.

Present challenge #19 - this is a guided challenge with a new block - **while loop**
while <facing empty tile> means - as long as the Beaver is still facing an empty till the loop will keep iterating.



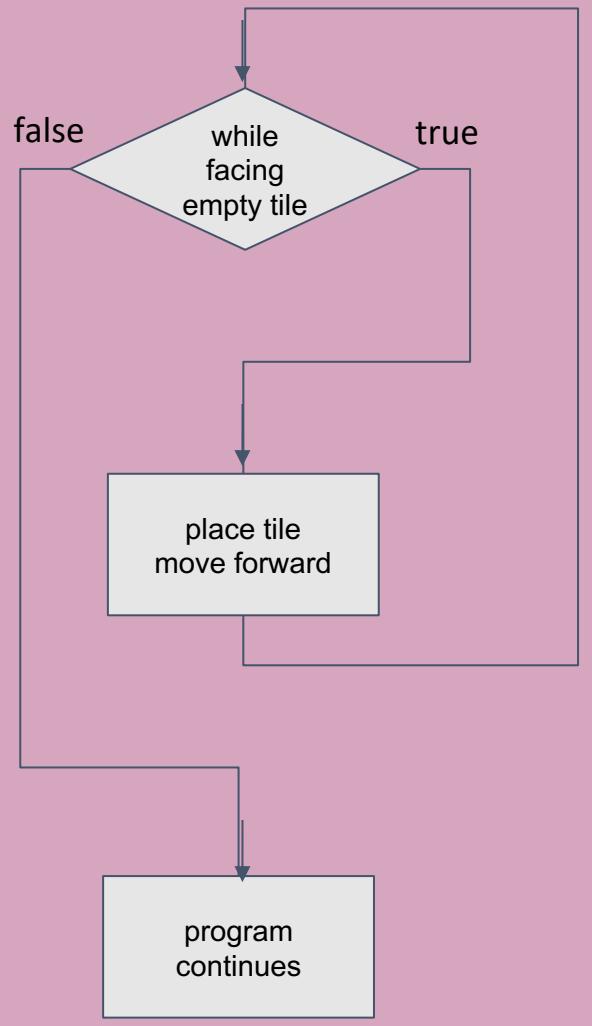
While loop definition - while loops are used when we do not know the number of iterations. We define a conditional statement that is checked before each new iteration and then executes the code within the loop as long as the the condition is true.
Once the conditional statement is checked and the answer is false the loop will stop.

Part 1

For the Teacher

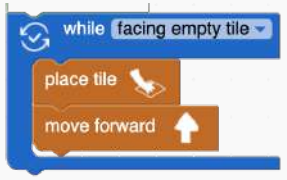
While Loop - The mechanism

The flowchart on the right illustrates the while loop mechanism - emphasize that the loop continues as long as the answer is true. Once the answer is false, the loop ends, and the program continues.



Part 1

For the Teacher

CATEGORY	REPEAT	UNTIL	WHILE
When to use	We will use the repeat loop when we know the number of iterations.	The until and while loops are used when the number of iterations can not be defined but we can create a conditional statement that controls when the loop stops.	
Initialization of loop	Define upper bound number of iterations.	Define a conditional statement that refers to a process/operation done within the loop. For example, the conditional statement is <plank is S> and the instruction is <chop> which reduces the size of the plank.	Define a conditional statement that refers to a process/operation done within the loop. For example, the conditional statement is <facing empty tile> and the instruction in the loop <place tile> <move forward> reduces the number of empty tiles on the route.
Loop termination	When reaching the upper bound.	When the condition statement becomes **true.**	When the condition statement becomes **false.**
Example			

