Enriched Module on Coding Education for Upper Primary Level

Technology Education Section
Curriculum Support Division
Education Bureau
June 2023

Primary Four 4
Student Version
Booklet 2

Preface

The Education Bureau actively promotes innovation and technology (I&T) education for all students. Continuous incorporation of I&T learning elements into both the primary and secondary curricula helps strengthen the cultivation of students' interest in and capability of learning information technology and I&T from an early age, equip students with 21st century skills, and unleash their creativity and potential.

To enhance I&T education, the Education Bureau has launched the "Enriched Module on Coding Education for Upper Primary Level" for schools to adopt. Designed in accordance with the revised "Computational Thinking - Coding Education: Supplement to the Primary Curriculum" published in 2020, the curriculum module helps teachers integrate I&T elements into classroom learning more systematically. Schools should conduct appropriate curriculum planning with reference to the content of the "Enriched Module on Coding Education for Upper Primary Level", and incorporate 10 to 14 hours of enriched coding education for all upper primary students every year in order to further develop their computational thinking and strengthen their I&T learning.

The "Enriched Module on Coding Education for Upper Primary Level" is adapted from learning and teaching resources of the "CoolThink@JC" project initiated and funded by The Hong Kong Jockey Club Charities Trust and co-created by The Education University of Hong Kong, Massachusetts Institute of Technology, and City University of Hong Kong. The Education Bureau is grateful for the collaboration with The Hong Kong Jockey Club Charities Trust in consolidating and drawing on the experience accumulated by the schools in the project to develop the "Enriched Module on Coding Education for Upper Primary Level" for adoption by all publicly-funded schools in Hong Kong. The Technology Education Section, Curriculum Support Division of the Education Bureau and Department of Mathematics and Information Technology of The Education University of Hong Kong co-developed the curriculum module based on the deliverables produced and experience gained in the project. Views on the content of the curriculum module were collected from the Committee on Technology Education of Curriculum Development Council and their support was sought.

The "Enriched Module on Coding Education for Upper Primary Level" covers basic coding and computational thinking concepts, namely abstraction, algorithm and automation, as well as connection with physical objects, the use of sensors and actuators to interact with the environment, etc., allowing students to develop their computational thinking as well as interest in and ability to learn I&T through the learning of coding.

This Primary 4 curriculum module, the first of three to be developed for upper primary levels (Primary 5 and 6 forthcoming), focuses on establishing a solid foundation for students' in the above basic concepts of coding and computational thinking; through

coding activities, logical thinking and problem solving skills are developed, and computational thinking is cultivated. There are a total of 8 units in the curriculum module, including 6 core units, and 2 optional extension units for schools to provide opportunities for students with a higher ability or strong interest in coding to enrich their learning and deepen their understanding of coding and innovative technology. The curriculum module also includes a project-based component that allows students to apply their computational thinking and creativity, and make good use of programming and innovative technology in different contexts, thereby formulating solutions to everyday problems for the benefit of society.

The recommended lesson time of the curriculum module (excluding the extension units) for each upper primary year level is 14 hours. Please refer to Table 1 and the Appendix for the arrangement of this Primary 4 curriculum module, the recommended lesson time, as well as the pedagogy to be adopted.

Table 1: Arrangement of the Primary 4 curriculum module and recommended lesson time

		Core Unit		**Extension Unit**	
Unit	**Unit Title**	**Recommended Lesson Time (in minutes)**	**No. of Lessons (35 minutes for each lesson)**	**Recommended Lesson Time (in minutes)**	**No. of Lessons (35 minutes for each lesson)**
1	Introducing Scratch Programming	70	2		
2	Exploring Under the Sea	70	2		
3	Storytelling	70	2		
4	Space Traveling	105	3		
5	Creating a Maze Game	140	4		
6	Creating a Maze Game with micro:bit			70	2
7	Drawing Shapes in Scratch	105	3		
8	Designing Line Pattern Art			70	2
	Final Project	280	8		
		840 (14 hours)	24	140	4

Views and suggestions on the "Enriched Module on Coding Education for Upper Primary Level" are always welcome. These may be sent to:

Chief Curriculum Development Officer (Technology Education)
Curriculum Support Division
Education Bureau
Room W101, 1/F, West Block
Kowloon Tong Education Services Centre
19 Suffolk Road, Kowloon Tong
Kowloon, Hong Kong

Fax: 2768 8664
E-mail: teched@edb.gov.hk

Appendix

Pedagogy

Teachers may make reference to the seven-step guide introduced in the Technological Pedagogical Content Knowledge (TPACK) framework for the teaching of computational thinking (CT). Technological content knowledge (TCK) refers to the knowledge of using block-based programming environments for coding. Content knowledge (CK) refers to the knowledge of CT concepts, practices, and attitudes to be taught. Pedagogical content knowledge (PCK) refers to pedagogies that do not involve the use of programming environments for teaching CK. TPACK refers to the integration of the use of technology and pedagogy to teach CK in context.

Based on the four dimensions of the TPACK framework above, teachers may adopt the seven-step guide in the instruction of each unit with a view to developing students' problem solving skills and digital creativity. The last three steps emphasise applying TCK to exploring the possible use of tools in the programming environments for the cultivation of digital creativity; revisiting and reviewing CK for consolidation; and reflection on PCK to engage in the improvement of teaching practices relevant to CK (Kong, Lai & Sun,2020; Kong & Lai, 2022; Kong, Lai & Li, 2023).

Step 1: TCK (Introducing features of the programming environment in a specific context)

Step 2: CK (Introducing computational thinking concepts, practices and attitudes to be taught)

Step 3: PCK (Adopting pedagogy such as allowing pre-coding access to games or apps to pave the way for reflection on the design of games or apps; and engaging in unplugged activities to enhance understanding of more difficult coding-related concepts, practices and attitudes)

Step 4: TPACK (Applying knowledge of using programming environments for teaching CK with appropriate pedagogy in a specific context)

Step 5: TCK (Encouraging students to suggest applications of relevant features of the programming environment in other contexts, thereby inspiring their digital creativity)

Step 6: CK (Helping students reflect on CT concepts, practices and attitudes to consolidate their learning)

Step 7: PCK (Conducting self-reflection on the pedagogy adopted in the unit with a view to improve the next round of teaching)

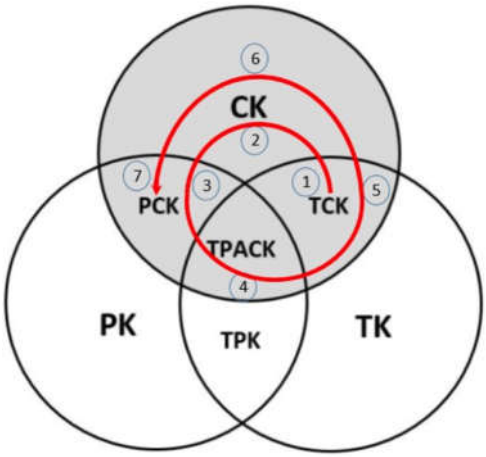


Figure 1 The seven steps in the shaded areas (CK, TCK, PCK, and TPACK) indicate those steps needed for teachers to teach content knowledge of CT. (Kong, Lai & Sun, 2020)

References

Education Bureau. (2020). *Computational Thinking - Coding Education: Supplement to the Primary Curriculum*. Hong Kong: Author.

Kong, S. C., & Lai, M. (2022). A proposed computational thinking teacher development framework for K-12 guided by the TPACK model. *Journal of Computers in Education, 9(3)*, 379-402.

Kong, S. C., Lai, M., & Sun, D. (2020). Teacher development in computational thinking: Design and learning outcomes of programming concepts, practices and pedagogy. *Computers & Education, 151*, 103872.

Kong, S. C., Lai, M., & Li, Y.G. (2023). Scaling up a teacher development programme for sustainable computational thinking education: TPACK surveys, concept tests and primary school visits. *Computers & Education, 194*, 104707.

Enriched Module on Coding Education for Upper Primary Level (Primary 4)
Editorial Board Members

Professor KONG Siu Cheung
Research Chair Professor of E-Learning & Digital Competency
Department of Mathematics and Information Technology
Director, Artificial Intelligence and Digital Competency Education Centre
Director, Centre for Learning, Teaching and Technology
The Education University of Hong Kong

Ms SIU Ka Yuk Cora
Educational Development Manager
Centre for Learning, Teaching and Technology
The Education University of Hong Kong

Ms MA Yunsi Tina
Assistant Educational Development Manager
Centre for Learning, Teaching and Technology
The Education University of Hong Kong

Miss TING Wan Yee
Assistant Educational Development Manager
Centre for Learning, Teaching and Technology
The Education University of Hong Kong

Mr LO Hoi Lam Ken
Educational Development Officer
Centre for Learning, Teaching and Technology
The Education University of Hong Kong

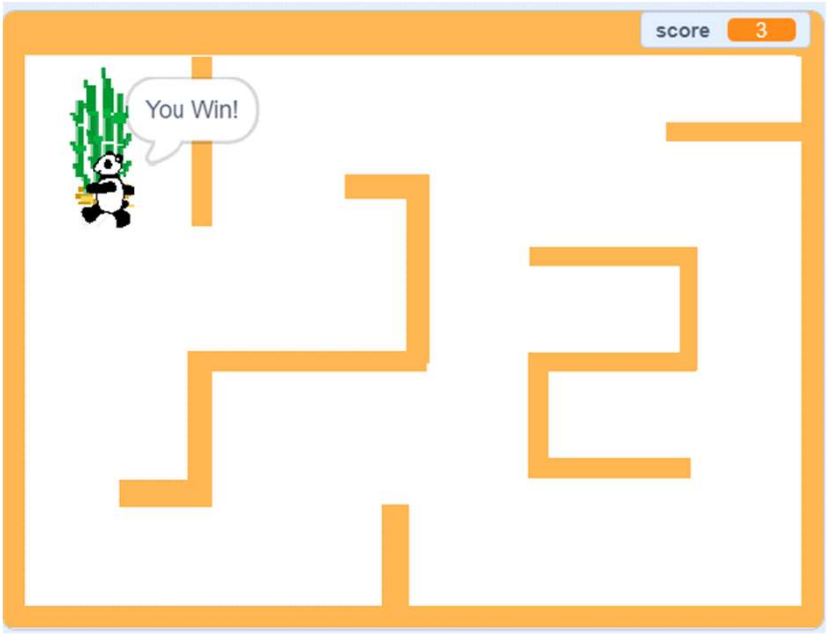
Content for Booklet 2

Unit 5: Creating a Maze Game Student Guide

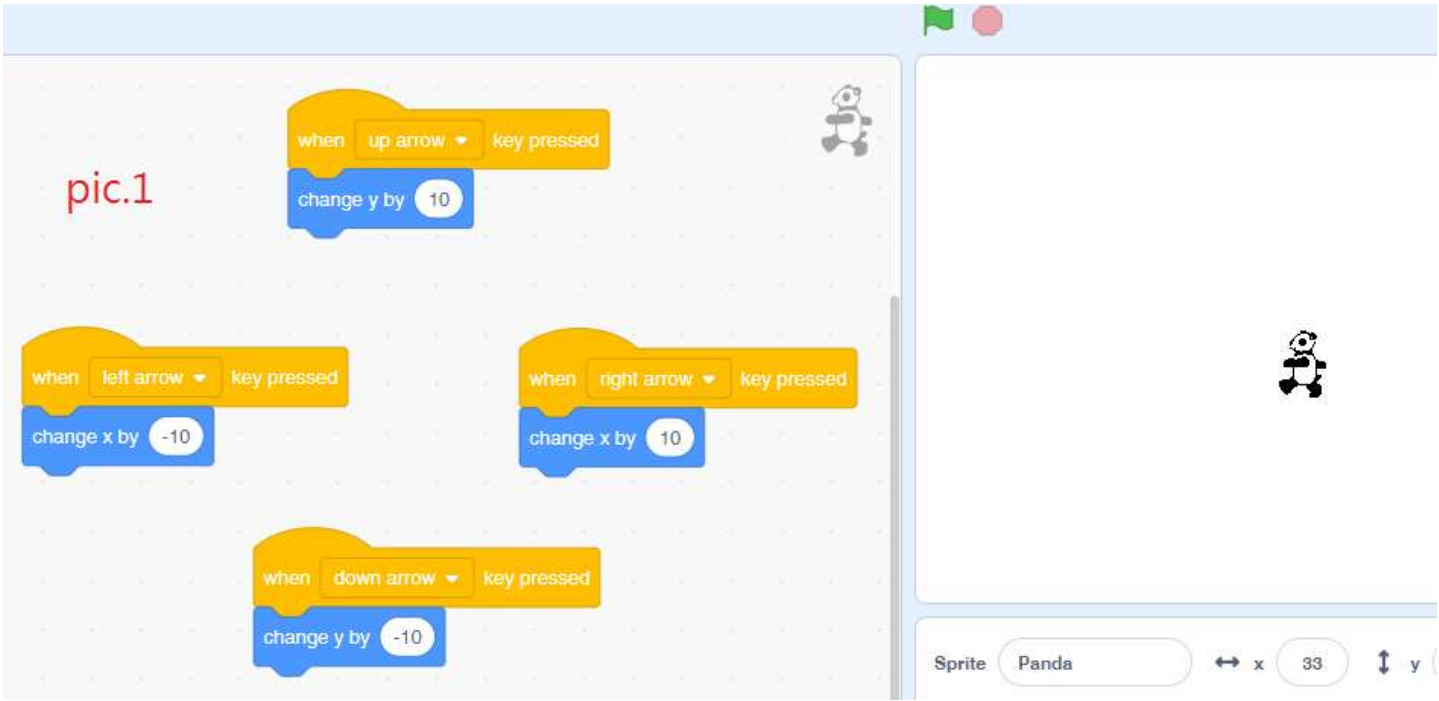
Content

Creating a Maze Game

Let's learn to make a game with Scratch!



To Play

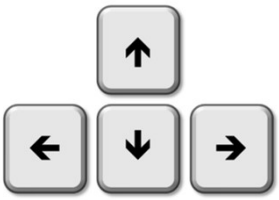


Creating a Maze Game

To Play

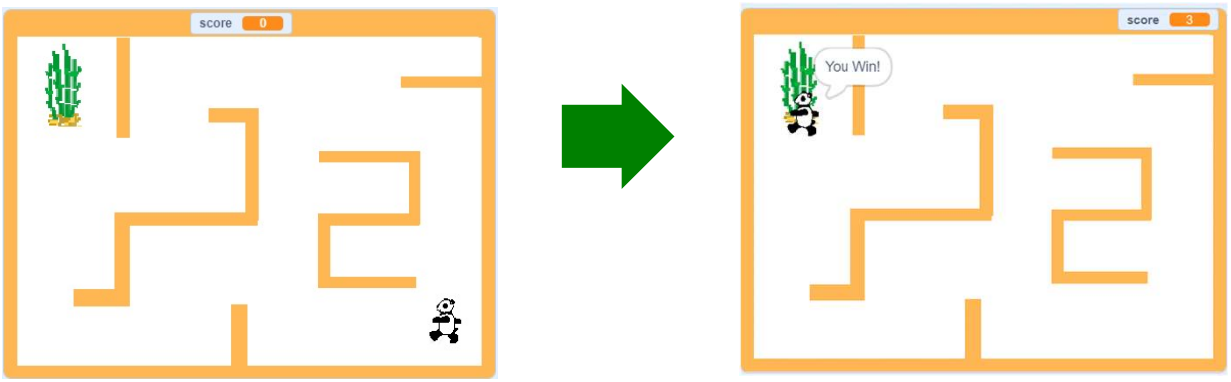
Play the Maze Game (Demo) https://scratch.mit.edu/projects/722154863 with keyboard (up / down /left / right arrow keys).

How can you win the game?



What happened when you touch the wall?

What happened when you touch the bamboo?

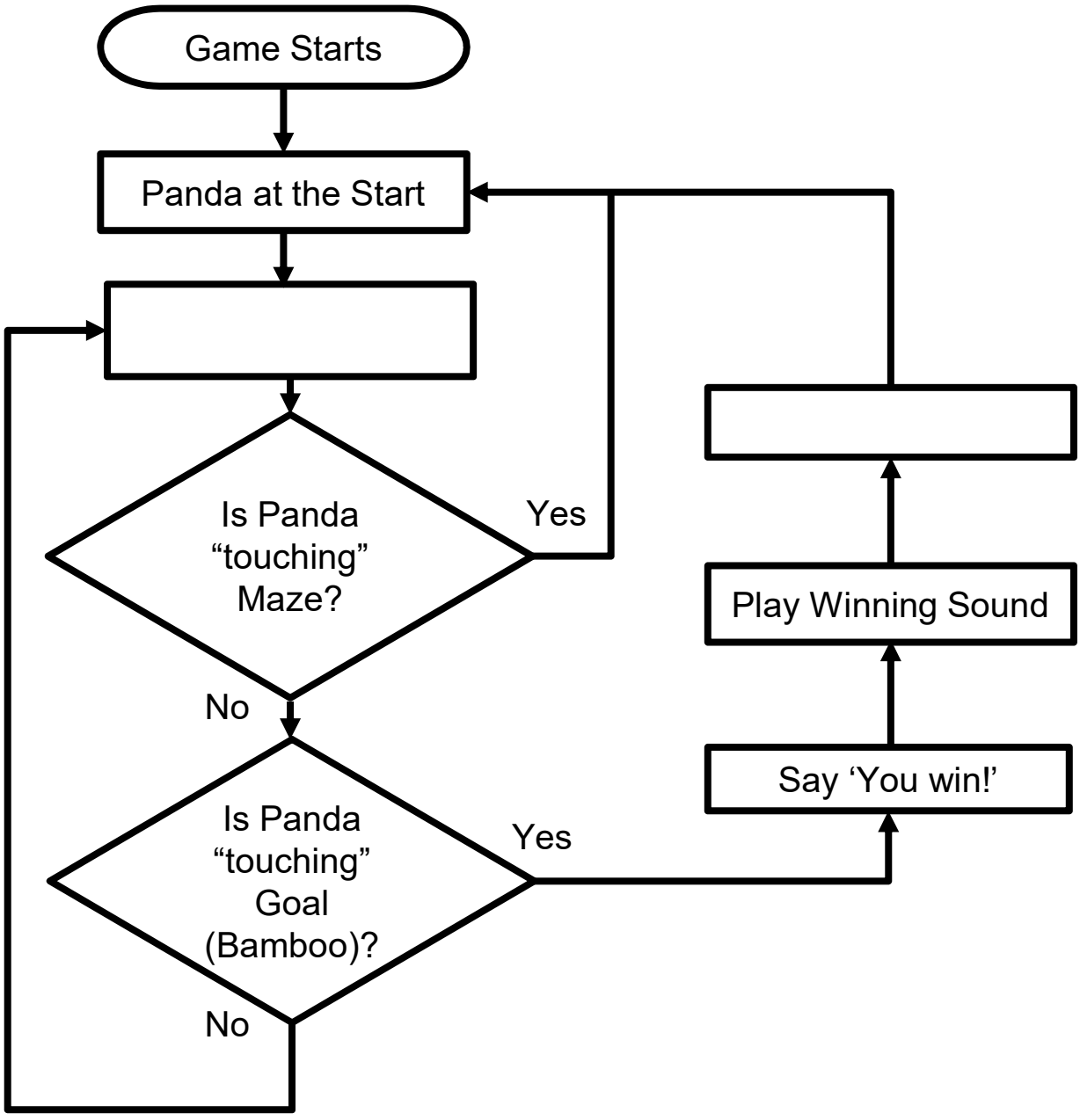


Creating a Maze Game

To Think

Complete the following flow diagram.

