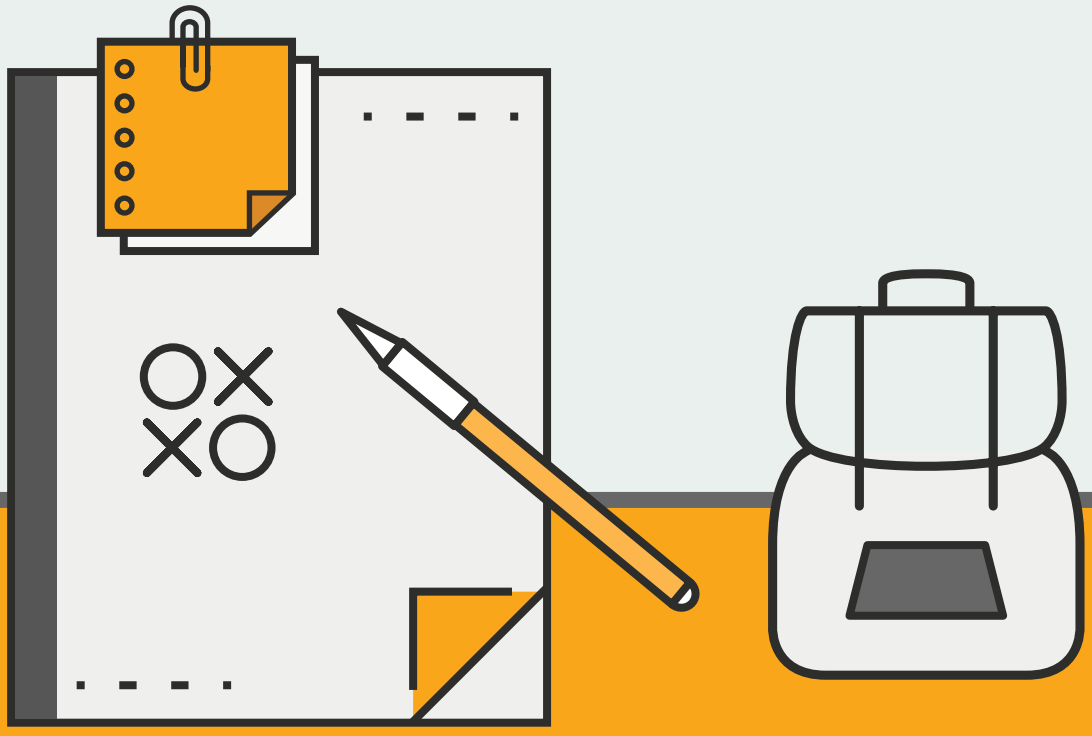
teachers learning code



DIGITAL TOOLBOX

Unplugged Activities

An educational program by

CANADA LEARNING CODE

Table of Contents

For more great lesson plans, check out our website:
canadalearningcode.ca/lesson-plans

Going Unplugged

What does "Unplugged" mean?

Unplugged activities are lessons that teach computational thinking skills without the requirement of computers or any expensive equipment. These activities are designed to be run easily, without much preparation or supplies. They are highly collaborative and can be modified to complement a variety of subjects and grade levels. Unplugged activities range from 10 minutes to an hour, designed to engage learners through exploration and play.

Making Coding Accessible

Why Unplugged?

While computational thinking skills are highly valued, many educators have the perception that these skills cannot be developed separate from actual computational devices. However, this is not true! Learners without access to computers are just as capable of learning programming concepts and practices. Unplugged activities are one way to facilitate this learning, and make computational thinking skills accessible to everyone.

Unplugged activities are a great way to subtly slip programming concepts and practices into seemingly Non-coding-related games. Learners are often unaware that they are learning about debugging as they stack a cup tower, or functions as they sing a song. Often, It isn't until the facilitator explains what they actually learned that the "aha" moment takes it place.

Transitioning to Devices

Unplugged activities can be used in conjunction with tech-dependant activities or as a stand-alone. However you are leveraging an activity, it is always important to explain the "why" behind each activity. What are we actually learning here? How does this relate to coding? Articulating the learning outcomes of each activity helps reinforce important concepts, and helps learners make direct connections between the activities and how technology really works. Once devices are introduced, your learners should be ready to recognize the same concepts as they transition to computer-mediated learning.

LESSON PLAN:

Do the Robot

DESCRIPTION

A basic "unplugged" challenge to get learners thinking about simple instructions and sequences, or in coding terms, creating algorithms.