Part 2: 30 Minutes

Playtime

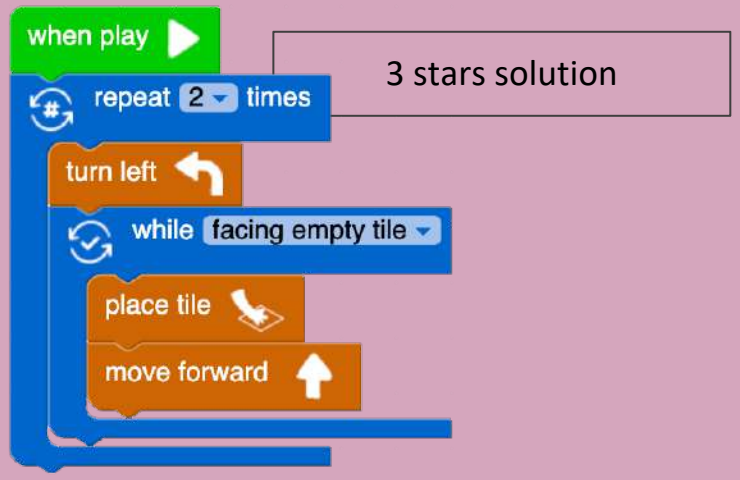
Playtime	**10 mins.**
Ask the students to complete challenges 19-20	
Open challenge #21 and present to the class - the situation here is a bit more complex. Ask the students to describe the complexity. It is a combination of challenges 19 and 20, which we already know how to solve. However, the beaver need to turn in the middle. How can we write a solution to fill all empty tiles when we need to turn in the middle? We can use 2 consecutive loops with turn left instruction between them. Ask the students to complete challenge #21 and then ask - What would you do if we had three empty routes that require turning in between? Do you have an idea for a mechanism that will simplify the code? This questions should lead the students to nested loop - remind them of challenges 11-15 (lesson 3). Which additional loop will help them simplify the code? Repeat loop	

Part 2: 30 Minutes

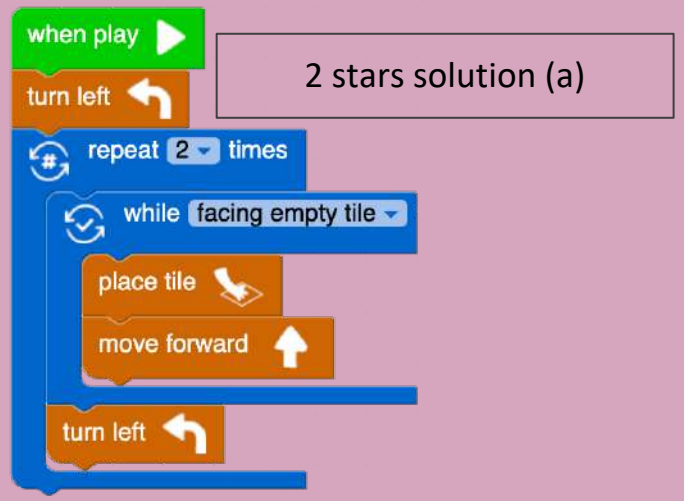
Playtime cont.

Playtime	**10 mins.**

Open challenge #22 - let the students write their code based on what we have learned so far. Check the students' solutions and choose 2-3 different solutions with different star-rating for a discussion with the class.
Characterize each of the 2 stars solutions - for example:
Solution (a) uses a repeat loop but does not exploit the full potential of the repetition using two turn left blocks before and in the repeat loop.
Solution (b) does not use the repeat loop but gives a clear view of the repeated process [turn left, while loop] which is then nested into the repeat loop in the 3 stars solution



3 stars solution



2 stars solution (a)

2 stars solution (b)

Part 2: 30 Minutes

Playtime cont.

Playtime	**10 mins.**
Challenges 23-25 are consecutive loop challenges; these challenges present the usage of different loops to perform a variety of instructions. In challenge #23 the students will use the until loop in order to make a turn right. In challenge #25 the students will use the repeat loop to move forward. Ask the students to complete challenges 23-25.	

Part 3: 5 Minutes

Debriefing

Summary	5 mins.

This was a lesson with many new terms and more complex challenges.
Ask the students:

1. What is the difference between an until and a while loop?
2. Which combination in consecutive loops and nested loops are possible?
3. Can we perform a turn right with an until loop? How?

Lesson 6 – Beware of the Fish

In this lesson the students will learn how to implement while loops with different conditional statements and practice more nested loops while helping the Beaver fixing broken tiles.