A. Control Panda with Keyboard	**B.** Score + 1

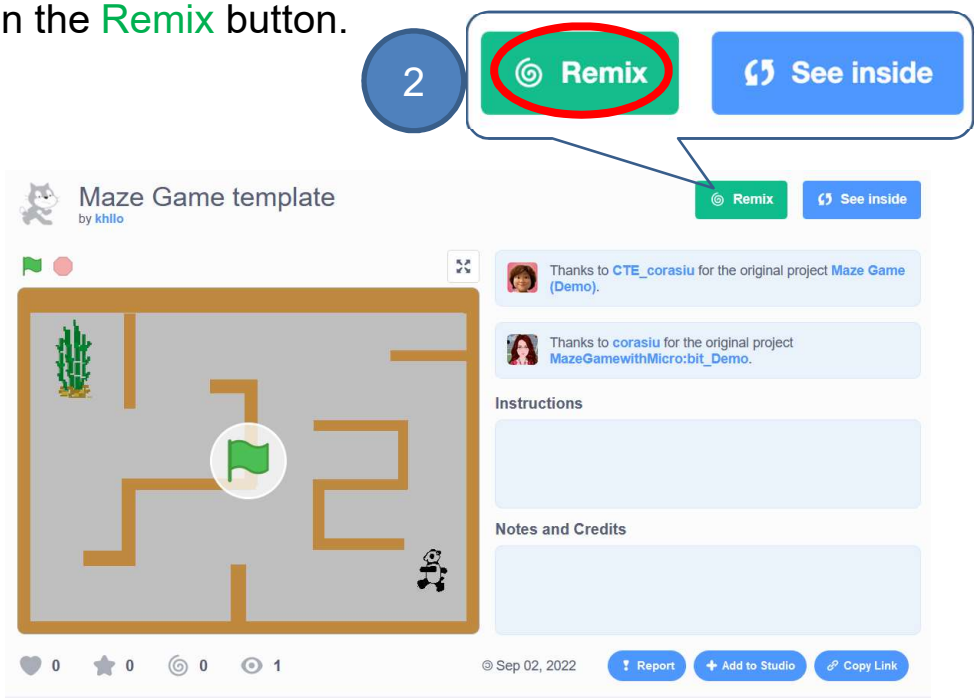


Creating a Maze Game

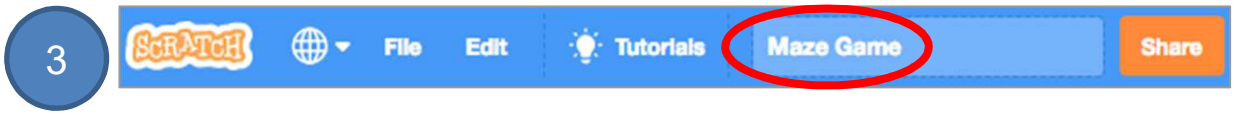
Start Here

1. Sign into your account at https://scratch.mit.edu/ 1

2. Open the Maze Game Template project:
   https://scratch.mit.edu/projects/727439171
   and click on the Remix button.



3. Rename the project as Maze Game.

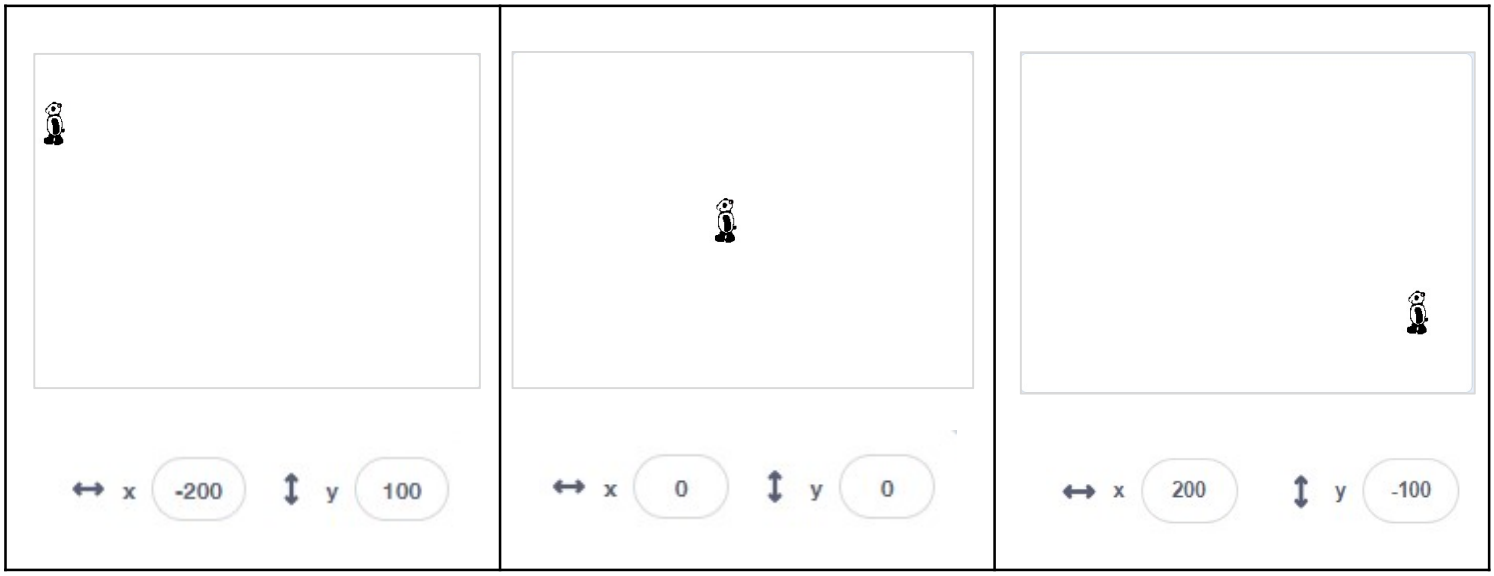


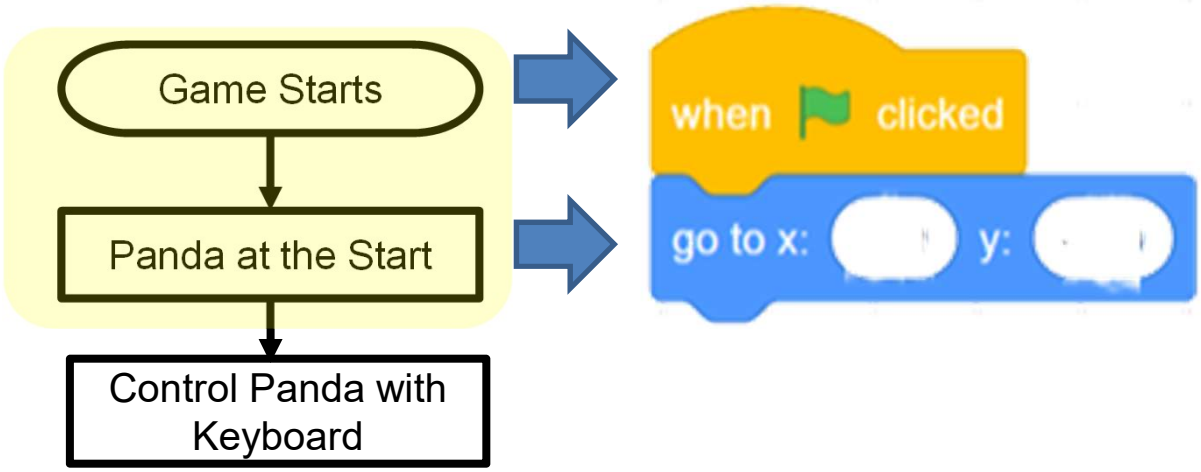
Creating a Maze Game

To Think and To Code: Panda at the Start

Any differences between these three pandas?

See Appendix P.33





Testing and Debugging

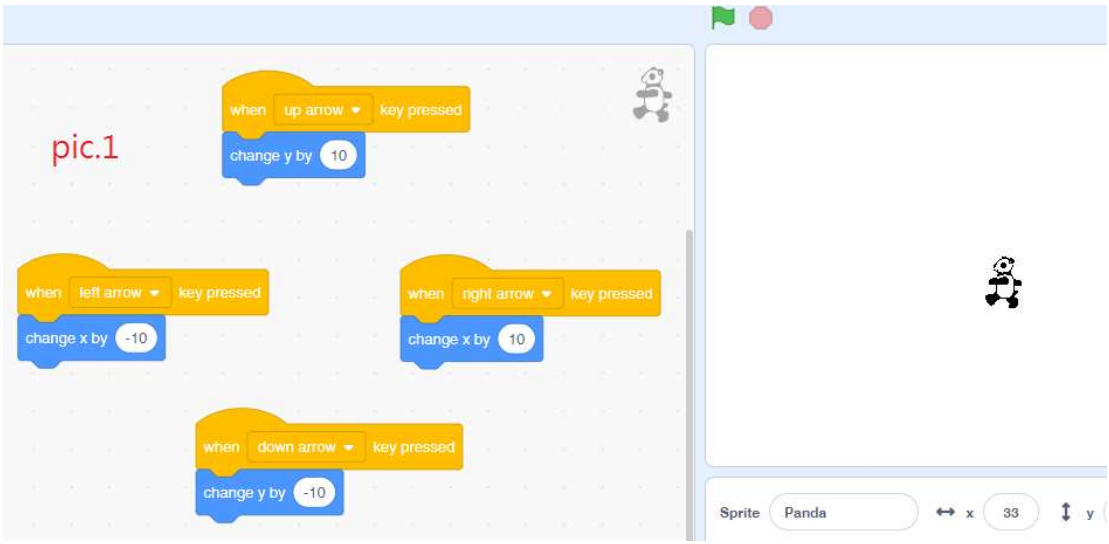
Click the green flag and see if Panda is back to the starting point.

Creating a Maze Game

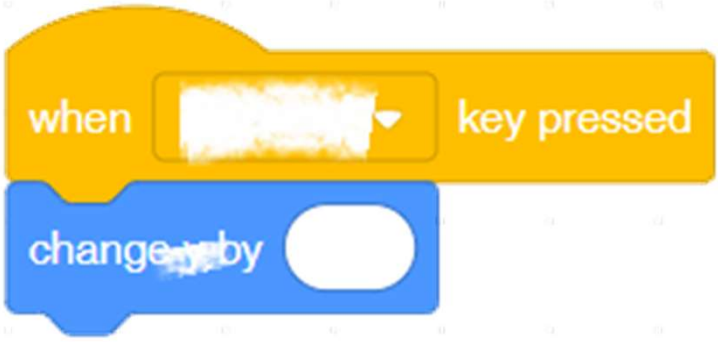
To Think and To Code: Control Panda with Keyboard

See Appendix P.33

Look at the blocks and backdrop:



Now modify the blocks to control the Panda's movement.



Testing and Debugging

Can you control the panda with four arrow keys?

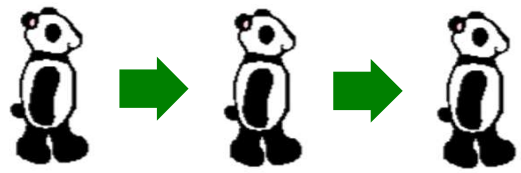
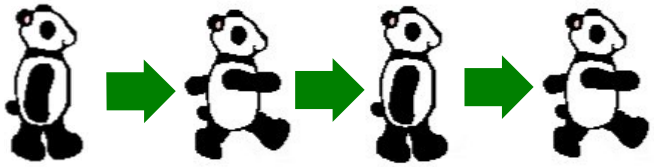
Panda at the Start	Control Panda with Keyboard		

Creating a Maze Game

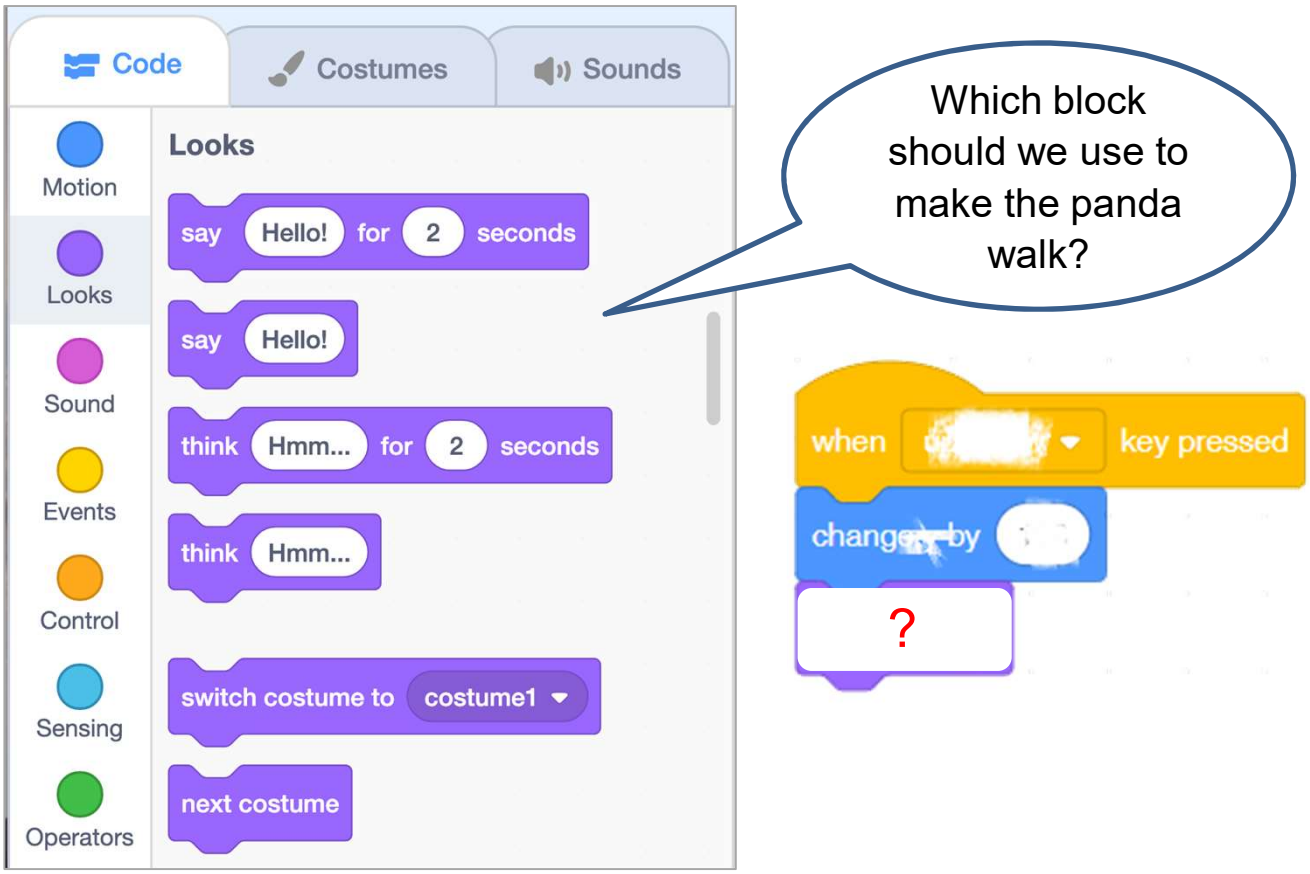
To Think and To Code: Make Panda Walk

Which panda seems like “walking” now?

See Appendix P.34

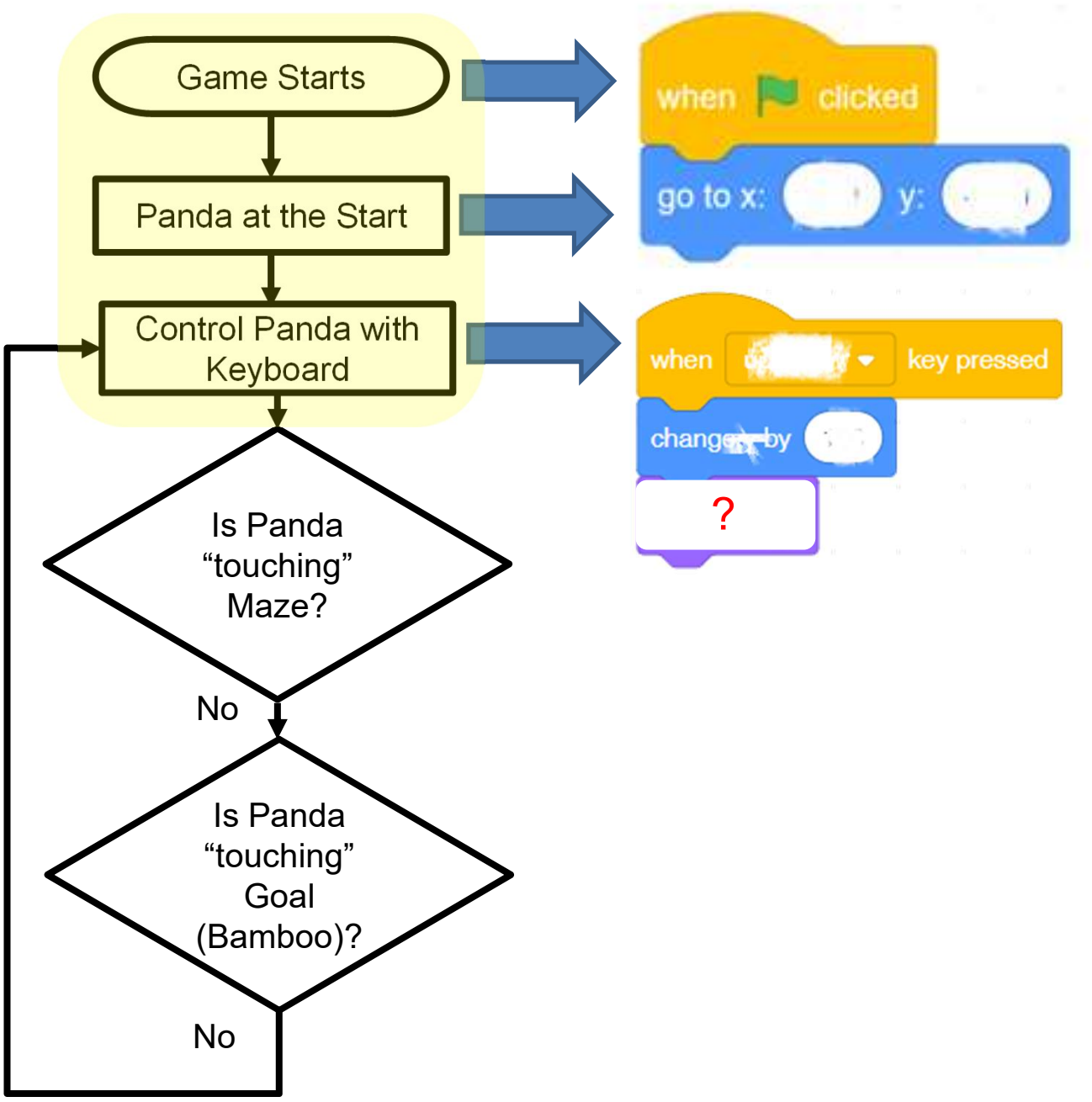
	
Panda A	Panda B

How would you change the sprite’s costume to make it look like it is walking?



