ncLab
More Than Coding

KAREL THE ROBOT

LESSON PLAN FOR HOUR OF CODE

OVERVIEW

A determined robot named Karel runs through jungles, swims in the sea and climbs up icy mountains, collecting useful things like candies and spiders while trying not to crash into walls. Karel is a robot after all and needs a human to guide him, a human willing to write simple programs so Karel knows which way to go, what to pick up and where to put it.

Karel the Robot teaches learners how to write programs using the Karel programming language, which is based on Python. Starting with simple keystrokes in Game 1, learners move on to typing lines of code to create and run programs. They learn basic commands, the repeat loop, if-else conditions, the while loop, and finally user-defined commands.

This Hour of Code lesson consists of 18 mini-games. Most of the coding has already been done; learners must fill in a few lines to complete each program, and then test it by running Karel the Robot through a maze. Successful completion of each game unlocks the next one.

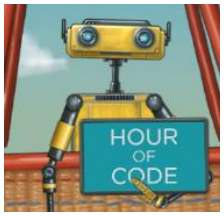
The lesson is designed for students ages 9 to 15, but can be enjoyed by anyone who wants to get a taste of programming.

OBJECTIVES

Students will learn to:

- Control movement using GO, LEFT, and RIGHT commands.
- Pick up and place objects using GET and PUT commands.
- Write lines of code using GO, LEFT, RIGHT, GET and PUT.
- Change repeated patterns of commands into REPEAT loops.
- Handle unknown features in the maze using IF-ELSE conditions.
- Control more random situations using the power of WHILE loops.
- Simplify their programs by using DEF to turn a series of commands into one command.

MATERIALS AND PREP – STUDENTS SHOULD BE LOGGED ON TO HOUR OF CODE AND HAVE KAREL THE ROBOT SELECTED BEFORE BEGINNING THE LESSON



- **Personal computers or tablets, with Internet access:** one per student. Both PC and MAC platforms are supported. Tablets can also be used. Preferred browsers are Chrome or Firefox.
- **Projector or Smartboard** (optional but recommended) attached to a computer for demonstration or modeling
- **Hour of Code Registration:** Follow instructions at http://www.hourofcode.com to register your class for the Hour of Code event.
- **YouTube videos:** some schools block YouTube, so the demonstration videos embedded in the games may need to be unblocked by an administrator to make them available to students.
- **Printable reminder cards** (see Introduction and Celebration sections) **and "Cheat Sheets" (code vocabulary definitions)** can be given to each student.
- **Assessment:** students can take notes as they make their way through the games, and use the notes to complete an exit ticket. A printable note-taker and exit ticket are attached.
- **Certificates:** Print certificates for students who successfully complete all 18 games.

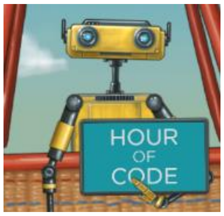
TEACHING TIPS

- Spend an hour or two ahead of time trying out the games yourself.
- Even simple maneuvers and commands may seem difficult at first, and students may need a suggestion or two to progress.
  - LEFT and RIGHT are always from the robot's point of view.
  - Watch for indentation errors: the lines following a REPEAT or WHILE line will be indented two spaces. The program has color-coded lines to help students line up their typing.
- As much as possible, allow the students the freedom to figure out what to do on their own.
- Decide ahead of time whether your students will be working alone, in pairs or as teams, or if certain students would benefit with being partnered with a mentor student. Generally speaking, younger or less experienced students benefit from discussion and partnership, whereas older or more experienced students prefer working on their own with minimal discussion.

VOCABULARY – CODING TERMINOLOGY

- The teaching videos included at the beginning of each new concept explain coding terminology.
- **Code** and **Program** are defined in "Beginning the Lesson".
- A printable "cheat sheet" is attached to this lesson. It explains the **basic commands**, **loops**, **conditions**, **logical expressions** and **custom commands**.
- Since students are just filling in a few lines of code or less, they will not need a deep understanding of these terms before they begin.
- Encourage students to watch what Karel does when they run the programs. His actions are based on the code and will help students learn what the code does.

INTRODUCING THE LESSON (5 MINUTES)



Today, you will be learning how to write code for Karel the Robot by playing some games.

> **Code** is a set of words and numbers that send instructions to the computer. For example, when you type the word `go`, Karel will move one square forward.
>
> A **program** is made up of lines of **code** that give a set of instructions. Once you learn some shortcuts, you will be surprised to see how few lines of code are needed. In a typical maze, Karel must pick up objects, put them in containers, and get to the home square without running into any walls. It is fun to watch the program run through its lines of code while Karel is following the instructions on the screen.