- **Grade:** Everyone
- **Length:** 15 minutes
- **Subjects:** Physical Education, Social Studies
- **Coding Concepts:** Algorithms, Sequence

PREP WORK

Think of sample activities for pairs to complete: i.e. tying a shoe, opening a door, doing the macarena.

The Lesson

- In groups of two, assign one learner to be the programmer and one to be the robot. Assign each pair an activity like tying a shoe or opening a door.
- These activities could be pulled from a hat, assigned, or chosen by the learners.
- Ask the programmer to explain to their partner (the robot) how to perform the steps needed to complete their activity using words only!
- Switch pairs.
- After the activity, use this as an opportunity to talk about the importance of simple, clear instructions and sequences.

Extensions

- Congratulate learners for creating their first algorithms!
- Have learners perform their coding sequence in front of the class.
- Which was harder being the programmer or the robot?
- Have the class guess what the pair may have missed as they were explaining the steps.
- Choose some code to review as a group — do learners recognize any patterns?
- Any ways they could simplify their algorithms?

Assessment

Learning Outcomes

- We used simple, clear instructions to perform a task.
- We created an algorithm (a step-by-step set of operations to be performed to help solve a problem).
- We learned the importance of sequence in coding — Computers read and perform commands in order from top to bottom (order matters!)

Suggested Lesson

Basketbots: Do the Robot meets basketball!

https://www.canadalearningcode.ca/lessons/basketbots/

DESCRIPTION

This activity reinforces basic computational thinking through using code to build cup towers. Learners will work in pairs as 'robots' and 'programmers' to write out algorithms, or step-by-step instructions, for each other, then debug and problem-solve their code together.

- **Grade:** Everyone
- **Length:** 30 minutes
- **Subjects:** Mathematics, Language Arts
- **Coding Concepts:** Algorithms, Debugging, Modularizing

PREP WORK

Materials

- Plastic cups (10 per pair)
- Paper and pencils
- Print/scan the Symbol Key and Cup Stack sheets for the main activity. Available at **http://bit.ly/speed-stacker-thinkersmith**. (1 per table)
- Cut out cup sheets ahead of time

THE LESSON

Run through this example as a class: **http://bit.ly/speed-stacker-eg**

Pair up learners. One person will begin as the **programmer**, and one as the **robot**.

Programmer instructions

1. Choose a Cup Stack sheet from the pile (don't show the robot!)
2. Use the symbols from the Symbol Key to write instructions for the robot to follow, in order to build this cup tower.
3. Give the robot your instructions.
4. After the robot has finished, debug your code (catch any problems and try to solve them). Write new instructions and try again, if needed.

Robot instructions

1. Take the cups and practice stacking them away from the programmer.
2. When the programmer calls you over, use their instructions to build a tower. Use the Symbol Key to decode their step-by-step instructions.
3. When you are finished, check to see if the tower looks the same as the Cup Tower sheet. If it does not, work with the programmer to fix the instructions.
4. Try again! Keep debugging until the instructions are correct.

After running through once, have learners switch with their partners.

Extensions

- Have pairs sit back-to-back, with the programmers facing you and the robots looking away (no peeking towards the front!)
- Show all programmers the same tower at once and have them race to write down an algorithm for their partner to build.
- When they are finished, they can pass their written algorithm to their robot partner (robots must still face away — no peeking!)
- Robots with an algorithm can begin building their cup tower. Programmers may watch and take away the paper at any time to "debug" and adjust their code before returning it to their robot.
- Whichever pair successfully builds the tower first, wins!

Assessment

Learning Outcomes

- We created and decoded algorithms (our cup stacking instructions).
- We learned the importance of sequence and clear instructions in coding.
- We worked with our partners to debug (or fix) our instructions.
- We practicing modularizing by breaking down a larger task into smaller parts.

Assessment Ideas

Have pairs submit their final instructions with a written reflection outlining their debugging process.