Creating a Maze Game

To Think and To Code: Make Panda Walk



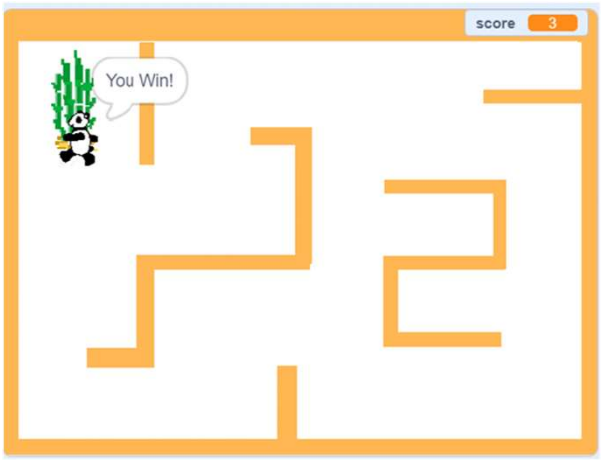
Testing and Debugging

Test to see if the Panda can walk now.

Panda at the Start → Control Panda with Keyboard → Make Panda Walk →

Creating a Maze Game

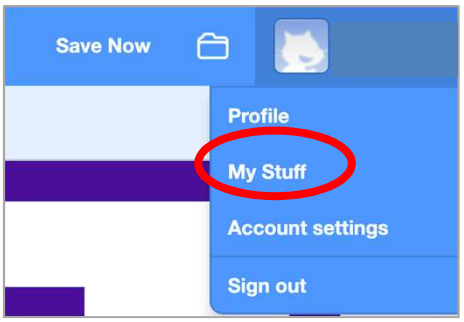
Let's finish our game!
In this game, you guide the panda to its bamboo forest with your keyboard, avoiding the walls of the maze. Reaching the maze earns + 1 point, hitting a wall puts panda back to the start. You will also add speech, score and sounds.



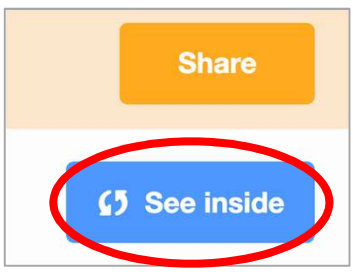
Start Here

1. Sign into your account at scratch.mit.edu.

2. Go to My Stuff and open up your maze game project.



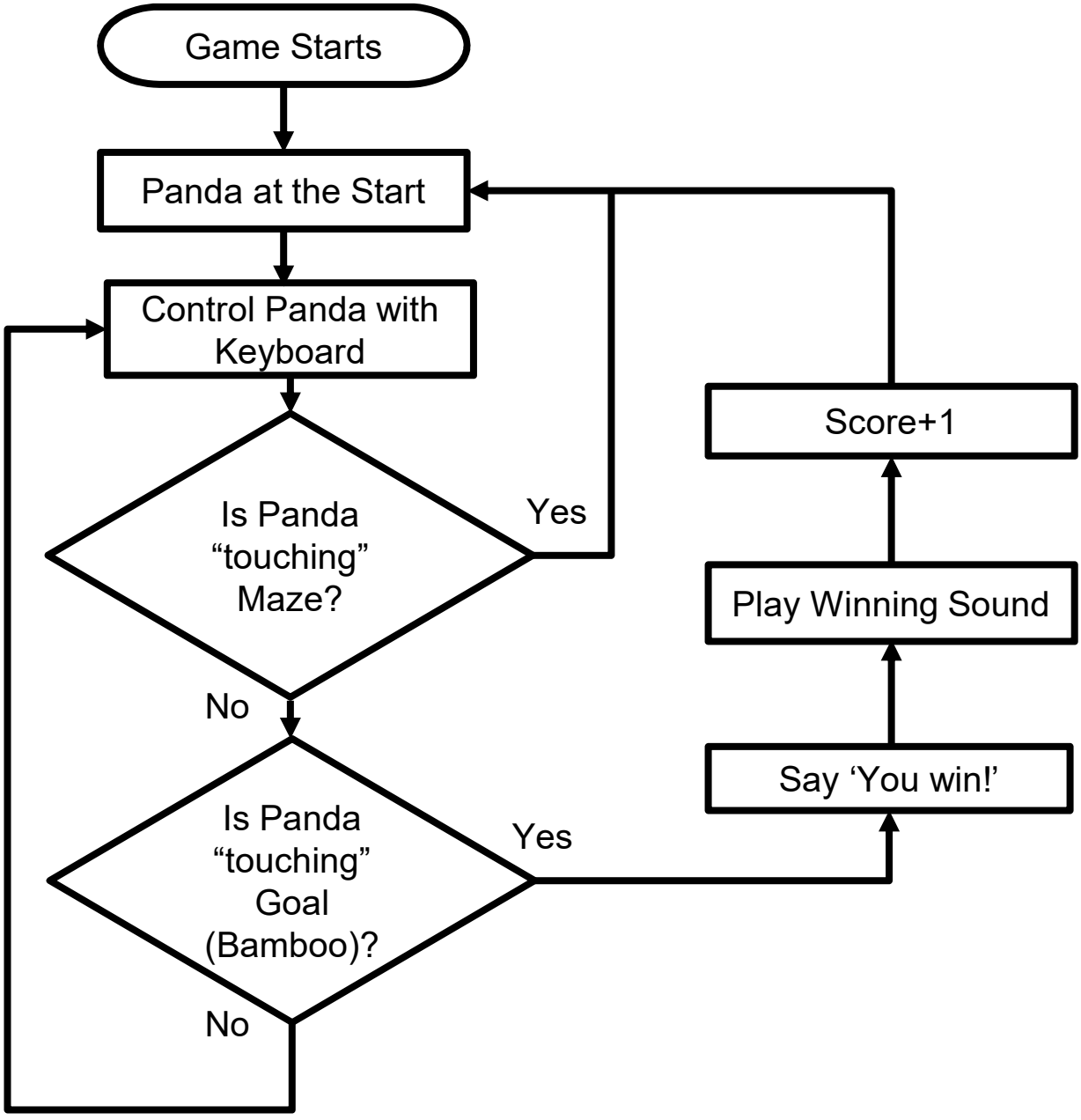
3. Click the See inside button in the top right area of the website to continue working on your game.



Creating a Maze Game

To Think: Branching / Selection (If the Panda touches…)

What will happen to the panda in the game?



Creating a Maze Game

To Think: Branching / Selection (If the Panda touches…)

In the template, we have the three sprites:

What would they do?

Panda is walking…

If touching [ ] ?

Then (what happens?)

_____

_____

If touching [ ] ?

Then

_____

_____

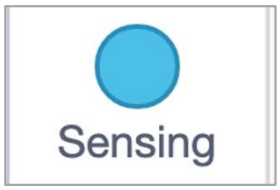
Creating a Maze Game

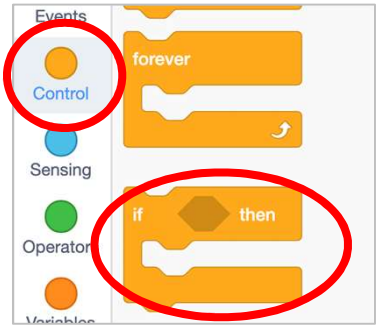
To Code: Touching Maze

See Appendix P.35

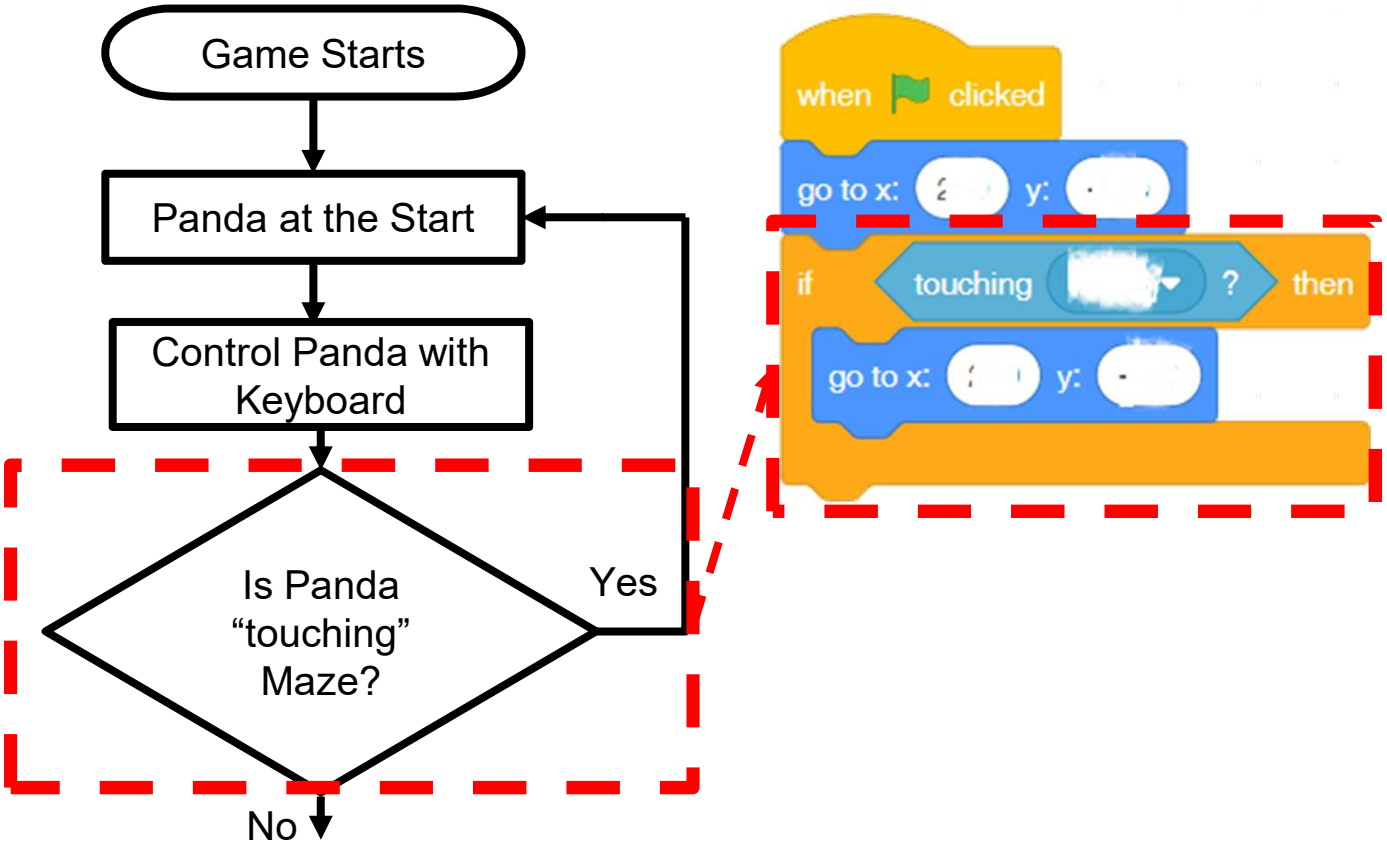
1. Throughout the game, we want the Panda sprite to act differently if it touches something. To do this, we should pull out an "if-then" block from the "Control" drawer.
2. With the algorithm you designed, what kind of conditions will trigger those actions?

   Hint: Take a look on the "Sensing" drawer.





3. If Panda is touching Maze:

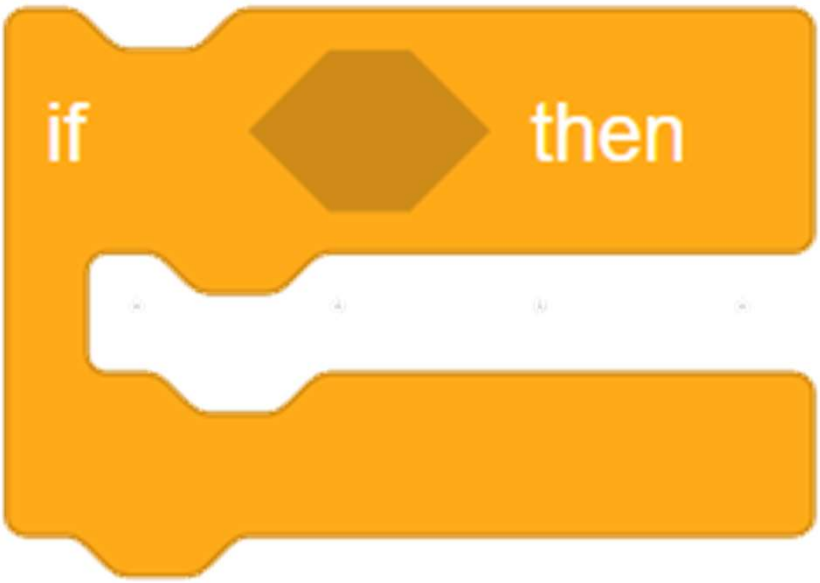


Creating a Maze Game

Review Branching / Selection

1. We use conditional statements in programming to enable computers to make _____.

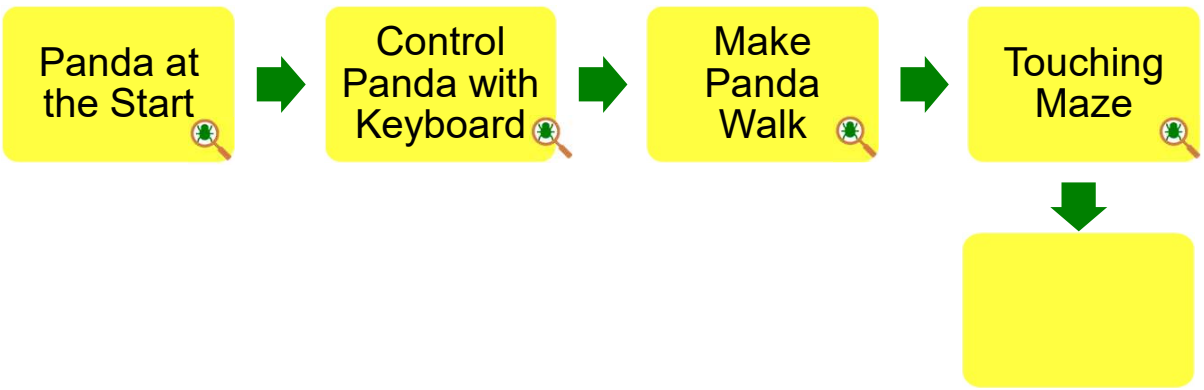
2. Conditionals always have an _____. part, which tells the program in the _____. part what to do when the condition is true.



Testing and Debugging

Time to test! Click the green flag and move the Panda hit the wall. Does it go back to the starting point?

Do it again, what is difference now?

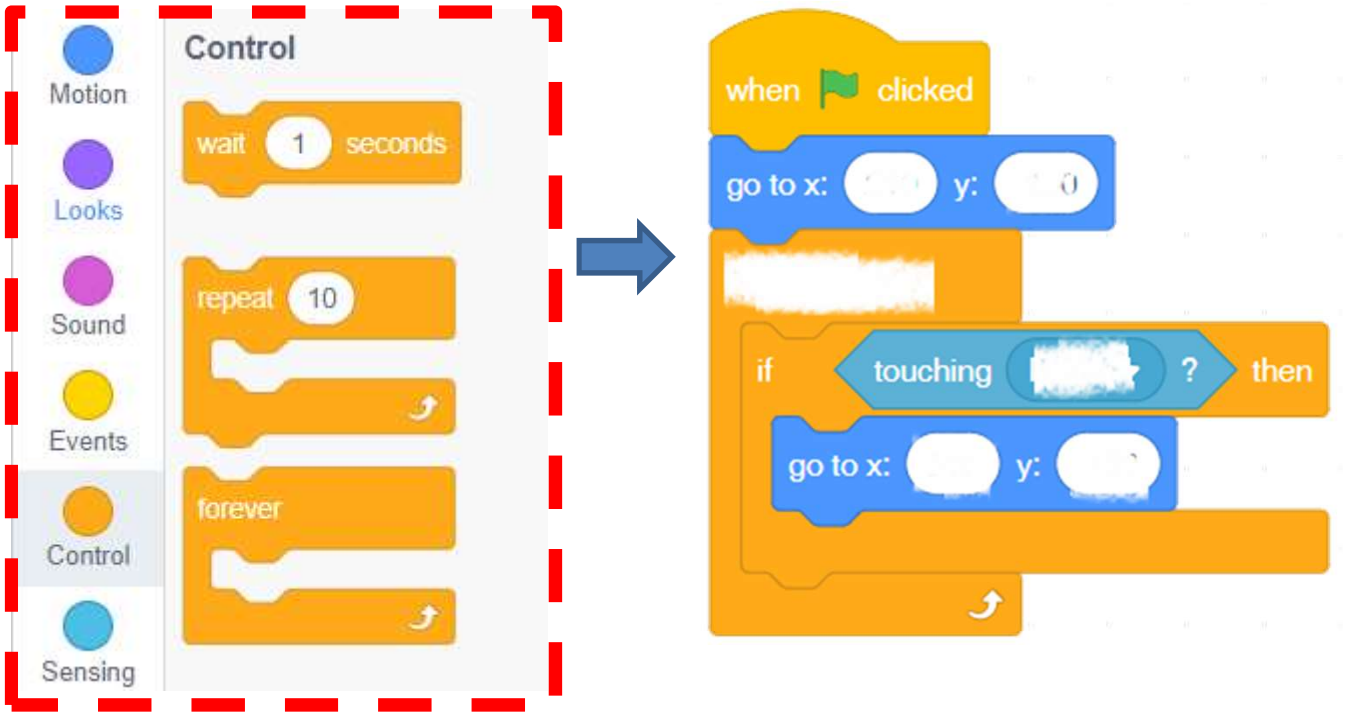


Creating a Maze Game

To Think and To Code: Iteration

Look at the "Control" drawer, which blocks will you use to help solve this problem?

See Appendix P.36



Do you need to keep checking every time the Panda walks?

☐ Yes ☐ No

Why? _____.

