

# Lesson Plan

- Lesson 1: Learning Sequence
- Lesson 2: Learning Branch, Jump (), goto ()
- Lesson 3: Making decisions, Conditional – if ()  
then else ()
- Lesson 4: Fixing Errors, Bug and Debugging
- Lesson 5: Looping with repeat, bounded loops
- Lesson 6: Understanding Functions

## *Bonus lessons*

- Lesson A: Introducing operations, greater, less than. Boolean - TRUE, FALSE
- Lesson B: Introducing Variable, string and numeric
- Lesson C: Nested repeat – Loop in Loop

## **Lesson 2: Jump and Branches**

**Before you start** – If you haven't already, please read CoderBunnyz Rulebook page 1-8 to be introduced to playing the game.

### **Lesson Overview**

Students will do an introductory worksheet. Then they will play the game level with puddles and be introduced to the jump and branch 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 jump and branches.
- Then they will play a level of CoderBunnyz, and arrange their code cards as a sequence of steps with jump as needed.
- 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**

- Jump worksheet(on the next page), game, algorithm sheet 2.1 (page 16), pencil

### **Getting Started**

- Instructor will explain worksheet 2. Players will do the exercise.
- After the worksheet is complete, arrange the game (see Rule Book page 9, Level 1.2), explain the cards, movements, and rules. Choose the destination and get ready to start the game.

### **Getting Started**

- Instructor will explain worksheet 1. Players will do the exercise.
- After the worksheet is complete, arrange the game (see Rule Book page 9, Level 1.1), explain the cards, movements, and rules. Choose the destination and get ready to start the game.

## Activity

- Play level 1.2 of CoderBunnyz to program your bunny to eat their colored carrot jump over the puddle and reach the destination. Continue till all players reach the destination.
- Then each player will review their code cards. That's the sequence of code (with jump ) they will write on their algorithm sheet.
- Count the number of cards used to reach the destination and write those on the sheet 2.1. Also write the algorithm of the game played.



### Fun Fact

The first game was created in 1961. Fun facts are that it didn't earn any money.

# 2.Jump( )

Jump and branch cause switching from normal path to different path

Read aloud



What do you do when you find a puddle in your way? You don't go in puddle and get wet but jump



Jump!!

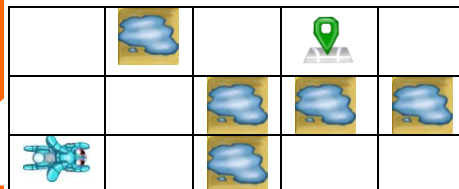
Read aloud

An example sequence showing jump



Read aloud

Your turn to help bunny ; Use



Practice Exercise

Create another ways from to .  
Which one is better? Why?

-----

Practice Exercise





## 2.1 Jump



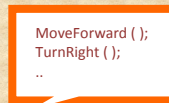
Look at your code : Count how many code-cards you use in your coding today:

\_\_\_\_\_ MoveForward ( );

\_\_\_\_\_ TurnLeft ( );

\_\_\_\_\_ TurnRight ( );

\_\_\_\_\_ Jump ( );

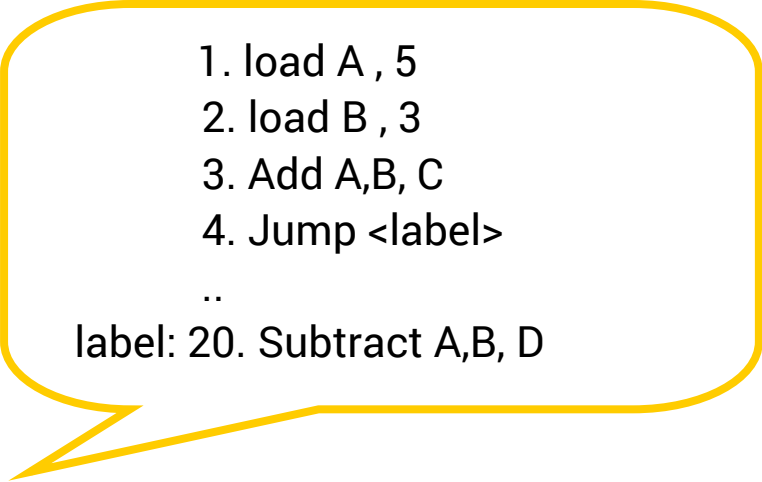


Using the symbol or text Write your Code for the game played!!

## Branch and Jump in real programming languages?

The “branch and jump” is used by assembly programming languages to move the flow from one section of the code to other. Assembly is a low level language that computers can understand directly. All language codes are eventually converted in that language. Java, Python, C, C++ languages implement Jump and Branch using conditionals (Chapter 3) and loops (Chapter 5).

A code example in Assembly to do addition & jump & do subtraction.



```
1. load A , 5
2. load B , 3
3. Add A,B, C
4. Jump <label>
..
label: 20. Subtract A,B, D
```

Only the line with ✓ will be valid but not the ones with ✗, because of jump

Code Line	Flow	Comment
1	✓	Load A with 5
2	✓	Load B with 3
3	✓	Add A, B to save it to C. C is now 5+3 = 8
4	✓	Now addition is done, jump to a place with <label>
5	✓	Nothing
..	✗	More code here
20	✓	Line 20 is where Jump move the code, now do subtraction here