Suggested Lesson

Unplugged intro to Coding for Kindies: **https://www.canadalearningcode.ca/lessons/unplugged-intro-to-coding-for-kindies/**

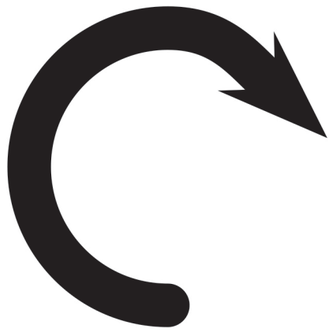
Speed Stacker: Symbol Key + Cup Stack Sheets **http://bit.ly/speed-stacker-thinkersmith**

Pick Up Cup

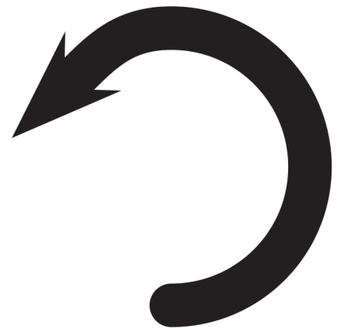
Put Down Cup

Step Forward

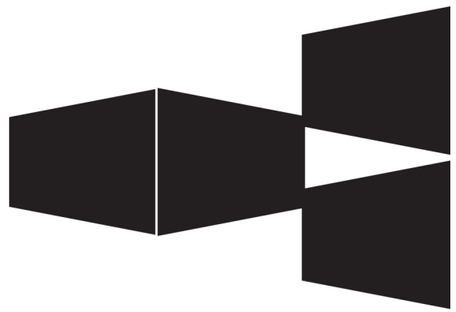
Step Backward

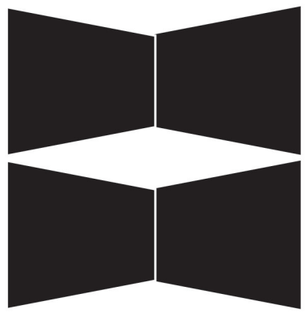


Turn Cup Right 90°



Turn Cup Left 90°





LESSON PLAN:

Pixel Programming

DESCRIPTION

Learners will use post-its to program pixel art. Then they will be able to write their very own pixel programs.

Grade: Grades 1-3, 4-6

Length: 20 minutes

Subjects: Mathematics, The Arts

Coding Concepts: Algorithms, Debugging, Conditionals

PREP WORK

- Display the Pixel Programming instructions where everyone can see
- Print/scan the Programs + Solutions for the main activity. Available at **http://bit.ly/pixel-programming** (1 copy per group)
- Post-it notes, two different colours (10 per pair)
- Paper and pencils

The Lesson

- Pair up learners.
- Go through the instructions (below) and run through Program 1 as an example together. Begin by choosing which colour you will start with.
- Give one partner a program sheet and the other the solution.
- Have the first partner try to follow a program by placing sticky notes on a table or wall (no peeking at the solution!)
- Have the other partner check the solution to see if it's correct.
- If it is incorrect, "debug" or try to solve it together.
- Have partners switch roles!
- If time, have pairs create new pixel programs for each other.

INSTRUCTIONS

1. We are programming left → right, top → bottom (like reading!)
2. Choose 2 different coloured sticky notes
3. The number is the amount of sticky notes to place down in a row
4. The comma means switch colours

Extensions

- **Work backwards:** Have one partner use post-its to create pixel art, then have their partner write the code for their art. Share the instructions with another pair to see if they can replicate it.
- **Spell with pixels:** Have learners try to spell out their names using the post-it notes, or create a pixel program for the class to solve that spells out a secret message. Use letters that can be created within the 3×3 grid.

Assessment

Learning Outcomes

- We followed simple, clear instructions to perform a task
- We created and decoded algorithms (when writing instructions)
- We made decisions like computers, using conditionals (IF there is a comma, then switch colours)
- We worked with our partners to debug (or fix) our instructions

Assessment Ideas

Have learners make their own pixel art in pairs, then submit the instructions used to create it.

Make it into a math game. Have learners count up from 1 and switch colours every time they count a multiple of 4 (e.g.), moving to a new row after every 10 post-its. This may result in some interesting-looking patterns.

Suggested Lesson

Pixel Programming for Step 1 ELL Students:
https://www.canadalearningcode.ca/lessons/pixel-programming-ell

Pixel Programming: Instructions, Programs + Solutions
http://bit.ly/pixel-programming

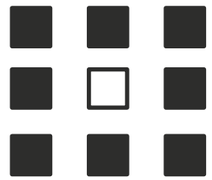
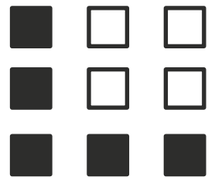
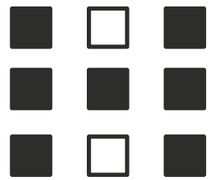
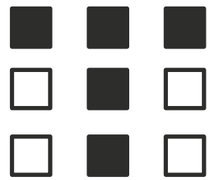
INSTRUCTIONS