Testing and Debugging

Panda at the Start → Control Panda with Keyboard → Make Panda Walk → Touching Maze → Iteration → ____

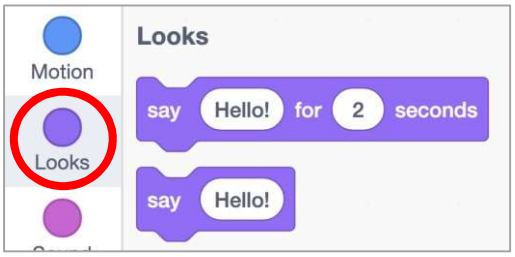
Creating a Maze Game

To Code: Touching Goal

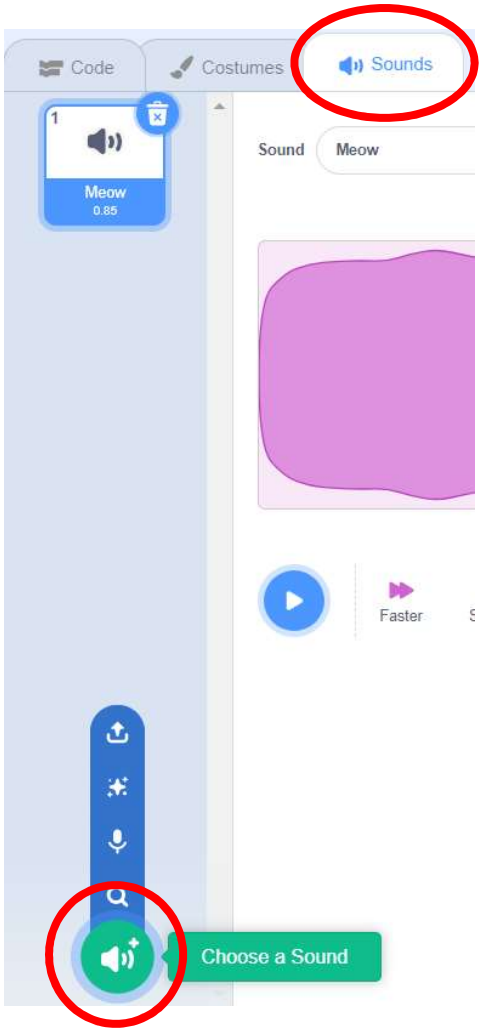
See Appendix P.37

If Panda touching the Goal (Bamboo):

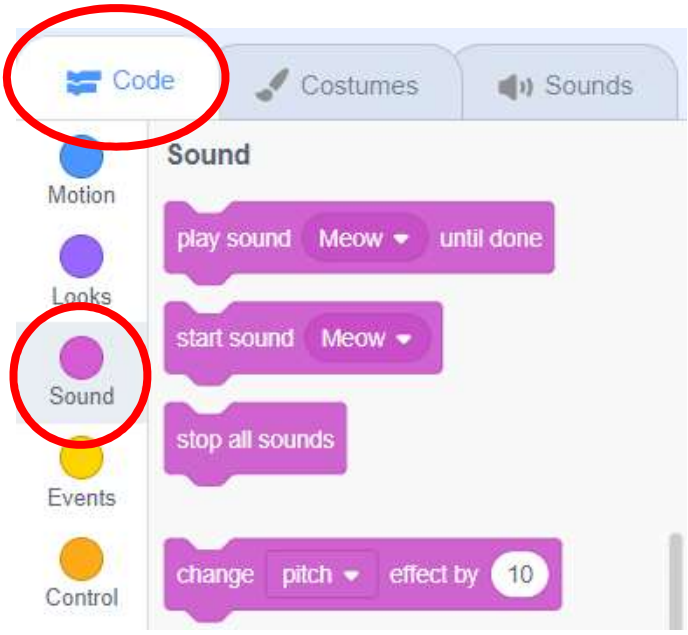
1. Make Panda say something, go to the "Looks" drawer.
   Which block do you use?



2. Add winning sound. Go to the "Sounds" tab. Click on the "Choose a Sound" icon at the bottom left to select a sound.



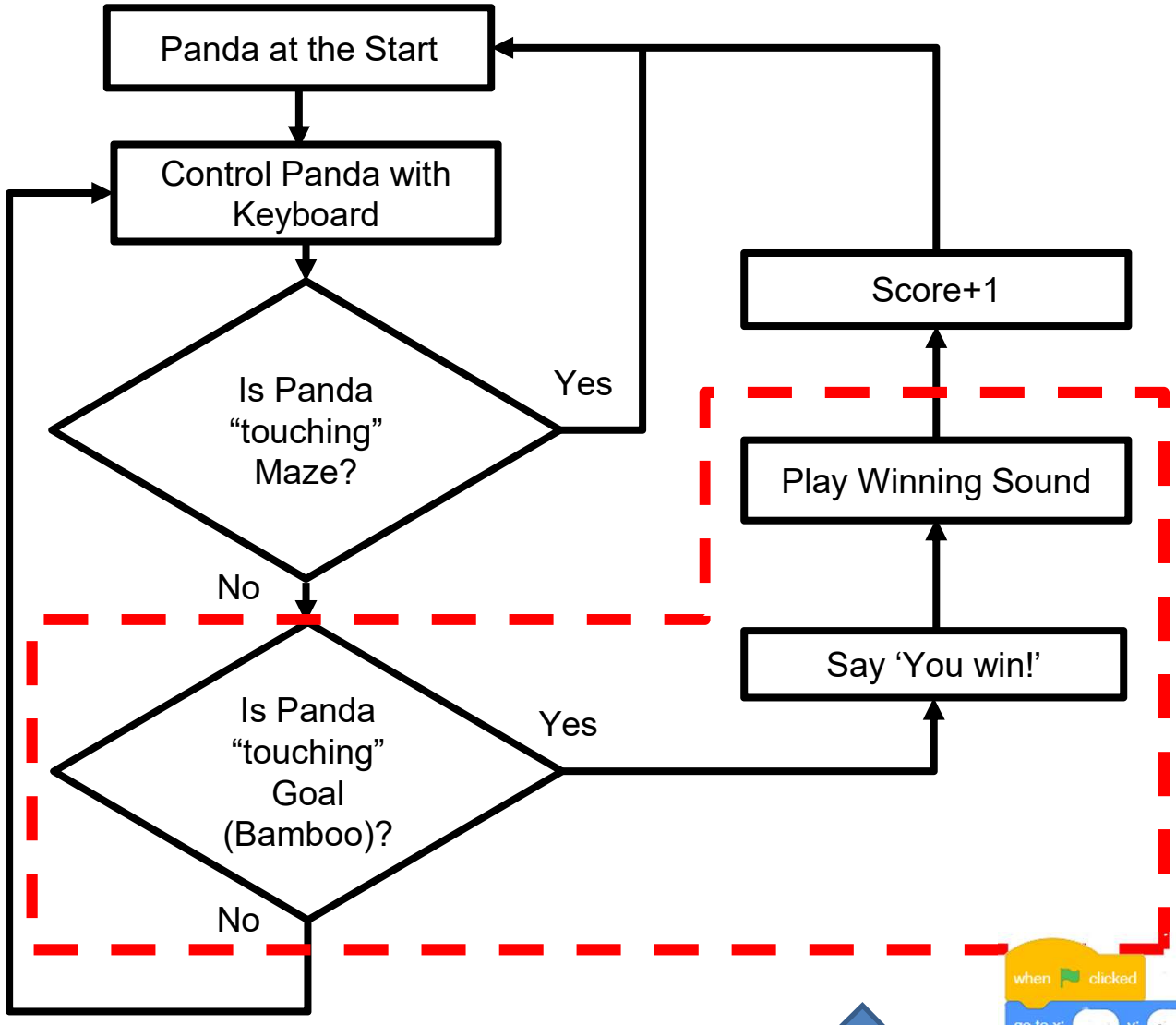
3. Do you remember how to make your sprite play the sound you found?
   Hint: Go to the "Sound" drawer of "Code" tab.

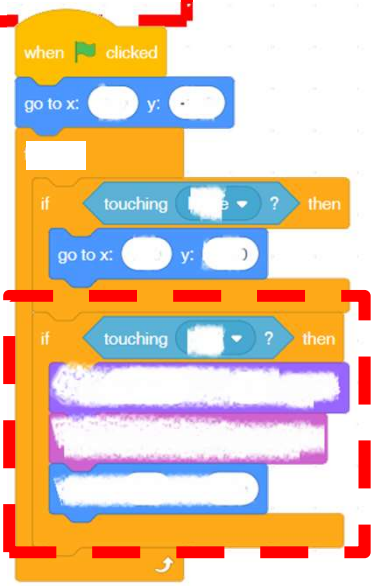


Creating a Maze Game

To Code: Touching Goal

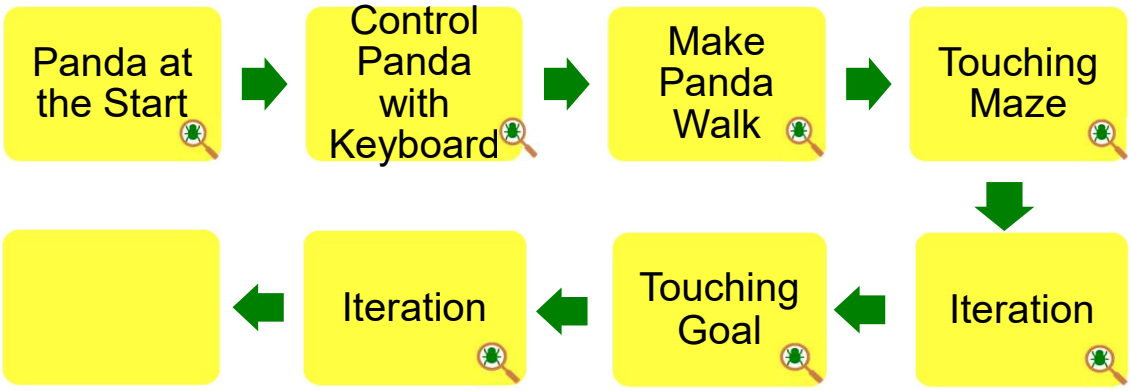
If Panda touching the Goal (Bamboo):





Testing and Debugging

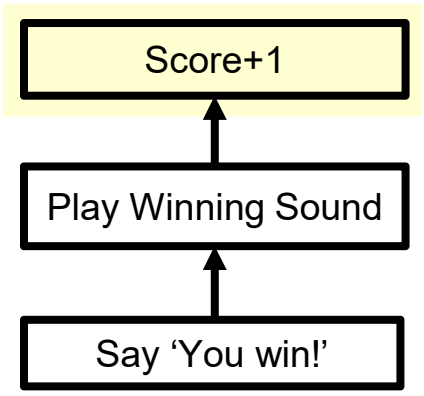
Try you best to move the Panda to the bamboo.
Don't forget to check the conditions constantly.

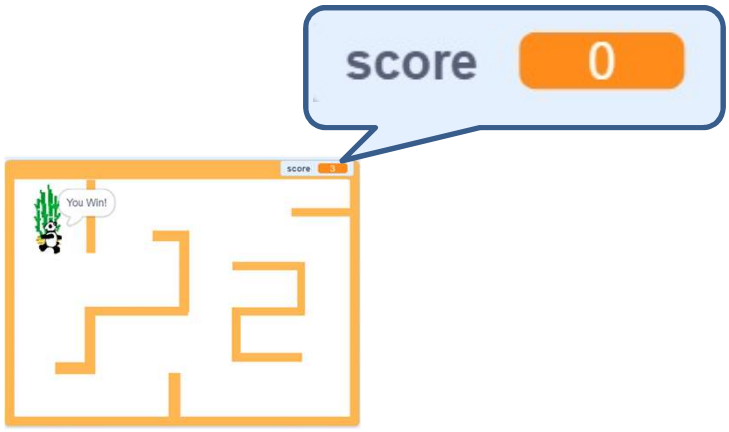


Creating a Maze Game

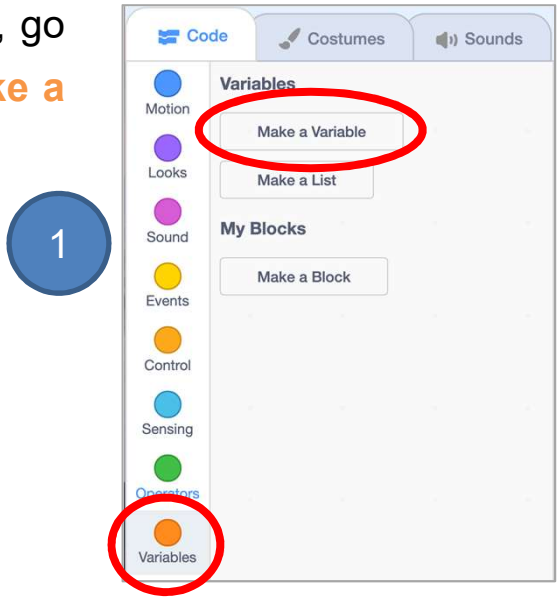
To Think and To Code: Add Score

Do you still remember what we get if the panda finds the bamboo? Yes! Now it's time to add score when you win.

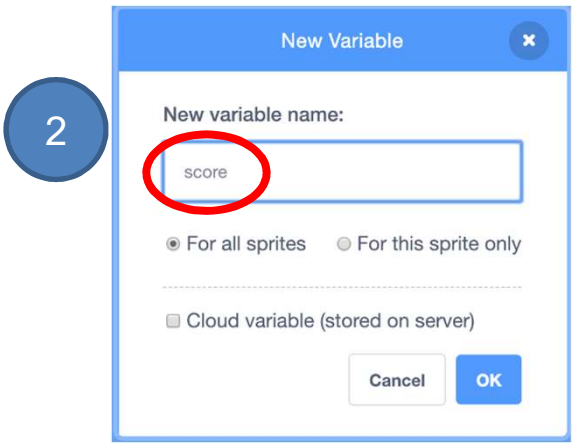




1. To add a variable for recording the score, go to the **"Variables"** drawer and click **"Make a Variable"**.



2. Name the new variable as **"score".**



Creating a Maze Game

Unplugged Activity: Variables

	Teacher takes out a box and writes “candy” on the box

Game 1: You can choose one of them, then teacher will do the action.

1. Set candy to “ 5”
2. Change candy by “-2”
3. Change candy by “2”

Game 2: Guess how many candies in the box:

		
☐	☐	☐

Creating a Maze Game

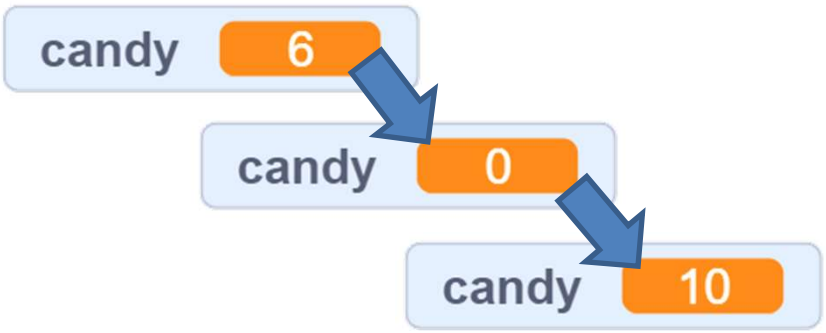
Knowledge Builds up: Variables

Variables are used to store values. Variables have the following properties.

Fill in the blank:

1. Variables have _____.

2. A variable can only store _____ value at a time.



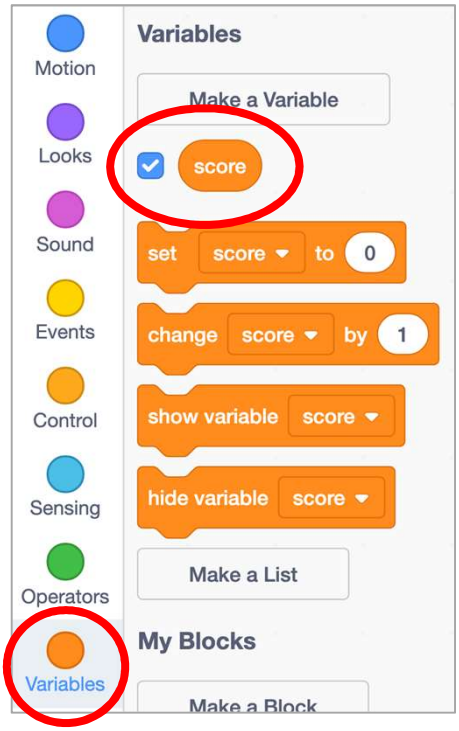
3. The value of the variable can be _____.

Creating a Maze Game

To Code: Add Score

See Appendix P.38

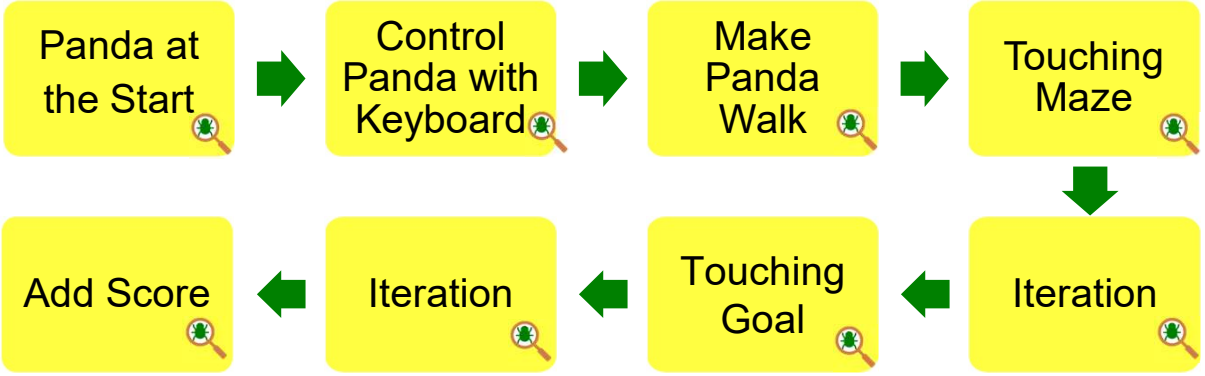
1. First of all, we need to show variable score (tick the score to show) and reset the score to 0 when the game starts. Hint: Check out the "Variables" Drawer to see which block should be added for resetting the score.



2. When the Panda sprite reaches the bamboo, what should the score change?

Testing and Debugging

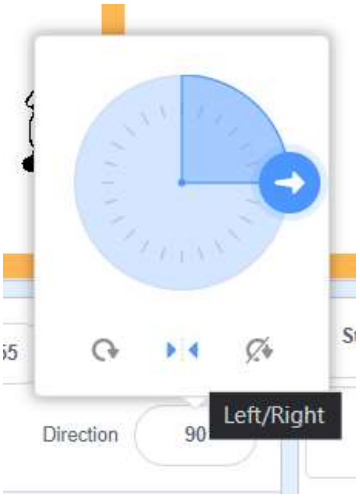
You did it! Play your maze game again to see everything works fine.