There are eighteen games in all. With each game, you will be learning a new idea in coding. Most of the program is already written: you just have to fill in the blank lines with the missing code.

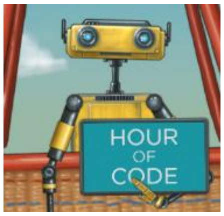
Watch the videos at the beginning of each section to learn what code skills are being introduced. You can also use your "What is Karel Doing?" cheat sheet.

Note takers: If you have chosen to use the note takers, hand them out at this time. Students can either jot down what they learned in each game as they go, or take notes after the whole session as a review.

Remind them of what to do if they get stuck. You may want to print out this reminder card for each student:

If you get stuck:

1. **Reread the task.**
2. **Run Karel through the maze mentally. Do you see any patterns?**
3. **Run the program line-by-line (the black arrow instead of the green arrow).**
4. **Read the error message.**
5. **Check the list of words that are required in the program. Have you used them all? Did you use a word that is not on the list?**
6. **Check for spacing, spelling and indents.**
7. **Check with a neighbor.**
8. **Check with another neighbor.**
9. **Ask the teacher for help.**



You must successfully complete each game to unlock the next one. Good luck, and let's get coding!

STUDENT ACTIVITY (45 MINUTES)

Students will play the games, progressing at their own pace. Students may be working as individuals, partners or teams, as determined by the teacher.

Encourage and cheer problem-solving skills and progress; help as needed.

Let students know when they have about 5 minutes left.

CELEBRATE! (5 MINUTES)

"How far did you get today?

Today, we participated in the Hour of Code, along with millions of other coders around the world.

Computers and real robots need humans to write instructions for them, whether they are used for fun or for work. You can write code to create art, build a car, test medicines, or design a video game. Coding is used everywhere."

Here is another printable card that can be given to each student.

For more free Karel the Robot resources, visit http://www.nclab.com/resources/

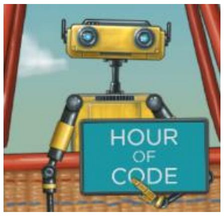
You will find:

1. **A free Karel Playground that lets you build your own mazes and write programs.**
2. **If you create a free user account in NCLab, you will be able to access a free Karel App that lets you create your own games, save them in your NCLab account, and share on the web so that your friends can play them.**
3. **NCLab provides self-paced courses Karel 1, Karel 2 and Karel 3 that will teach you how to think like a computer scientist, and write programs - using Karel the Robot of course.**
4. **After finishing Karel, NCLab has a follow-up intermediate programming course Turtle Tina, where you will learn Python syntax while drawing beautiful art and even exporting it for 3D printing.**
5. **After finishing Turtle Tina, you will be ready for the advanced Python programming course. Python is a leading programming language of modern engineering and science, and a real job skill.**
6. **NCLab has a self-paced 3D modeling course that will teach you how to design your own 3D models using simple Python code. You will be able to export your 3D models as STL files and print on any 3D printer.**
7. **There are lots of other tools in NCLab to help you with science and math.**

ASSESSMENT (5 MINUTES)

Give each student an Exit Ticket to write down a few sentences about what they learned, what was easy, what was challenging, what they could imagine doing with their coding skills.

EXTENDED LEARNING



Karel 1 and Karel 2 teach students how to code using the maze games. By the end of Karel 2, students will be fully-fledged programmers. In the Hour of Code set of games, students just add a few lines to complete a program. In the full courses, students write their own programs from start to finish.

A set of lesson plans includes student journals, game assessments and detailed instructions on how to set up a class. Please visit http://www.nclab.com for more information on the Karel courses and many other resources for teachers and students.

Students may also want to explore other activities on Hour of Code.

DIFFERENTIATION

Support:

Beginning students may require more modeling before launching into the games.

- Model Game 1 (mouse controls), then have the students do Game 1,2, and 3.
- If students have difficulty with turning the robot left and right, try modeling or practicing with real students on squares (square tiles on a tiled floor, or squares marked off with tape).
- Model Game 4, which is the first game that requires coding. Point out features on the screen. Run the program and have students compare Karel's actions to the line of code.

If many students are struggling with a game, go over the cheat sheet hint or watch the video together for that skill, and discuss it.