1. We are programming left → right, top → bottom (like reading!)
2. Choose 2 different coloured sticky notes
3. The number is the amount of sticky notes to place down in a row
4. The comma means switch colours

PROGRAMS

Program 1 3 1, 1, 1 3	Program 2 1, 2, 1, 2, 3
Program 3 1, 1, 1 3 1, 1, 1	Program 4 3, 1, 1, 1 1, 1, 1

SOLUTIONS

Program 1 	Program 2 
Program 3 	Program 4 

LESSON PLAN:

Deconstructing Code

DESCRIPTION

In this activity, learners practice interpreting code by acting out printed algorithms in small groups, then work to debug (or fix) the code together as a class.

Grade: 4-6, 7-8

Length: 20 minutes

Subjects: Language Arts

Coding Concepts: Events, Loops, Conditional statements

PREP WORK

- Print/scan the Decode Sheets for the main activity. Available at **http://bit.ly/scratch-decode** (1 per group of 6 learners).
- Cut out the decode sheets ahead of time.
- This activity requires some space to move around.
- The closure of this activity requires a computer with access to the internet (or the Scratch offline editor installed).

THE LESSON

Activity

- Divide the class into groups of 5-6.
- Give each group a stack of the cut out decode sheets.
- Instruct each learner to take one paper (#1-6) – No peeking!
- Learners will try to 'decode' or interpret the code on their paper by acting it out. *Note: Group participation may be required.
- Instruct learners to begin with a high-five (to replace the "When sprite is clicked" or "When space key is pressed" events).
- If learners try acting it out but get stuck, they can ask their group for help.

Closure

- All of the code that we looked at is originally from a project in Scratch!
- Check out the original project: **https://bit.ly/de-code-solution**
- Click "See Inside".
- Go through each number: Select the character from the Sprites area to see the code, then do what it says (e.g. click on it or press the space key).
- For each, ask the learners that were assigned this number if this is what they 'decoded' when they acted it out.
- Point out **Events** (Orange 'hat' blocks that tell us when to begin), **Loops** (yellow "C" shaped blocks that make things happen more than once), and Conditional statements (yellow "IF, THEN" blocks that only happen IF something is true).
- Note: One of the sprites doesn't do anything when we click on it (#4) – Why not? Try to debug (fix it) together. What does this tell us about computers? (They take things very literally! We need to give them super clear instructions).

Extensions

If you have access to computers, have learners create their own algorithms in Scratch, then have a partner try to decode and act out their instructions before seeing how the sprite interprets the code.

Alternatively, print out larger paper versions of 5-10 different blocks from Scratch and have learners put them together like puzzle pieces. Then, have another learner decode their algorithm by acting it out.

Assessment

Learning Outcomes

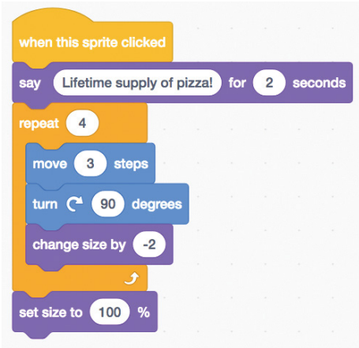
- Computers need very clear instructions in a specific order (or sequence).
- Events tell the computer when to start executing code.
- Loops make things happen more than once.
- Conditional statements control IF something happens or not.
- Sometimes computers don't understand our instructions, and we need to debug (or fix) the code.

Suggested Lesson

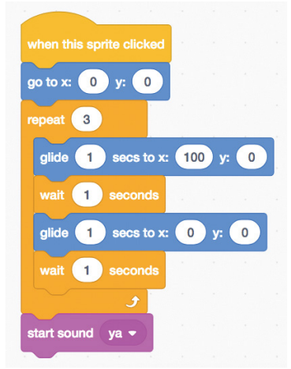
Source Whisperer: Decode the HTML behind a webpage: **https://www.canadalearningcode.ca/lessons/source-whisperer/**