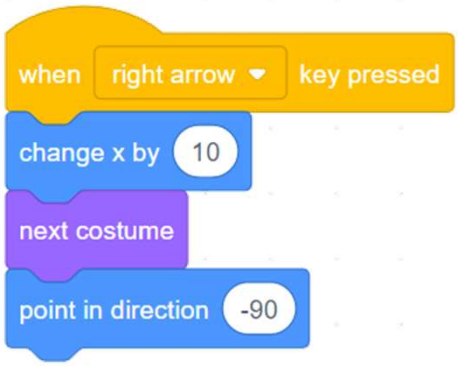


Creating a Maze Game

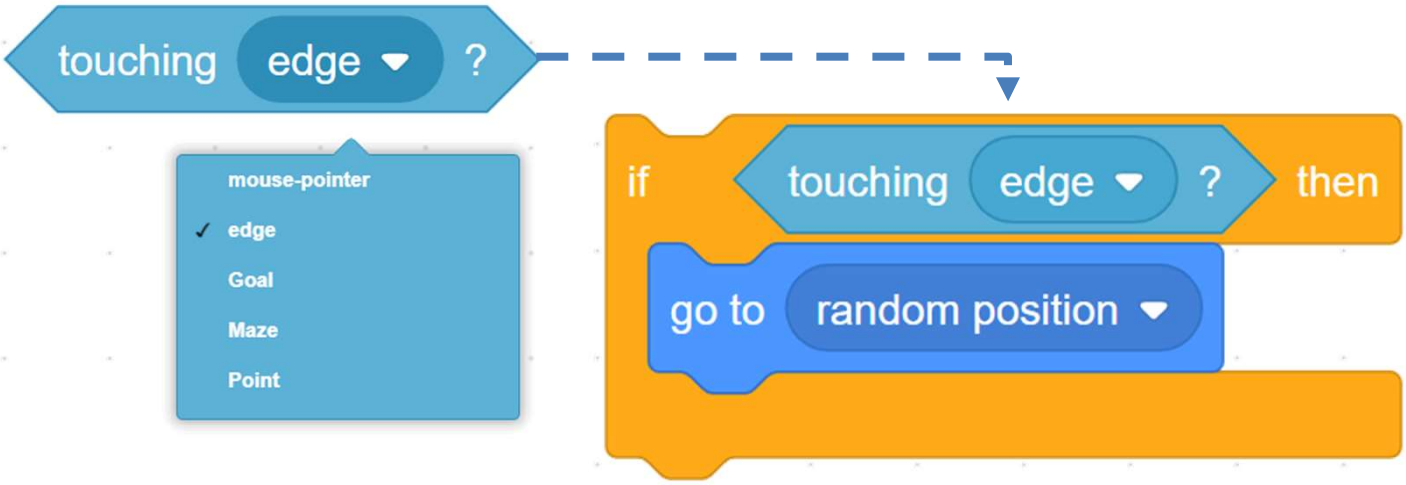
Program codes for reference:

Changing the direction of the panda's face when walking backwards:





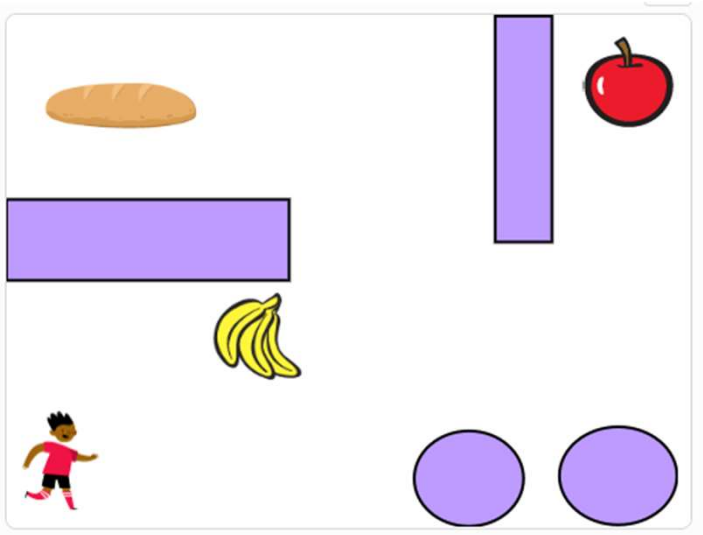
Adding edge / bouncing obstacles:



Creating a Maze Game

To Create: Supermarket Maze

1. With your creativity, you are free to create another maze game, or you may take a look at the example project, Supermarket Maze.
2. There is another project similar to the maze game we just learnt: Control the boy sprite to find the bread in the supermarket maze. if the boy is “touching” the bread, then he says, “I find it!”
   Example: Supermarket Maze https://scratch.mit.edu/projects/731275755/



Write down your idea:

(e.g.) If the boy is touching banana  then go back to the starting point.

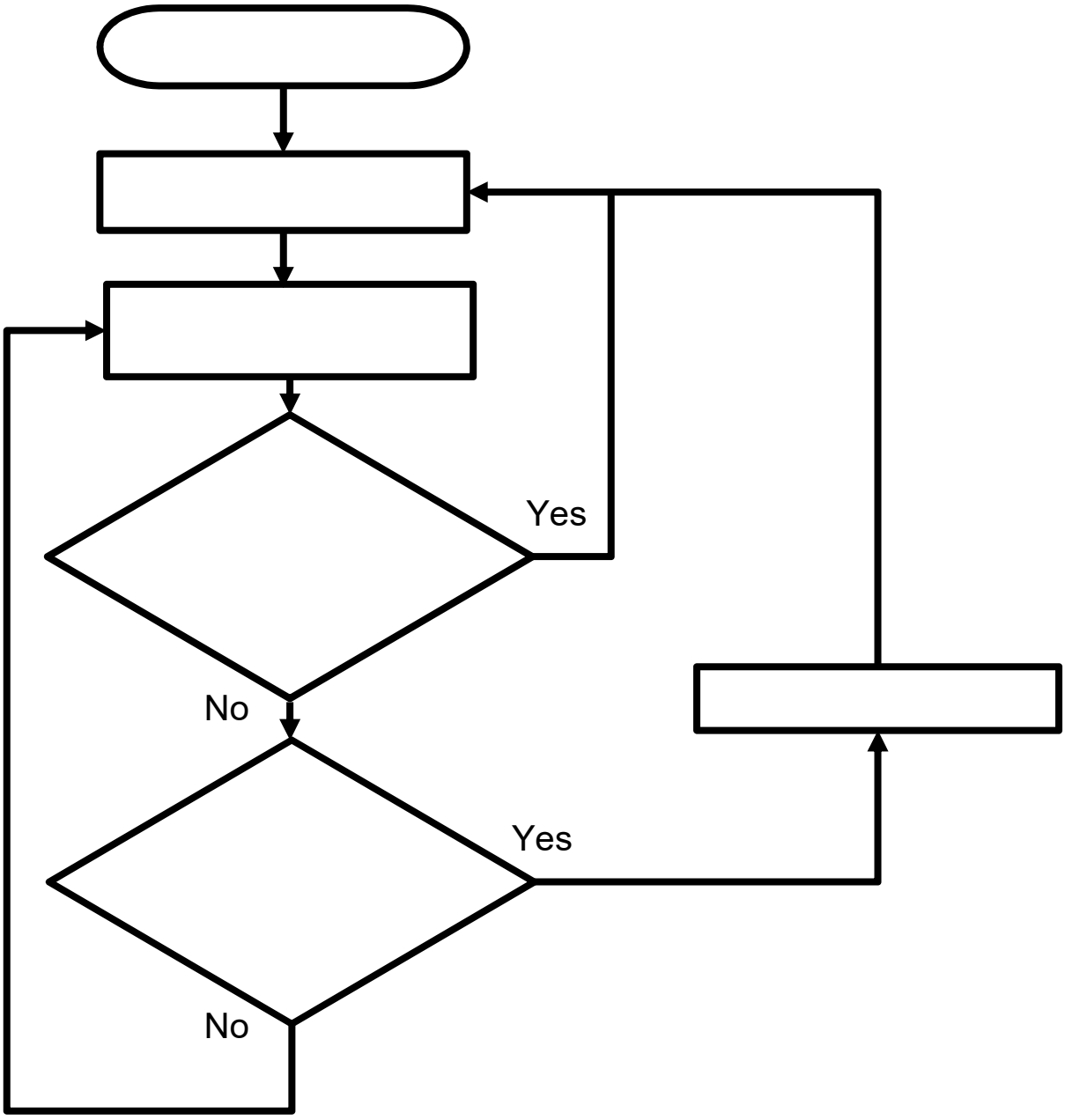
1. If the boy is touching_____, then _____.

2. If the boy is touching_____, then _____.

Creating a Maze Game

To Think and To Code

Based on what you have learnt in this unit, think about the sequence and algorithm design of your own Maze Game. Fill in the blank to complete the flow diagram .



With the flow diagram you design, program it to achieve your idea!

Creating a Maze Game

To Reflect: Two Stars and a Wish Worksheet

Name of Project: _____ Name of Creator: _____

Please write down two things that you like about this project.

1

_____

2

_____

What is one thing you would like to add or change to make this project better?

_____

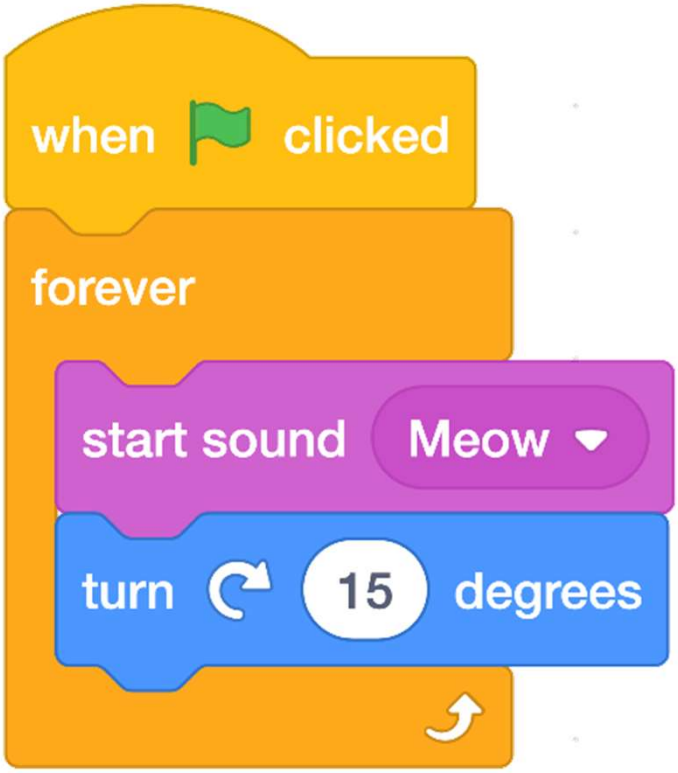
_____

_____

Creating a Maze Game

Review Questions

1. What happens to the cat when you click the green flag?

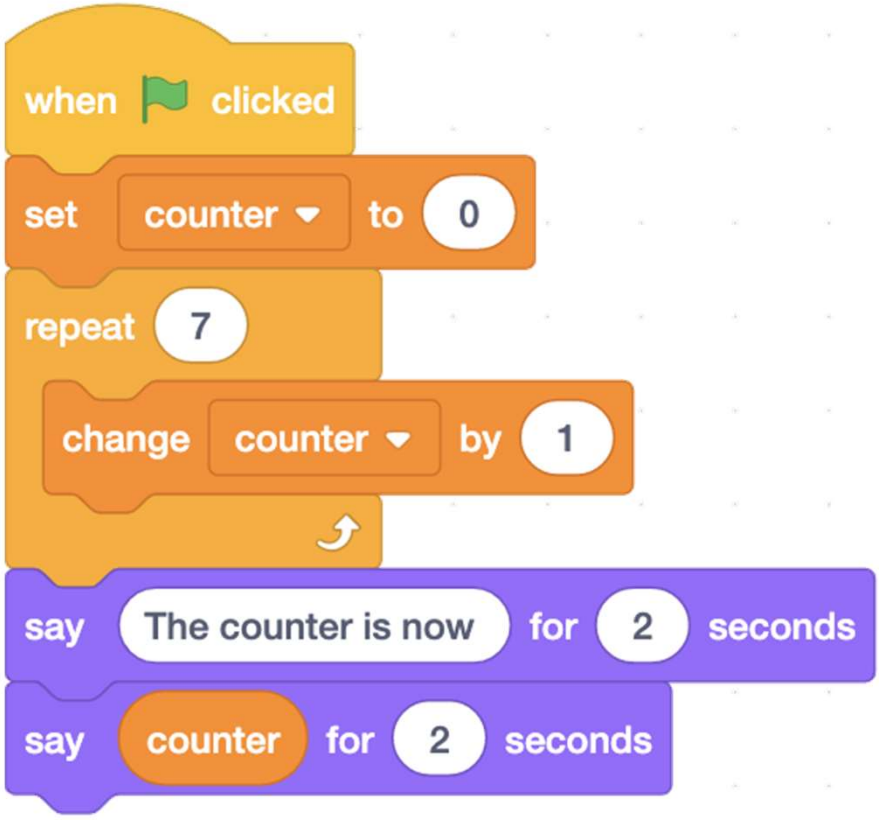


A. The cat meows once and turns 15 degrees once.
B. The cat meows repeatedly forever but never turns.
C. The cat meows once but then turns 15 degrees repeatedly (spins clockwise) without meowing.
D. The cat meows repeatedly while turning 15 degrees (spins clockwise) repeatedly.

Creating a Maze Game

Review Questions

2. What happens when the code blocks below run?

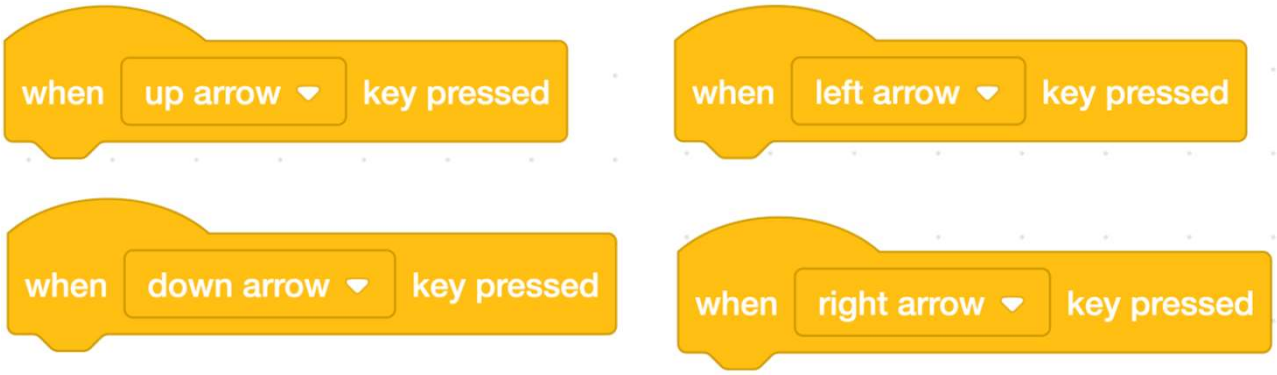


A. The Sprite says “The counter is now” and then says“1”, “2”, “3”, “4”, “5”, “6”, and “7” for 2 seconds.
B. The Sprite says “The counter is now” and then says “7” for 2 seconds.
C. The Sprite says “The counter is now” and then says “counter”.
D. The Sprite says “The counter is now” and then says “7” seven times for 2 seconds.

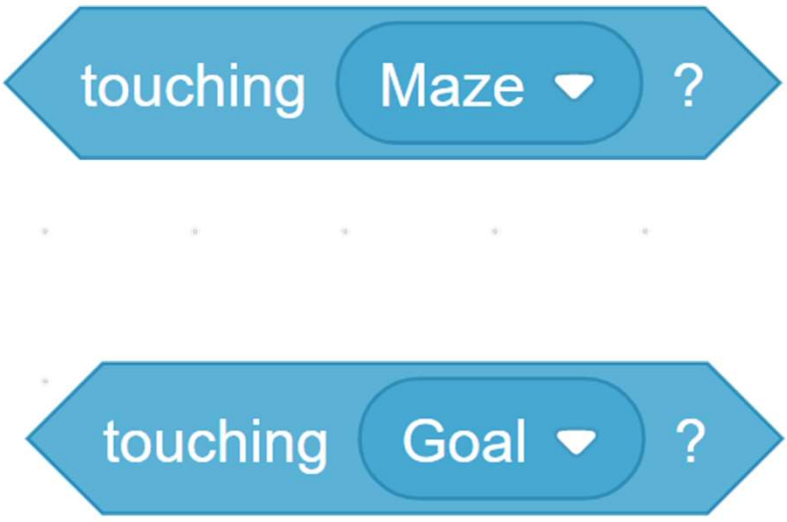
Creating a Maze Game

Revision on Key Features

Key pressed events:



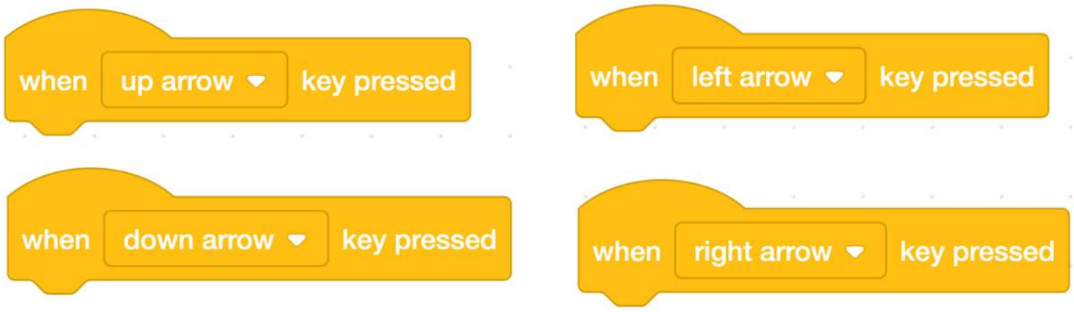
“Touching” blocks:



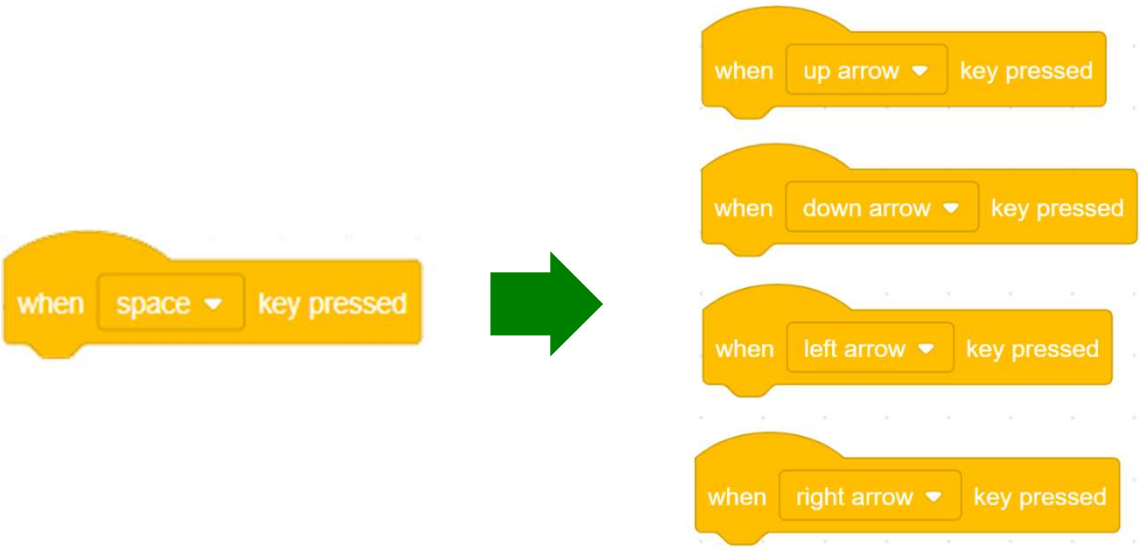
Creating a Maze Game

Revision on Key Concepts & Practices

Events: We use event blocks (keyboard events) to trigger Scratch to take actions (sprite movement).



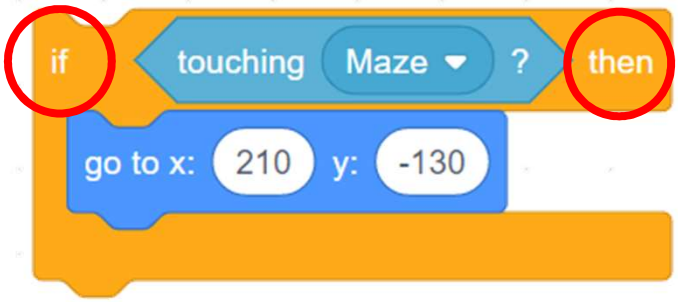
Reuse and Remix programs/codes: The reuse and remix of the works of other programmers are crucial in the online communities of Scratch. We can reuse and remix the codes of one keyboard event and use them for the rest of arrow key pressed events.