For students who finish the games early:

- Have students review the games and fill in the note taker.
- Encourage them to explore other games on Hour of Code.
- If you have accounts with NCLab set up for your students, they can log on and try to build their own maze and game using Karel Playground.

STANDARDS

Common Core Math Standards: Students look for patterns, use tools appropriately, solve problems, and use logical reasoning. (Standards for Mathematical Practices)

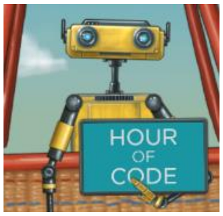
Next Generation Science Standards: Students learn engineering skills that can be used to build their own programs (for example, creating a game with Karel Playground)

English Language Arts: Students discuss solutions, and write reflections on what they have learned.

RESOURCES

On the following pages: student note takers and exit tickets, a certificate of completion, and a "cheat sheet".

KAREL THE ROBOT: HOUR OF CODE

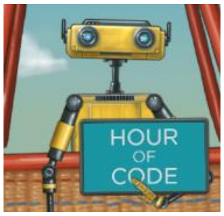


NAME: _________________________________

WRITE BRIEF NOTES ON EACH GAME. WHAT DID YOU LEARN? WHAT WAS CHALLENGING?

GAME 1	**GAME 2**
GAME 3	**GAME 4**
GAME 5	**GAME 6**
GAME 7	**GAME 8**
GAME 9	**GAME 10**
GAME 11	**GAME 12**
GAME 13	**GAME 14**
GME 15	**GAME 16**
GAME 17	**GAME 18**

KAREL THE ROBOT: HOUR OF CODE



NAME: ________________________________

EXIT TICKET

What did you enjoy the most about Karel the Robot?

What parts of Karel the Robot were challenging?

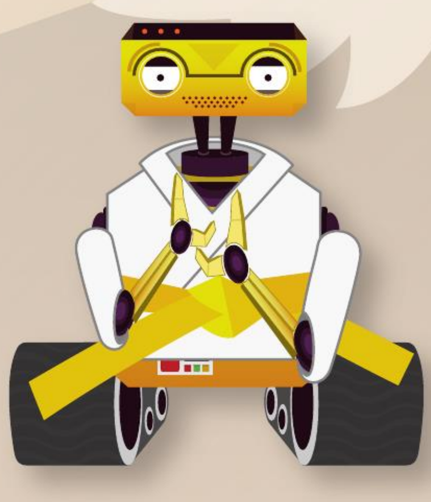
What would you like to create with code?

What would you like to learn next?

ncLab
More Than Coding

CERTIFICATE OF COMPLETION

This is to certify that



successfully completed an Hour of
Code by solving 18 puzzles for Karel the Robot,
demonstrating awesome computational thinking
and problem solving skills

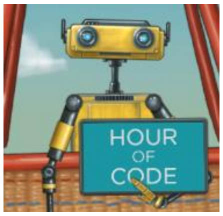
Place, date

Official signature

HOUR OF CODE

Visit NCLab.com to learn Karel programming, Turtle programming, Python programming, 3D modeling and more!

What is Karel Doing?



GAMES 4 -18

BASIC COMMANDS

`go`: Make one step forward.

`left`: Turn 90 degrees left.

`right`: Turn 90 degrees right.

`get`: Collect an object from the ground.

`put`: Put an object on the ground

GAMES 7-18

LOOPS

There are two types of loops:

The `Repeat` loop is used when the number of repetitions is known in advance:

```
repeat 5
  go
```

The `while` loop should be used when we do **not** know in advance how many repetitions will be needed:

```
while not wall
  go
```

Both loops can contain basic commands, conditions, other loops, and custom commands.

GAMES 16-18

CUSTOM COMMANDS

Custom commands can be defined for "smaller tasks" that are done more than once in the program:

```
def turnback
  repeat 2
    left
```

Now you can use `turnback` as a command!

GAMES 10 – 18

CONDITIONS

The `if – else` conditions help the robot check his surroundings and make decisions. Notice the two-space indentation on the command lines.

The `else` branch can be omitted if not needed.

Example:
```
if wall
  left
  left
else
  go
```

GAMES 13-18

LOGICAL EXPRESSIONS

Keyword `not` means negation. It returns `True` if the operand is `False` and vice versa.

Example:

```
while not wall
  go
```

Keyword `and` returns `True` if both statements are `True`, else it returns `False`.

Example:

```
while not wall and not home
  go
```

Keyword `or` returns `True` if at least one of the statements is `True`, otherwise it returns `False`.

Example:

```
if gem or nugget
  get
```