Deconstructing Code: Decode Sheets **http://bit.ly/scratch-decode**

CHALLENGE: #1

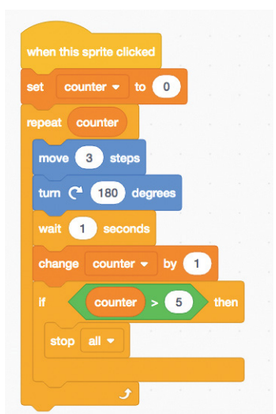


CHALLENGE: #2

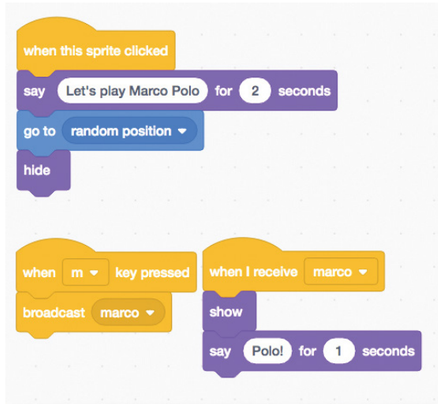


CHALLENGE: #3

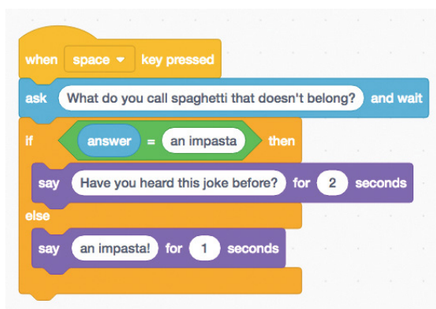
CHALLENGE: #4



CHALLENGE: #5



CHALLENGE: #6



LESSON PLAN:

Songwriting

This activity was created by Code.org

DESCRIPTION

One of the most magnificent structures in the computer science world is the function. Functions (sometimes called procedures) are mini programs that you can use over and over inside of your bigger program. This lesson will help learners intuitively understand why combining chunks of code into functions can be such a helpful practice.

Grade: 4-6

Length: 45 minutes

Subjects: Language Arts

Coding Concepts: Functions, Abstraction, Debugging

PREP WORK

- Watch the Songwriting and Songwriting with Parameters (Functions) – Teacher Video.
- Watch the Functions Unplugged: Songwriting – Lesson in Action Video.
- Print several copies of the Functions Unplugged: Songwriting – Worksheet for each group.
- Print one copy of the Functions Unplugged: Songwriting – Assessment for each learner.
- Access to the internet, or pre-downloaded songs and lyrics for activity.
- Make sure every learner has a Think Spot Journal – Reflection Journal.

Links to videos available at: **http://bit.ly/unplugged-songwriting**

THE LESSON

Introduction (20 minutes)

Vocabulary

This lesson has one new and important word:

- Function – Say it with me: Func-shun

A piece of code that you can call over and over again.

Sing a Song

- Let the class know that today is song day!
- We're going to learn a song together.
  - Start with a simple song, either written out or projected on the screen.
  - Point to the chorus and be sure that the class knows how it goes before you begin on the song.
  - Blast through the song, singing it with them in the beginning, then see what happens when you get to the part where it calls the chorus.

Chorus

Little bunny Foo Foo
Hopping through the forest
Scooping up the field mice
And bopping 'em on the head
Down came the Fairy
And she said
"Little bunny Foo Foo
I don't wanna see you
Scooping up the field mice
*And bopping 'em on the head"**

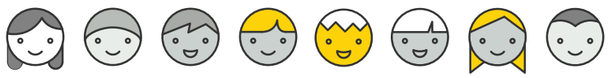
Song

Chorus
I'll give you 3 chances.
Then I'll turn you into a goon!
The next day. . .

Chorus
I'll give you 2 chances.
Then I'll turn you into a goon!
The next day. . .

Chorus
I'll give you 1 chance.
Then I'll turn you into a goon!
The next day. . .

Chorus
"I gave you two chances.
Now I'll turn you into a goon!"
(POOF!)
And the moral of the story is:
Hare today, goon tomorrow!



→ It's quite likely that the majority of the class will sing the lyrics for the chorus when you point to that bit.

Stop the song once that happens, and explicitly highlight what just happened.

- You defined the chorus.
- You called the chorus.
- They sang the chorus.

Ask the class why they suppose you only wrote the chorus once at the top of the paper instead of writing it over and over in each place where it is supposed to be sung.

- What are other benefits of only writing the chorus once when you sing it many times?

TEACHING TIP

Little Bunny Foo Foo is being used here as an example only. If your learners know this song, feel free to use it. Otherwise, choose an appropriate song that they might be more familiar with (either from music class or the radio.)

Now, imagine that this song is a computer program. Defining a title (like "chorus") for a little piece of code that you use over and over again is called creating a function.