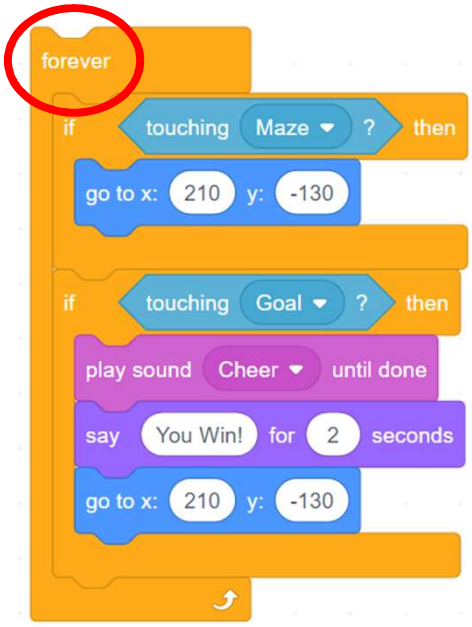
Creating a Maze Game

Revision on Key Concepts & Practices

Branching/Selection: We use conditional statements in programming to enable computers to make decisions. Conditionals always have an "If" part, which tells the program in the "Then" part what to do when the condition is true.



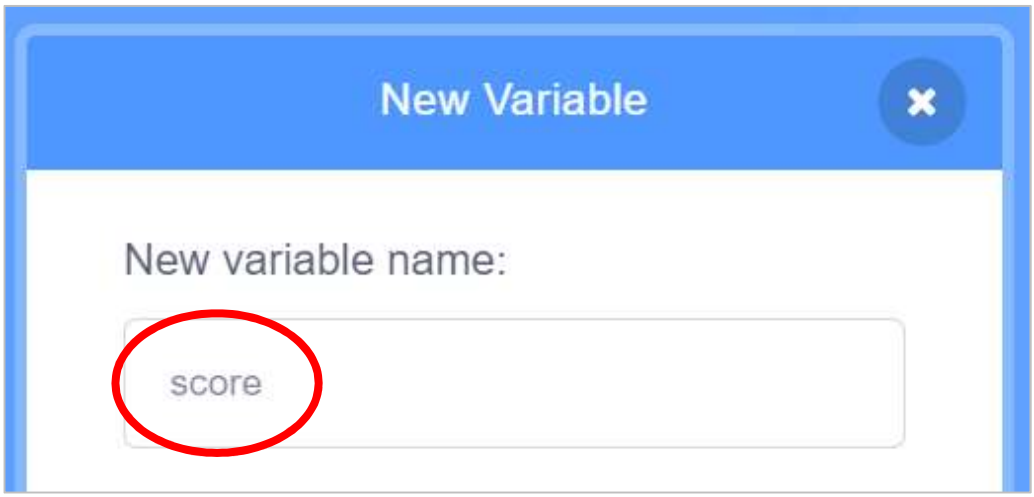
Iteration - Forever: Iteration is repeating a process in order to produce a sequence of outcomes. Forever blocks can trigger iteration in Scratch.

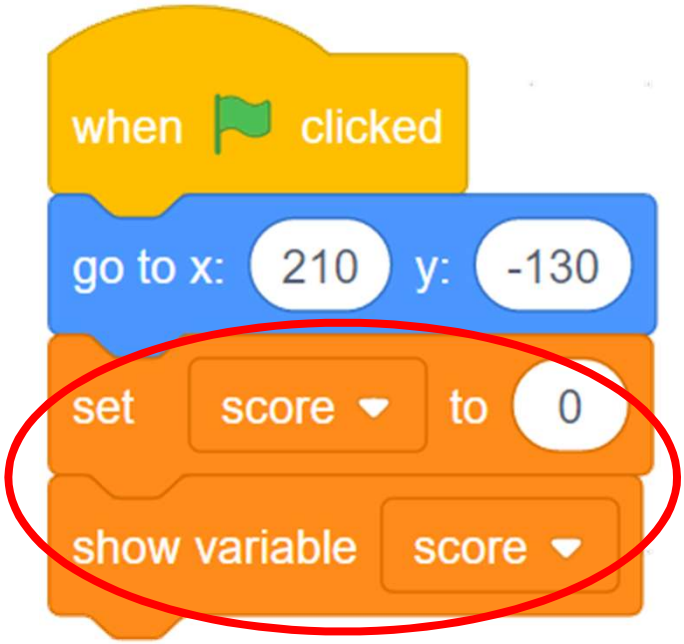


Creating a Maze Game

Revision on Key Concepts & Practices

Variables: In programming, variables are used to store values. It has a name, can only store one value at a time and be updated. For example, We can create a variable called “score" to store the score of a game.

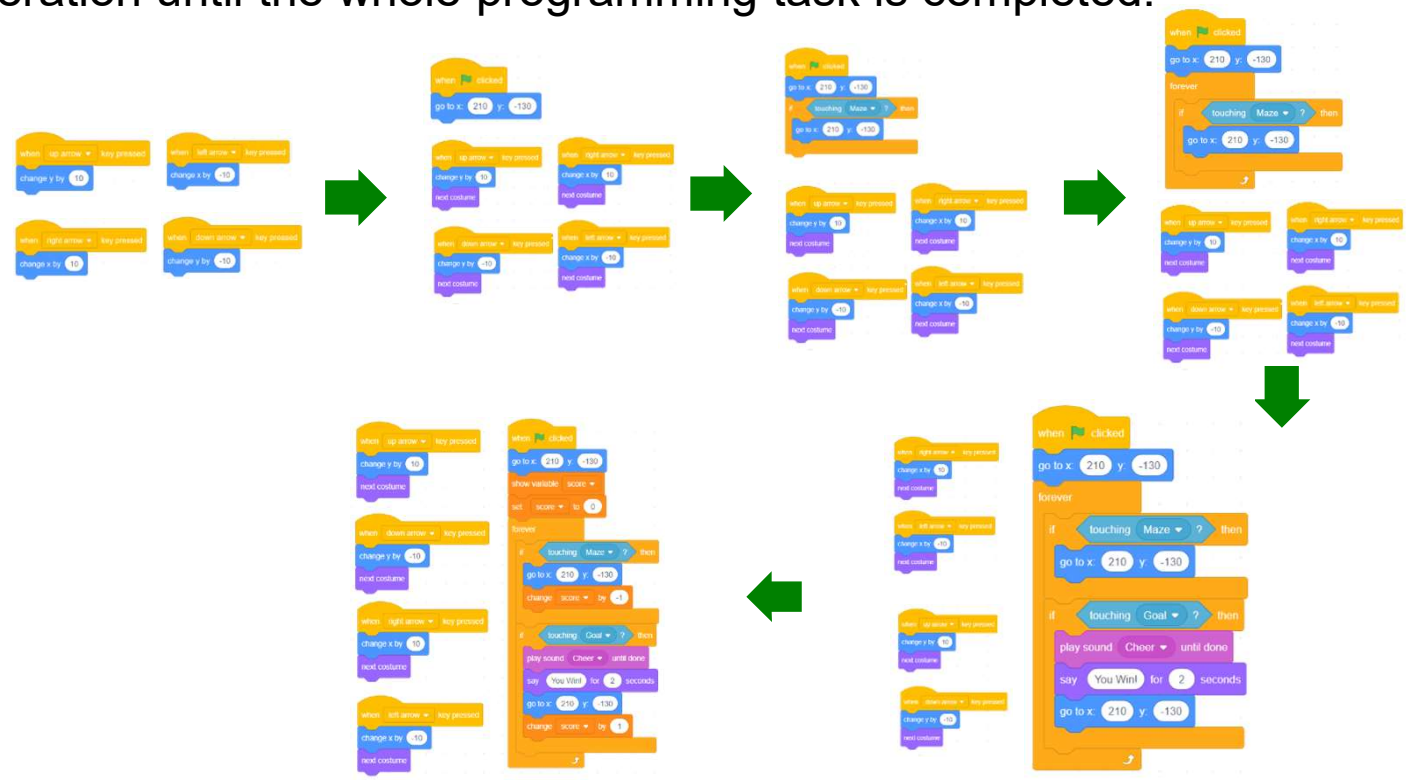




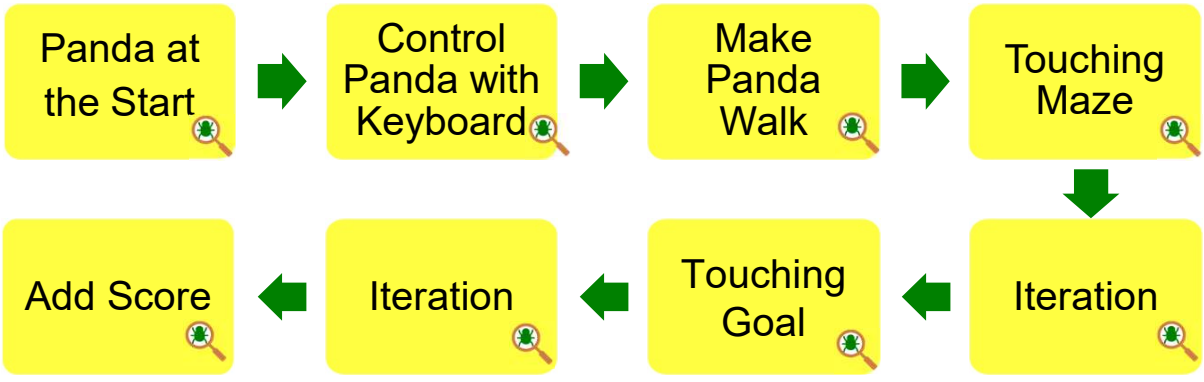
Creating a Maze Game

Revision on Key Concepts & Practices

Being incremental and iterative: to work out a sub-task as an iteration, try it out, then work out another sub-task in one more iteration until the whole programming task is completed.



Testing and Debugging: Testing a computer program is the process of checking if it can produce results as designed. Debugging a computer program is the process of finding out ways to revise the program so that the bugs can be removed.



Appendix

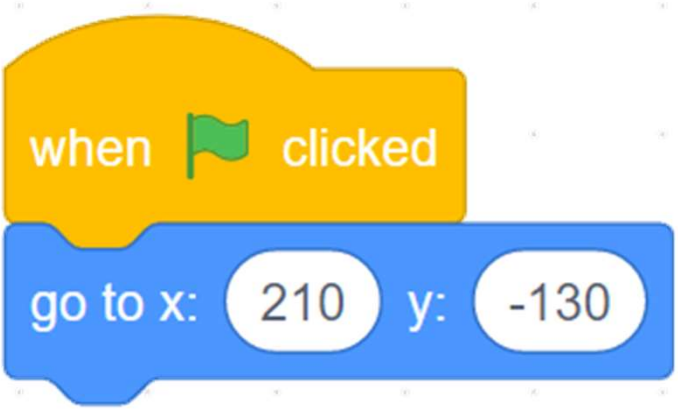
Operation Manual

Creating a Maze Game

To Code: Panda at the Start

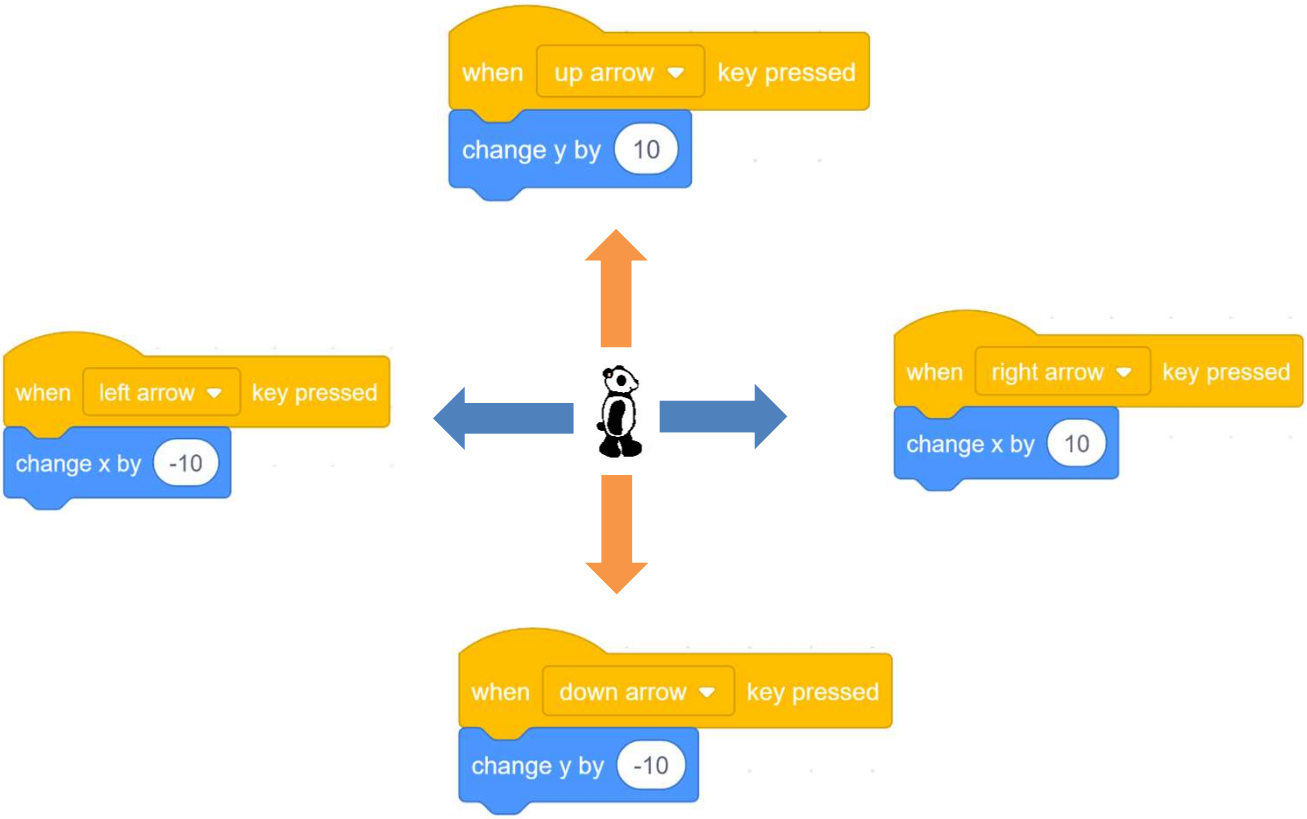
See Student Guide P.5

Make your sprite go to your desired starting coordinates when the green flag is clicked.



To Code: Control Panda with Keyboard

See Student Guide P.6

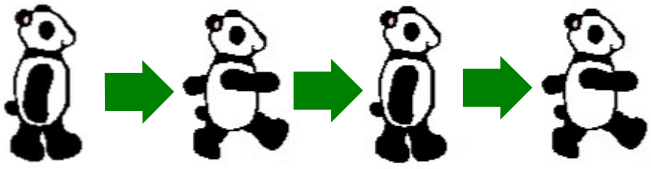


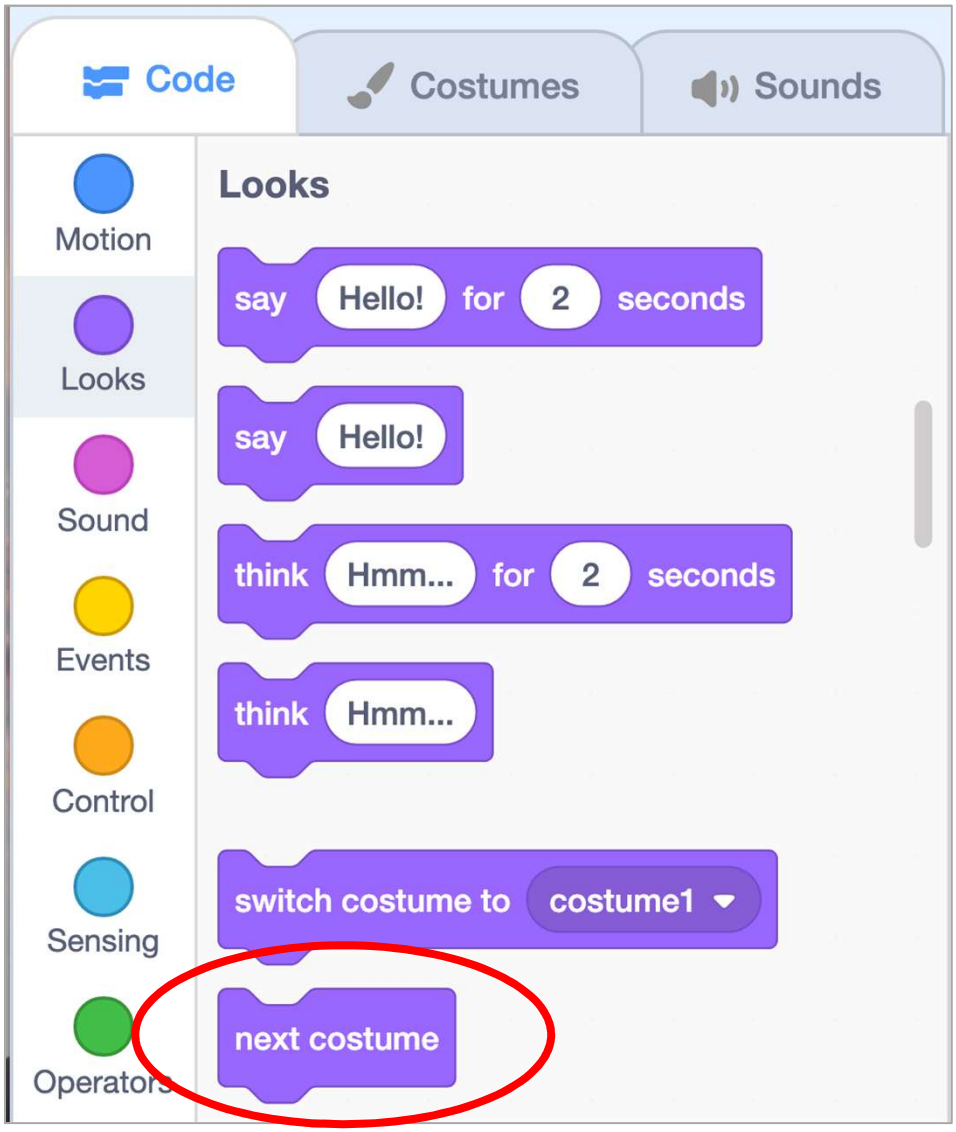
Creating a Maze Game

To Think and To Code: Make Panda Walk

Use the "**next costume**" block to make the panda look like it is walking.