Objectives

In this lesson, students will:

- Deepen their understanding of the while loop conditional statement
- Practice bottom-up problem solving
- Use more combinations of nested loops
- Complete challenges 26-30

U.S. Standards Addressed

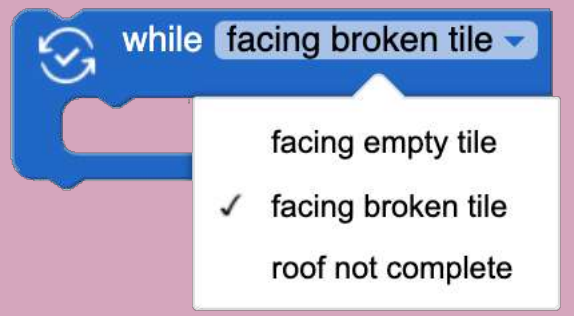
CSTA-K12 Computer Science Standards	
1A-NI-04	1B-AP-08
1A-AP-08	1B-AP-10
1A-AP-10	1B-AP-11
1A-AP-11	1B-AP-13
1A-AP-12	1B-AP-15
1A-AP-14	1B-AP-16
1A-AP-15	1B-AP-17
1A-IC-18	

Part 1: 15 Minutes

Introduction

PREPARE IN ADVANCE:
Basketball (or any other ball), a spacious class or an outside space

Game	**12 mins.**
Arrange the class in a circle and ask the students to throw the ball between them based on the following rules: 1. Keep playing while the ball is still in the air 2. Keep playing while the number of misses is lower than 10 Discussion: This is a real-life example for a while loop, both are correct each one of them is using a different conditional statement. It allows us to use while (and until) loops in different scenarios. Ask the class to think about additional conditional statements for a ball game.	
Walk through	**3 mins.**
Present challenge #26- this is a challenge with a new condition for the **while loop** while <facing broken tile> means - as long as the Beaver is still standing in front on a broken tile, that needs to be fixed, the loop will keep iterating.	



Part 2: 25 Minutes
Playtime

Playtime	**15 mins.**
Ask the students to complete challenges 26-28	
Focus on the bottom-up process - dividing the bigger problem into small problems and combine them together to solve the bigger problem.	
Explain	**5 mins.**
Open challenge #29 - draw on the board a matrix of 3X4 (like the empty tiles in the challenge). Ask the students what are the possible progress paths within the matrix - encourage them to think about the path that can be the core of the small-problem that repeats itself. Draw the path on the board and outline a division into repeated small problems.	
Playtime	**5 mins.**
Ask the students to complete challenges 29-30	

Part 3: 5 Minutes

Debriefing

Summary	5 mins.
This lesson was about practicing everything we have learned so far. Ask the students, which of the 5 challenges was the easiest? the most challenging? why? When practicing nested loops, we will see the beauty of diverse way of thinking and problem solving. Encourage the students to share their 2-star solutions - these solutions hide many topics for discussion - how can the students improve their solutions?	

Lesson 7 – The Loop Master

Congratulations – you are about to complete another chapter of Beaver Achiever - and it is time to make sure all the students understand this topic.

Objectives

In this lesson, students will:

- Practice bottom-up problem solving
- Use more combinations of nested loops with repeat, while and until loops
- Complete challenges 31-35

U.S. Standards Addressed

CSTA-K12 Computer Science Standards	
★ 1A-NI-04	★ 1B-AP-08
★ 1A-AP-08	★ 1B-AP-10
★ 1A-AP-10	★ 1B-AP-11
★ 1A-AP-11	★ 1B-AP-13
★ 1A-AP-12	★ 1B-AP-15
★ 1A-AP-14	★ 1B-AP-16
★ 1A-AP-15	★ 1B-AP-17
★ 1A-IC-18	

Part 1: 10 Minutes

Introduction

Discussion	**10 mins.**

Ask your students “What are the different loops we learned in this course?”

Ask for three students volunteers, and ask each of them to explain one of the three loops:

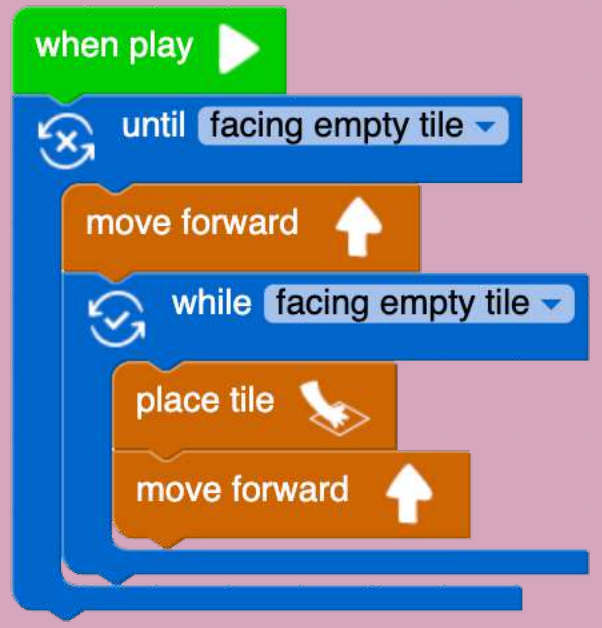
- **Repeat loop** - a sequence of instructions that repeats a specified number of times.
- **Until loop** - contains a block of code that will continue to run until a specific condition is not met (false)
- **While loop** - contains a block of code that will continue to run as long as a specific condition is met (true)

Ask the students what is the difference between consecutive loops and nested loops and give an example from life (or from Beaver).

Part 2: 30 Minutes

Playtime

Analysis	**10 mins.**
Present challenge #33 with the following solution (on the right) and ask the students to tell what will be the result of running the code. Practicing code dry-run is a useful tool for teaching students to find their own mistakes/bugs and analyze the objectives of a code.	
Playtime	**20 mins.**
Ask the students to complete challenges 31-35 with at least 2 stars. Split the group into couples and ask the students to check one another's code before running the game.	



Part 3: 5 Minutes

Debriefing

Summary	**5 mins.**

Congratulations! You have completed another chapter of Beaver Achiever and you are now the masters of loops.

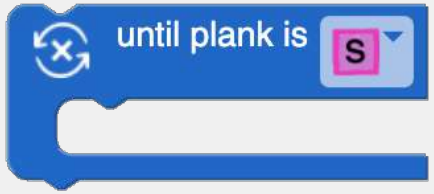
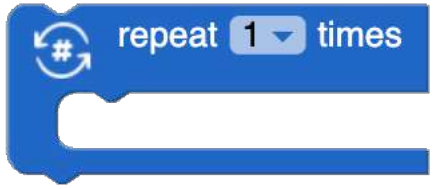
Loops are a fundamental programming structure that will be relevant for any programming language the students will use in the future.

Our daily routines are based on repeated execution. Ask the students to describe a daily or a human process using as many loops as possible.

Quiz

Quiz	
Now that your students completed the course, you can assign a quiz for them to take. The quiz includes 5 challenges that will test their knowledge on: • Until loops • While loops • Nested loops You can assign quizzes to your class from the Quizzes tab on your teacher dashboard.	

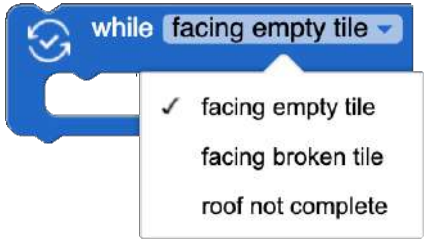
Reference Card

Block	Description
	Pick up - makes the beaver pick up an uncut piece of plank to prepare to cut it to the right size.
	Chop - makes the beaver chop the piece of plank it is holding, to make it shorter. This block is responsible for a single chop.
	Until loop – makes the beaver repeat all the blocks placed inside this block until the plank's length is the length chosen in the dropdown.
	Repeat Loop - makes the beaver repeat all the blocks placed inside the loop block for the number of times chosen in the dropdown.

Reference Card

Block	Description
	Place tile - makes the beaver place a tile in front of it. Placing a tile on a not empty place will result in an error.
	Move forward - makes the beaver move a single step forward to the next tile. If the next tile is empty, the beaver will fall.
	Turn left - makes the beaver turn left while keep standing on the same tile.
	Fix tile - makes the beaver fix a broken tile in front of it. Trying to fix a tile that is not broken, or an empty place will result in an error.

Reference Card

Block	Description
	Until loop – makes the beaver repeat all the blocks placed inside this block until the beaver is facing the type of tile chosen in the dropdown (empty or broken).
	While loop – makes the beaver repeat all the blocks placed inside this block while the condition chosen from the dropdown list is true.

Great Job!