This is helpful to computer scientists for some of the same reasons that it is helpful to songwriters.

- It saves time not having to write all the code over and over in the program.
- If you make a mistake, you only have to change it one place.
- The program feels less complicated with the repeating pieces defined just once at the top.

We are going to play with songs a little more, to try to really understand how often this technique is used!

Activity (20 minutes)

A fantastic way to compare functions to something we see in our everyday lives is to look at songs. Songs often have certain groups of lyrics that repeat over and over. We call such a group a "chorus."

Directions

- Divide into groups of 4, 5, or 6.
- Give each group several copies of the Songwriting Worksheet.
- Play a short song for the class that contains a clear chorus that does not change from verse to verse.
- Challenge the class to identify (and write down) the chorus.
- Compare results from each group.

Did everyone get the same thing? Sing your choruses together to find out! Play this game over and over until the class has little trouble identifying the choruses.

It is often easier just to have the class listen to (or watch) the song, then vote on what the chorus is by singing it together, rather than writing the whole thing down. If you choose this method, consider having the class do a written chorus for the final song selection to be sure that the visual learners get proper reinforcement.

Closure (5 minutes)

Flash Chat: What did we learn?

Lesson Tip

Flash Chat questions are intended to spark big-picture thinking about how the lesson relates to the greater world and the learners' greater future. Use your knowledge of your classroom to decide if you want to discuss these as a class, in groups, or with an elbow partner.

- Would you rather write lyrics over and over again or define a chorus?
- Do you think it's possible to make multiple choruses for the same song?
- Does it make sense to make a new chorus for every time it's needed in a song?

SONGWRITING LESSON TIP

Journaling

Having learners write about what they learned, why it's useful, and how they feel about it can help solidify any knowledge they obtained today and build a review sheet for them to look to in the future.

Journal Prompts

- What was today's lesson about?
- How do you feel about today's lesson?
- What is a function and how do you use it?
- Can you think of another activity where you might want to call a special group of instructions several times?

Extensions

Use these activities to enhance learning. They can be used as outside of class activities or other enrichment.

Visit the CS Fundamentals Unplugged Table (**https://code.org/curriculum/unplugged**). This activity does take a few supplies from the craft store, but it helps learners to see the value of calling multiple functions.

Create Your Song

- Start by creating a chorus together, then repeat it between verses of a song that you develop around it.
- Make a change to the chorus, and ponder how much easier it is to change in just one place.
- Change the chorus again, making it much longer than it was originally.
- Add a second chorus and alternate between them in your verses.

Songwriting a Program

- What if we acted out songs instead of singing them? All of a sudden, our chorus would be a function of repeated actions, rather than words.
- Use the concepts of the arrows from the Graph Paper Programming (**https://code.org/curriculum/unplugged**) lesson and create a program with lots of repeating instructions.
  - Circle those repeating actions so that the class can see where they are.
  - Define a function called "Chorus" above the program.
  - Cross out everywhere the repeating actions appear in the program and write "Chorus" instead.
- Repeat until the class can go through this process with little direction.

ASSESSMENT

Learning Outcomes

Learners will be able to:

- Locate repeating phrases inside song lyrics.
- Identify sections of a song to pull into a function.
- Describe how functions can make programs easier to write.

Assessment Ideas

Hand out the assessment worksheet and allow learners to complete the activity independently after the instructions have been well explained. This should feel familiar, thanks to the previous activities.

Suggested Lesson

Additional unplugged activities by Code.org:
https://code.org/curriculum/unplugged

Functions Unplugged: Songwriting - Worksheet
https://code.org/curriculum/course3/9/Activity9-Songwriting.pdf

Functions Unplugged: Songwriting - Assessment
https://code.org/curriculum/course3/9/Assessment9-Songwriting.pdf

* This activity was created by Code.org

SONGWRITING

Using Lyrics to Explain Functions and Procedures

One of the most magnificent structures in the computer science world is the function. Functions (sometimes called procedures) are mini programs that you can use over and over inside of your bigger program.

A fantastic way to compare functions to something we see in our everyday lives is to look at songs. Songs often have certain groups of lyrics that repeat over and over. We call such a group a "chorus."

Directions

1. Divide into groups of 4, 5, or 6.
2. Give each group several copies of the Songwriting Worksheet.
3. Play a short song for the class that contains a clear chorus that does not change from verse to verse.
4. Challenge the class to identify (and write down) the chorus.
5. Compare results from each group. Did everyone get the same thing?

