Lesson Plan

Lesson 5 - Loops – repeat

Before you start – If you haven't already, please read CoderBunnyz rulebook page 1-8 for introduction to playing the game and also read rulebook page 11, 12 for introduction to advance cards.

Lesson Overview

Students will do a concept introduction worksheet. Then they will play an advance game level with loops and will be introduced to the repeat and loop concepts in coding. Finally, they will write their code as an algorithm.

Lesson Objective

- First students will do an introduction worksheet to introduce the concept of loops and repeat. In this sheet they will convert a series of multiple actions into repeat loops
- Then they will play a level of CoderBunnyz using repeat cards.
- Finally, they will count the number of code cards used as instructions and write/draw the sequence of cards taken. This is called algorithm writing.

Materials needed

- Repeat loops worksheet (on the next page), game, algorithm sheet 5.1, pencil

Getting Started

- Instructor will explain worksheet 5. Players will do the exercise.
- After the worksheet is complete, arrange the board and cards based on Rule book level 2.1 (see page 12), explain the cards, movements, and rules to the student. Choose a destination and get ready to start the game.

Activity

- Play the Rulebook level 2.1 of CoderBunnyz. Your goal is to program your bunny to eat their colored carrot, move around the maze, and reach the destination. Continue till all players reach the destinations.
- Then each player will review their code cards. That's the sequence of code. Count the number of cards used to reach the destination. Also, write the algorithm of the game played on sheet 5.1

Fun Fact

The first computer was actually a loom called the Jacquard loom, an automated, mechanical loom, which didn't use any electricity.

Read aloud

5. Loops

An efficient way of repeating code to make it execute over and over again.

Read aloud

Creating repeat loops

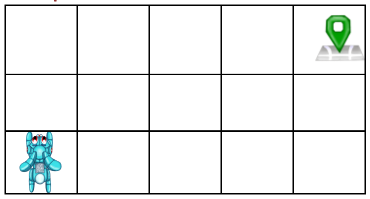
If someone asks you to do this -

Go Forward, Go Forward, Go Forward, Go Forward

You could also say

Go Forward – 4 times OR Repeat (4) – Go Forward

Circle the paths that could be converted to repeat loops



⬆ ⬆ ↱ ⬆ ⬆ ⬆ ⬆ ⬆

Here's an example of converting simple code to repeats

⬆ ⬆ = Repeat(2) ⬆

Practice Exercise

Convert this sequence to repeat by filling the blank

⬆ ⬆ ↱ ⬆ ⬆ ⬆ ⬆

Repeat (__) ⬆ ↱ Repeat (__) ⬆

Think of 2 situations
in your daily life
where you could use repeat

1)

2)

Practice Exercise

5.1 Loops

Look at your code : Count how many code-cards you use in your coding today:

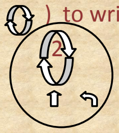
_____	MoveForward ( );	⇧
_____	TurnLeft ( );	↰
_____	TurnRight ( );	↱
_____	Jump ( );	↷
_____	Repeat (2);	2
_____	Repeat (3);	3
_____	Repeat (4);	4

Add these three, You use _____ Total_Code_Cards( );

---ADVANCE ---

Write your Code for today's game. (Use back of your page to continue)

Hint: Use arrows (⇧, ↱, ↰ , ↷ , ⟳) to write your moves or use codewords (MoveForward () ; TurnLeft(1);..) or both! **How to use repeat?**



Is same as Repeat (2) { MoveForward( ); TurnLeft ( ); }

Loop and Repeat in real programming languages?

In computer programming, a loop is a sequence of instructions that is continually repeated until a certain condition is reached or accomplished. Typically, you ask a computer to do a certain process, such as getting an item from the data and changing it. When it is done, a condition is checked, such as whether the data has reached a certain number. If it hasn't, the next instruction in the sequence is to return to the first instruction and repeat the sequence. If the condition has been reached, the code proceeds onto the next sequential instruction, or branches outside the loop to continue the code. A loop is a fundamental programming idea that is commonly used in writing programs.

A code example in many programming languages using conditional -

```
   count = 0;
1. while (count < 2){
2.  moveforward ( );
3.  count = count+1; }
```

If the condition in line 1 is TRUE (✓) then line 2,3 will be executed and if condition in line 1 is FALSE (✗) then loop will exit out.

count	Condition(Line 1)	Action
0	✓ (TRUE)	Moveforward( ); count =1 (Line 2,3)
1	✓ (TRUE)	Moveforward( ); count =2 (Line 2,3)
2	✗ (FALSE	Exit loop