See Student Guide P.7



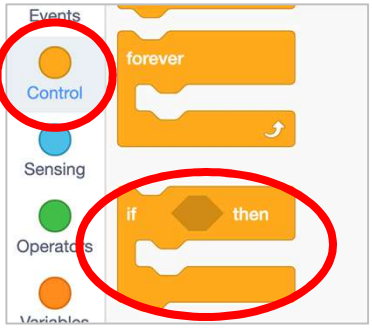


Creating a Maze Game

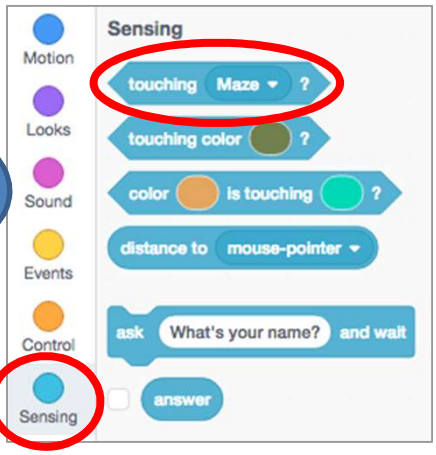
To Code: Touching Maze

See Student Guide P.12

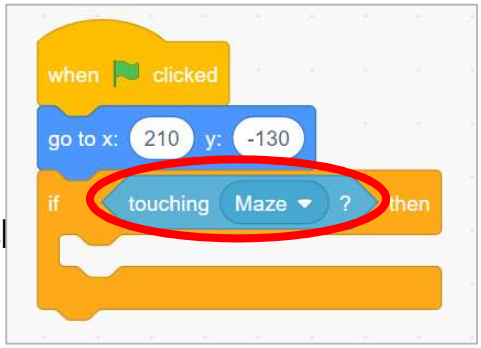
1. In this game, we want to send the Panda sprite to the starting point if it touches the walls of the maze. To do this, pull out an "**if-then**" block from the "**Control**" drawer to complete this task.



2. We want the Panda sprite to go to the start if it touches the Maze sprite, so also pull out a "**touching Maze**" block from the "**Sensing**" drawer.



3. Put the "**touching Maze**" block in the empty slot of the " **if-then**" block and put that combined block under the "**when green flag clicked**" block.



4. Copy and paste the "**go to x:# y:#**" block inside the "**if-then**" block.

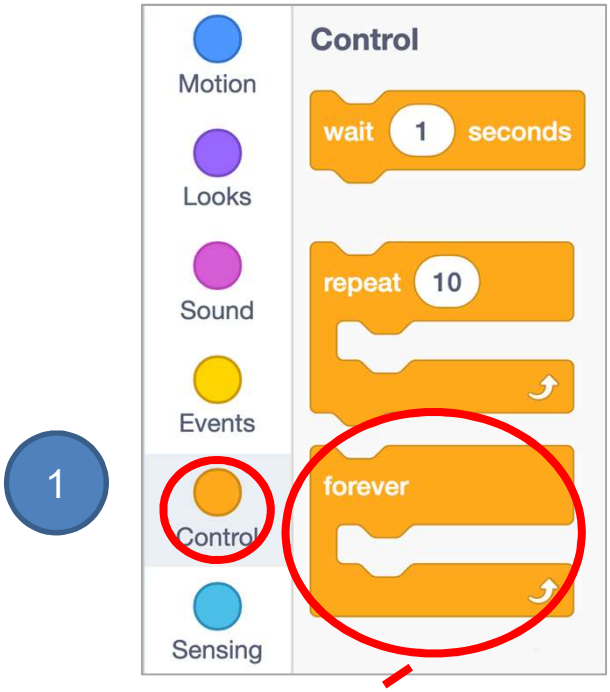
Creating a Maze Game

To Think and To Code: Iteration

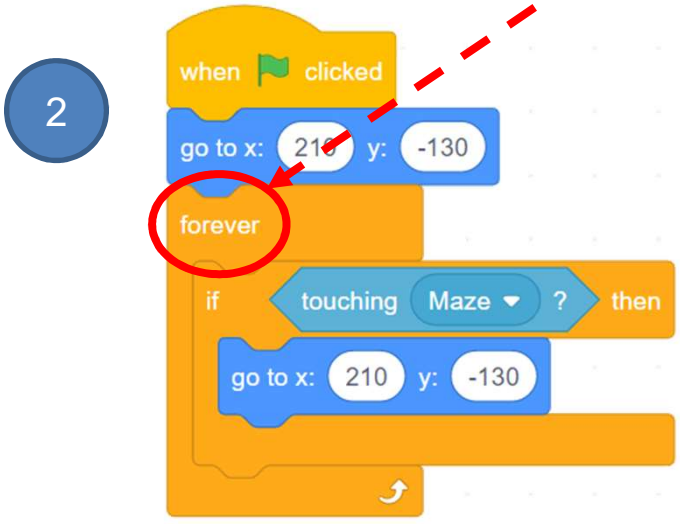
See Student Guide P.14

The "**forever**" block allows the program to constantly check for the conditions you are testing.

1. Add the "**forever**" block from the "**Control**" drawer.



2. Put the **if touching Maze** block inside the **forever** block.



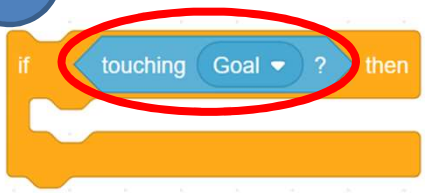
Creating a Maze Game

To Code: Touching Goal

See Student Guide P.15

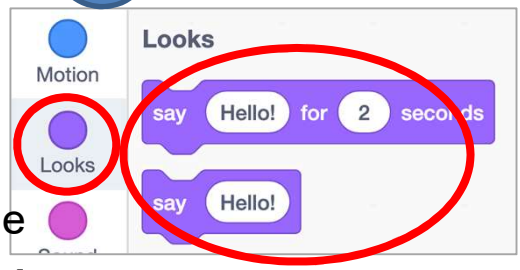
1. Add the appropriate additional "**if-then**" block if the Panda sprite touches the goal, it says "You win!" and plays a sound.

1



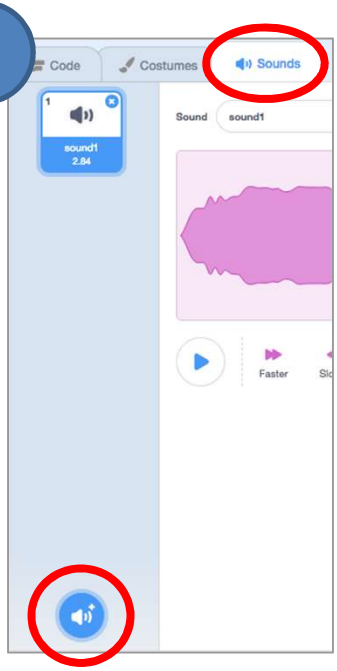
2. To make your sprite say something, go to the "**Looks**" drawer.

2



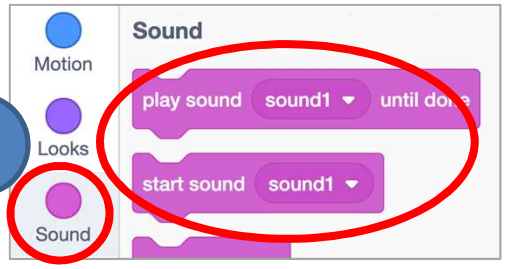
3. To add a sound to your project, go to the "**Sounds**" tab. Click on the **Choose a Sound** icon at the bottom left to select a sound.

3



4. To make your sprite play the sound you found, go to the "**Sound**" drawer of "**Code**" tab.

4



5. Finally send the Panda sprite back to its starting position.

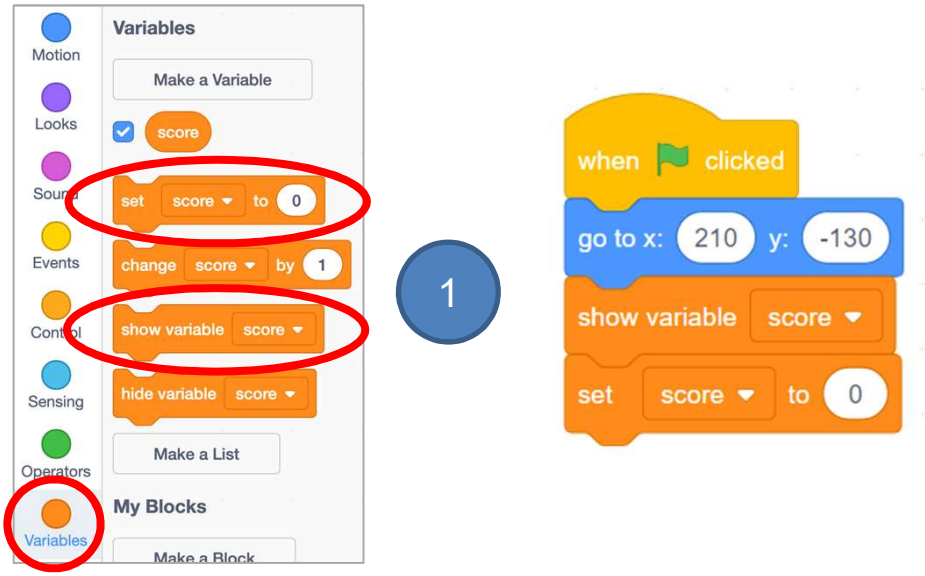
5

Creating a Maze Game

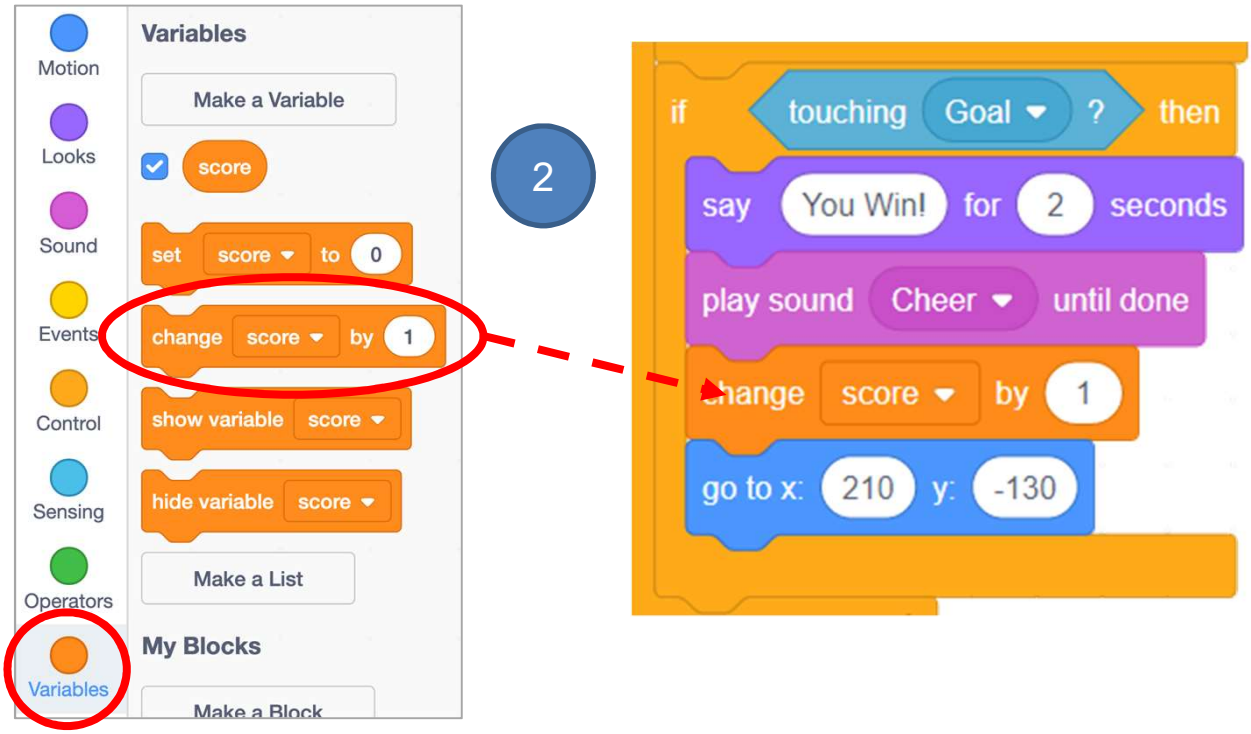
To Code: Add Score

See Student Guide P.20

1. Drag “**set score to 0**” and “**show variable score**” blocks to “**when the green flag is clicked**” block.



2. Change the score by 1 when the Panda sprite reaches the bamboo. Put “**change score by 1**” under “**if touching goal**” block.



Unit 6: Creating a Maze Game with micro:bit (Extension Unit) Student Guide

Content

Creating a Maze Game with micro:bit

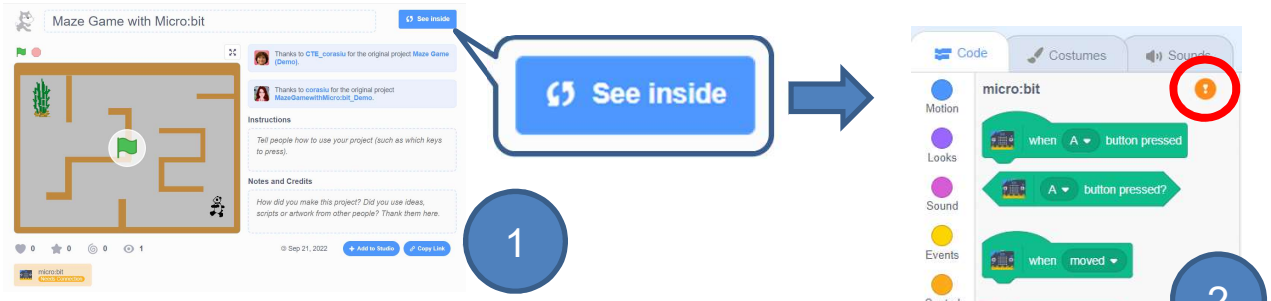
Let's create a maze game with micro:bit to control the sprite.

This game is similar to Unit 5. However, besides keyboard, you will control the panda sprite with a joystick (with micro:bit inside) this time.

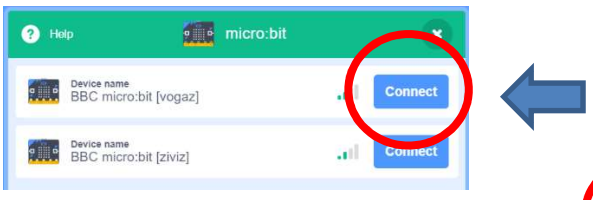
To Play

Prepare your computer, make sure you have Scratch Link installed and running, and bluetooth is on. (Teacher may also provide you a joystick with micro:bit inside to play with).

1. Open https://scratch.mit.edu/projects/734787236/ and click **"See inside".**



2. Click the orange exclamation mark and connect your micro:bit to Scratch.



3. Play the maze game with joystick, is it easier to control and find the bamboo? 3

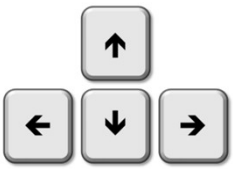
Creating a Maze Game with micro:bit

To Think

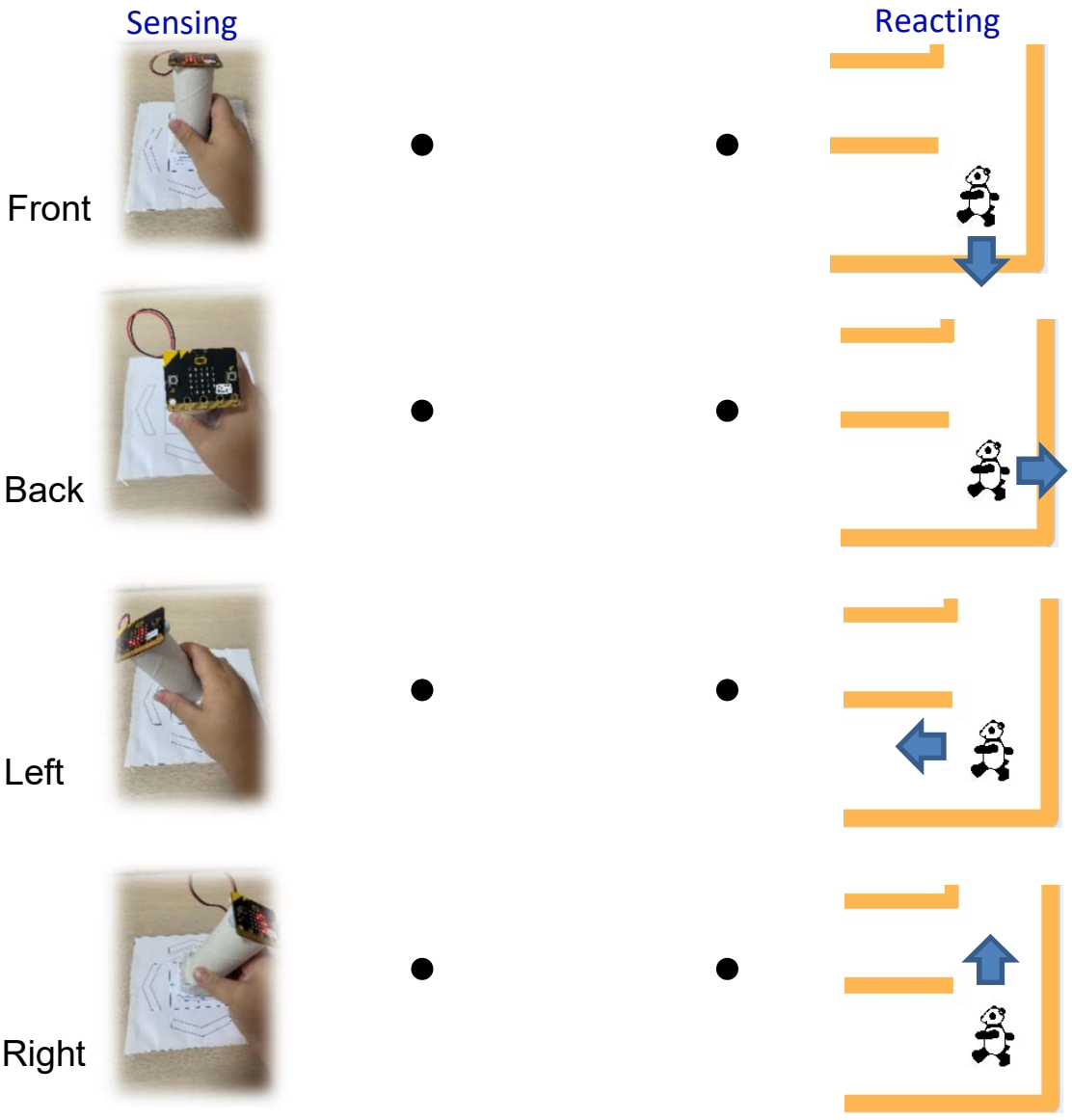
In this unit, the Maze Game looks a bit different. Mark the changes as follows:

1. How can you control the panda sprite this time? (You may tick ☑ more than one box.)

❑ Keyboard ❑ Micro:bit ❑ Mouse



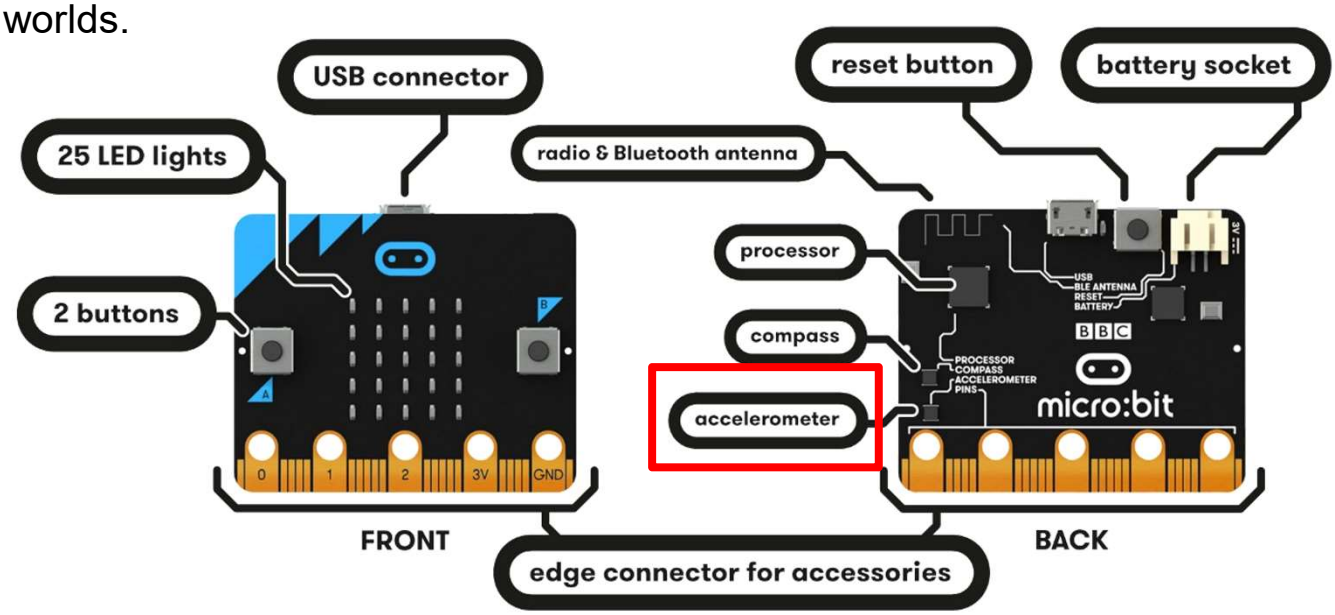
2. What will the panda sprite react when the hand-made joystick moves to the following directions? Match the sensing and reacting photos.