Let's make a function for the bits that we use most often so that we don't need to write so much as we go.

Name:	Date:

Using Lyrics to Explain Functions and Procedures

SONG NAME:

CHORUS:

SONG NAME:

CHORUS:

Group Name:	Date:

SONGWRITING

Using Lyrics to Explain Functions – Assessment

Look at the lyrics for the two songs below.

If it were your job to write these songs as computer programs, what chunk of code from each would you turn into a function so that you could use it over and over again with just one word?

Circle the segments of each program that repeat most often. Is everything that youcircled exactly the same? If so, that can be your chorus!

Finish by writing the chorus for each song on the Songwriting Worksheet and give it a name. Those are your functions!

Song 1: I'm a Nut

I'm a little acorn brown
sitting on the cold, cold ground.
Everybody steps on me
that is why I'm cracked, you see.

I'm a nut
I'm a nut
I'm a nut, I'm a nut, I'm a nut

Called myself on the telephone
just to see if I was home.
Asked myself out on a date.
Picked me up at half-past eight.

I'm a nut
I'm a nut
I'm a nut, I'm a nut, I'm a nut

Took myself to the picture show.
Sat myself in the very first row.
Wrapped my arms around my waist.
Got so fresh, I slapped my face!

I'm a nut
I'm a nut
I'm a nut, I'm a nut, I'm a nut

Song 2: Skip to my Lou

Lou, Lou, skip to my Lou,
Lou, Lou, skip to my Lou,
Lou, Lou, skip to my Lou,
Skip to my Lou, my darlin'.

Fly's in the buttermilk,
Shoo, fly, shoo,
Fly's in the buttermilk,
Shoo, fly, shoo,
Fly's in the buttermilk,
Shoo, fly, shoo,

Skip to my Lou, my darlin'.
Lou, Lou, skip to my Lou,
Lou, Lou, skip to my Lou,
Lou, Lou, skip to my Lou,

Skip to my Lou, my darlin'.
Cows in the cornfield,
What'll I do?
Cows in the cornfield,
What'll I do?
Cows in the cornfield,
What'll I do?

Skip to my Lou, my darlin'.
Lou, Lou, skip to my Lou,
Lou, Lou, skip to my Lou,
Skip to my Lou, my darlin'.

Group Name:	Date:

* This activity was created by Code.org

Using Lyrics to Explain Functions – Assessment

SONG NAME:

CHORUS:

SONG NAME:

CHORUS:

Group Name:	Date:

Resources

Visit the Teachers Learning Code Website for more unplugged activities and lesson plans:
https://www.canadalearningcode.ca/lesson-plans/

DO THE ROBOT LESSON

Lesson plan on Canada Learning Code website: https://www.canadalearningcode.ca/lessons/do-the-robot

Suggested lesson - Basketbots: https://www.canadalearningcode.ca/lessons/basketbots

DECONSTRUCTING CODE LESSON

Lesson plan on Canada Learning Code website: https://www.canadalearningcode.ca/lessons/deconstructing-code

Decode Sheets: http://bit.ly/scratch-decode

Solution in Scratch: https://bit.ly/de-code-solution

Suggested lesson – Source Whisperer: https://www.canadalearningcode.ca/lessons/source-whisperer

SPEED STACKER LESSON

Lesson plan on Canada Learning Code website: https://www.canadalearningcode.ca/lessons/speed-stacker

Symbol Key and Cup Stack sheets: http://bit.ly/speed-stacker-thinkersmith

Example Cup Stack solution: http://bit.ly/speed-stacker-eg

Suggested lesson – Unplugged intro to Coding for Kindies: https://www.canadalearningcode.ca/lessons/unplugged-intro-to-coding-for-kindies

SONGWRITING LESSON

Lesson plan on Code.org: http://bit.ly/unplugged-songwriting

*Including links to videos and print materials

For more unplugged activities and lesson plans, check out our website: canadalearningcode.ca/lesson-plans

PIXEL PROGRAMMING LESSON

Lesson plan on Canada Learning Code website: https://www.canadalearningcode.ca/lessons/pixel-programming

Pixel Programming Instruction, Programs, and Solutions: http://bit.ly/pixel-programming

Suggested lesson – Pixel Programming for Step 1 ELL Students: https://www.canadalearningcode.ca/lessons/pixel-programming-ell

ADDITIONAL RESOURCES

CS Unplugged: https://csunplugged.org/en/

Robot Turtles: http://www.robotturtles.com

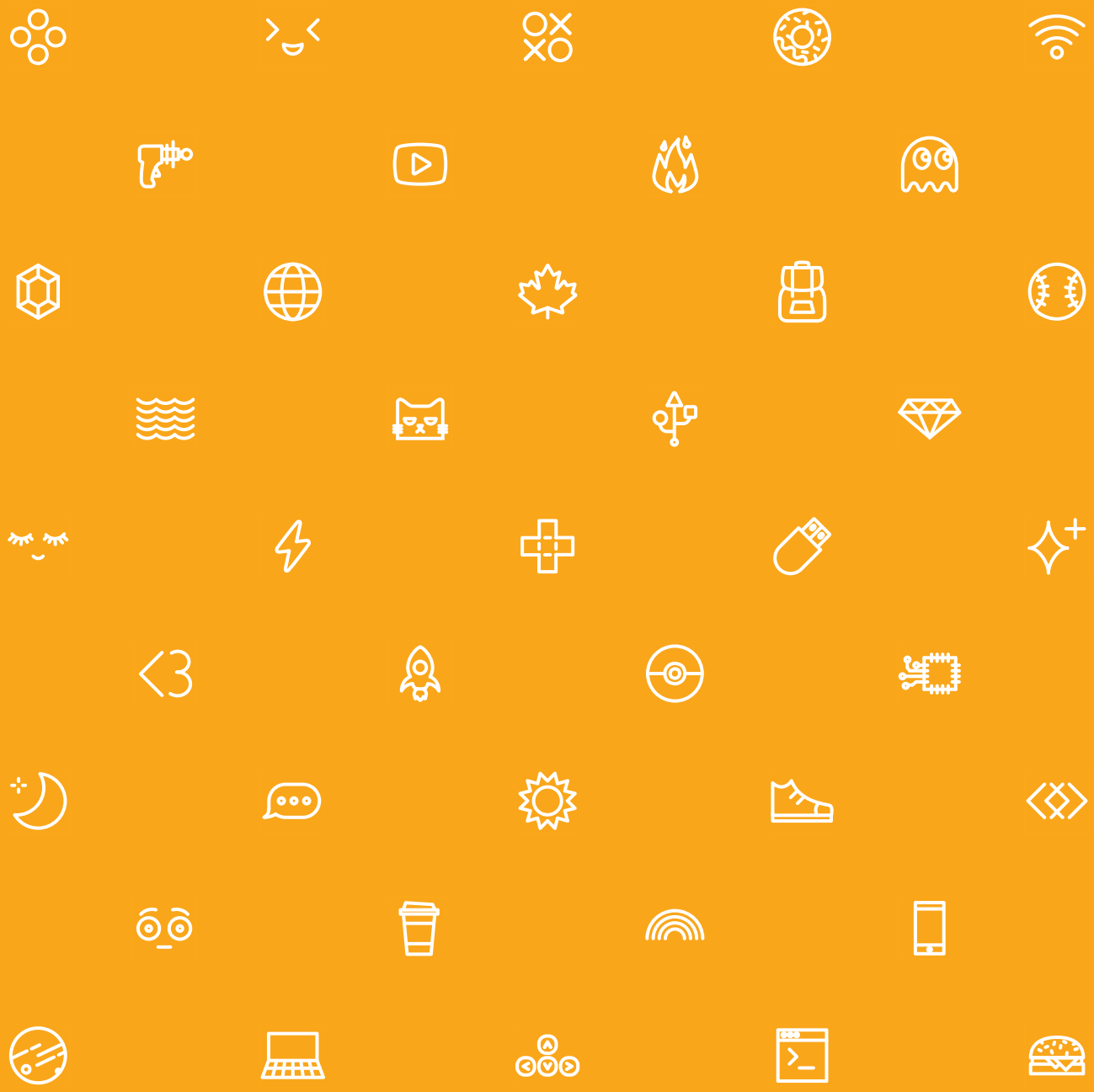
Littlecodr cards: http://littlecodr.com

Potato Pirates: https://www.potatopirates.game

Scratch Jr Coding Cards: http://bit.ly/scratchjr-coding-cards-indigo

DATE:

DATE:



GET IN TOUCH
Teachers Learning Code
30 St Patrick St
Toronto, ON M5T 3A3

info@teacherslearningcode.ca
teacherslearningcode.ca

teachers learning code