Creating a Maze Game with micro:bit

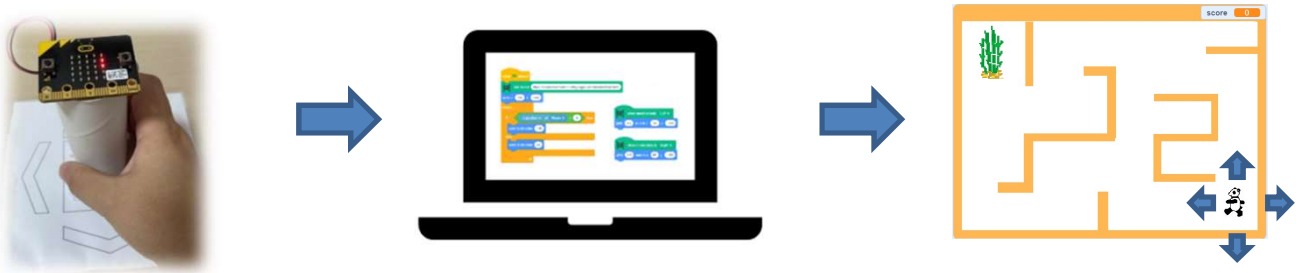
To Learn: micro:bit

Here we can see all the micro:bit's features, including an LED display, 2 buttons, and a motion sensor (accelerometer). You can connect it to Scratch and build creative projects that combine the magic of the digital and physical worlds.



In this unit, we will make use of micro:bit's feature – accelerometer, together with programming, to create a joystick to control the panda.

Knowledge builds up: Internet of Things (IoT)
Sensing-Reasoning-Reacting



Sensing	Reasoning	Reacting
Collect data from the sensors.	Judge and make decisions based on the data.	Respond according to the result of reasoning.

Creating a Maze Game with micro:bit

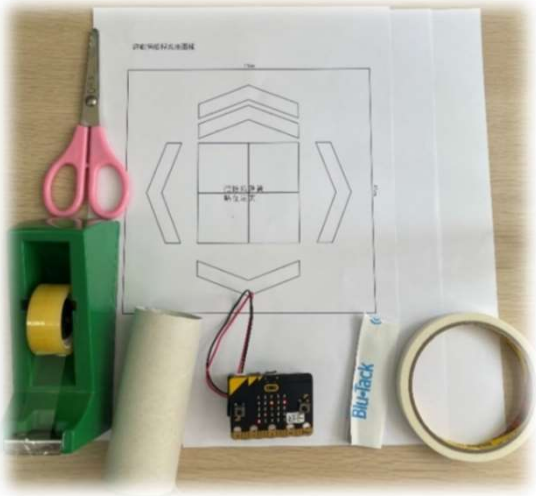
Pre-Lesson Task (Make the Joystick at Home)

You can try the followings to make a joystick model:

1. Prepare the following handicraft materials:

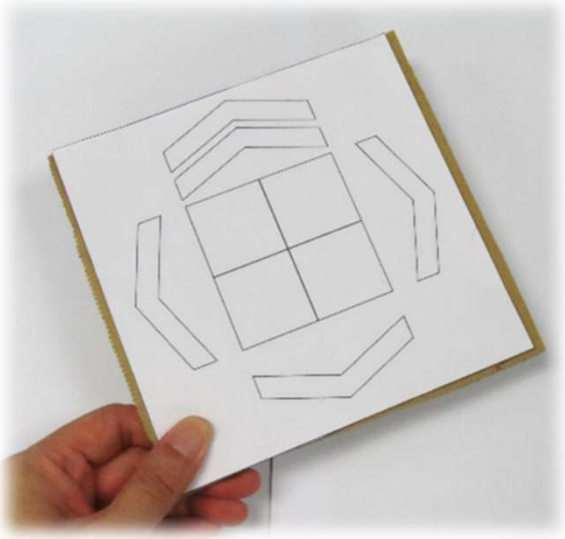
- Power-on micro:bit
- Toilet paper roll
- 2 pieces of A4 paper
- Paper pattern of base on Student Guide
- Cello tape
- Double-sided tape
- Scissors
- Blu-Tack

1



2. Print and cut out the Base Pattern of Joystick in next page (you can stick it to a cardboard with similar size, or simply stick on the table during the lesson at school).

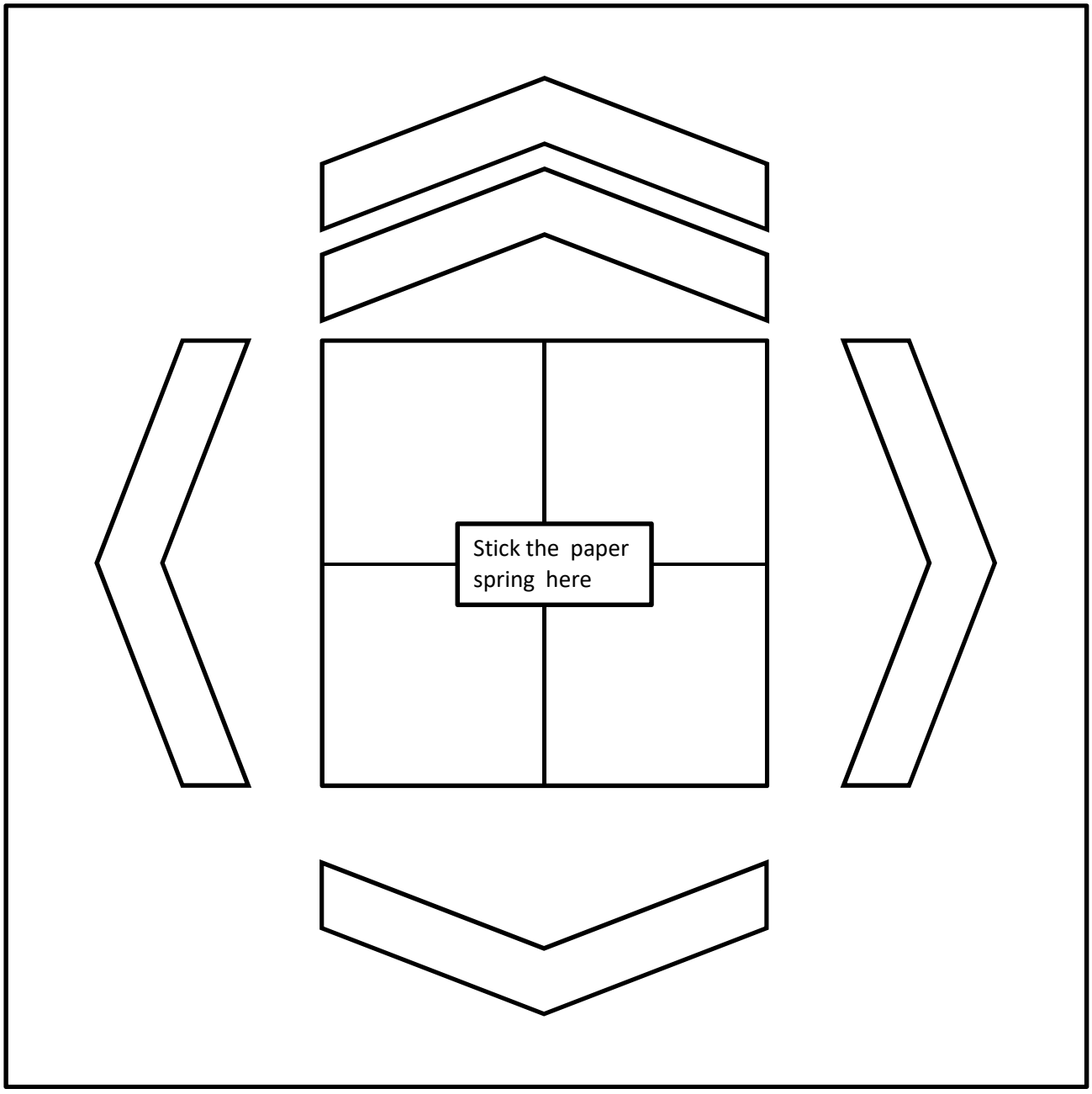
2



Creating a Maze Game with micro:bit

Pre-Lesson Task (Make the Joystick at Home)

Base Pattern of Joystick:

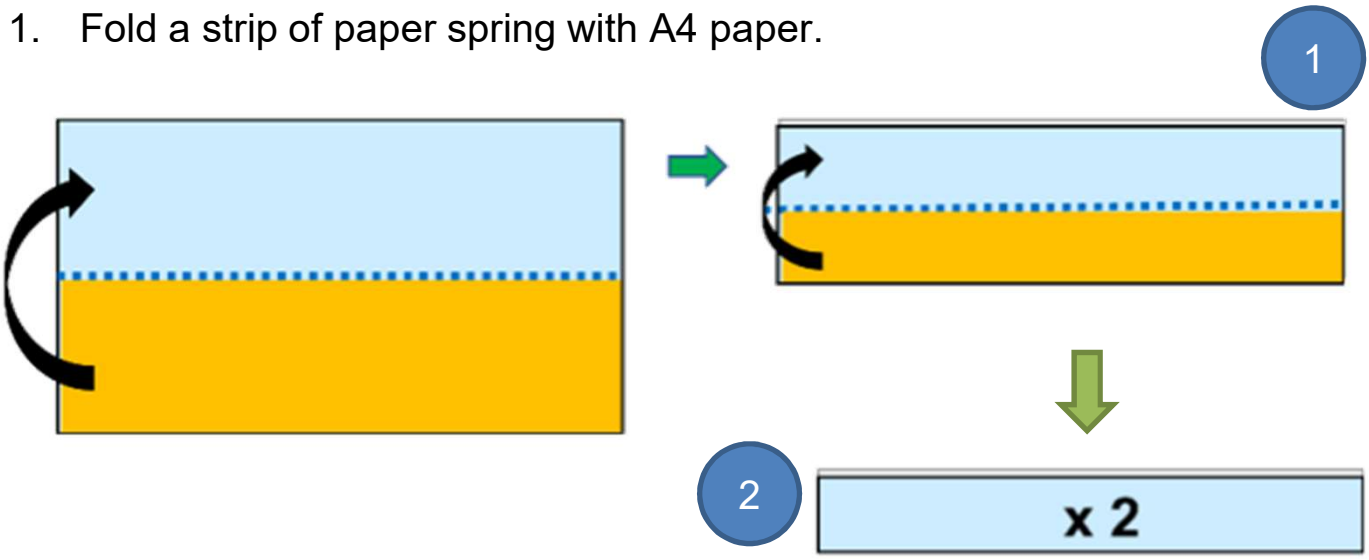


Blank Page

Creating a Maze Game with micro:bit

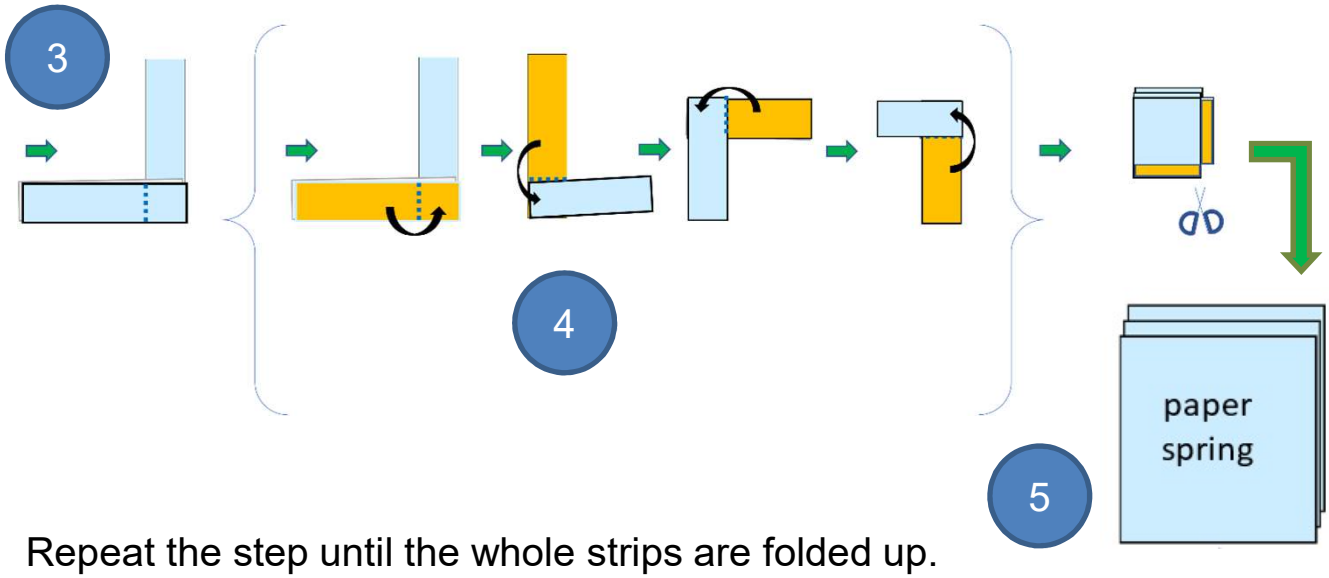
Pre-Lesson Task (Make the Joystick at Home)

1. Fold a strip of paper spring with A4 paper.



2. Fold two pieces of A4 paper in half vertically twice to form two thick strips.

3. Hold one strip horizontally with left hand and put another one to fix on the end of the horizontal strip with right hand.



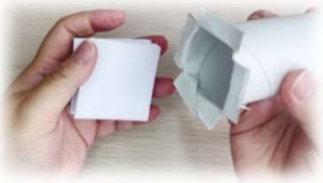
4. Repeat the step until the whole strips are folded up.

5. Cut off the excess parts and fix the ends with double-sided tape to form a paper spring.

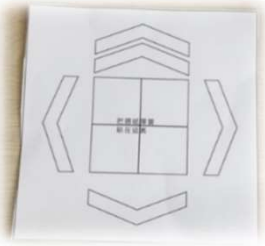
Creating a Maze Game with micro:bit

Pre-Lesson Task (Make the Joystick at Home)

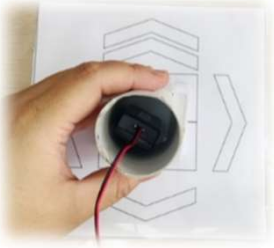
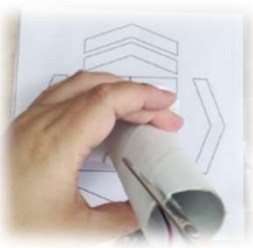
1. Cut four gaps about 1 cm long with similar distance at one end of the toilet paper tube with scissors; fold the gaps out and fix them with the paper spring with two-sided tape.



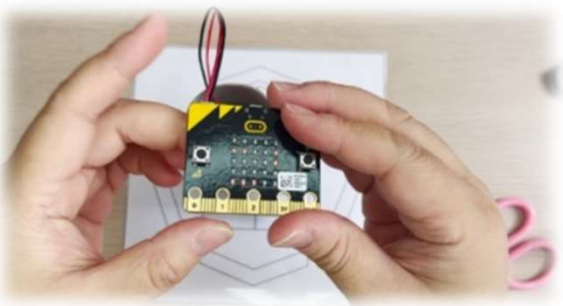

2. Fix another end of the paper spring to the centre of the paper base pattern with two-sided tape.



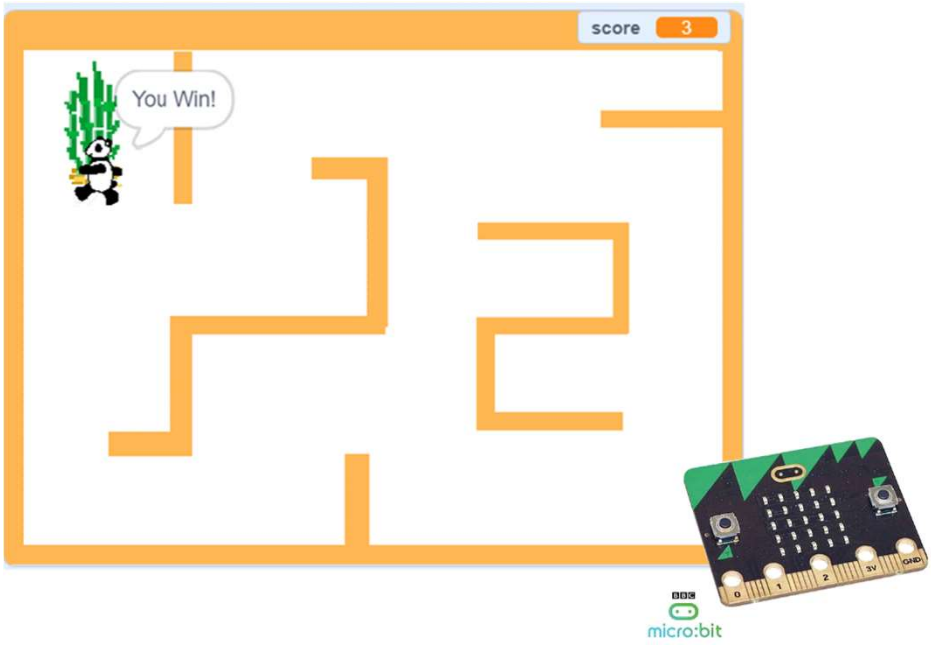

3. When you get the micro:bit from teacher (or you may complete last few steps at the class), put the battery box of micro:bit into the toilet paper tube, and cut a 4-cm long gap to hide the wire; and then stick the micro:bit to the top of the toilet paper tube with a Blu-Tack.

4. The joystick is made!



Creating a Maze Game with micro:bit



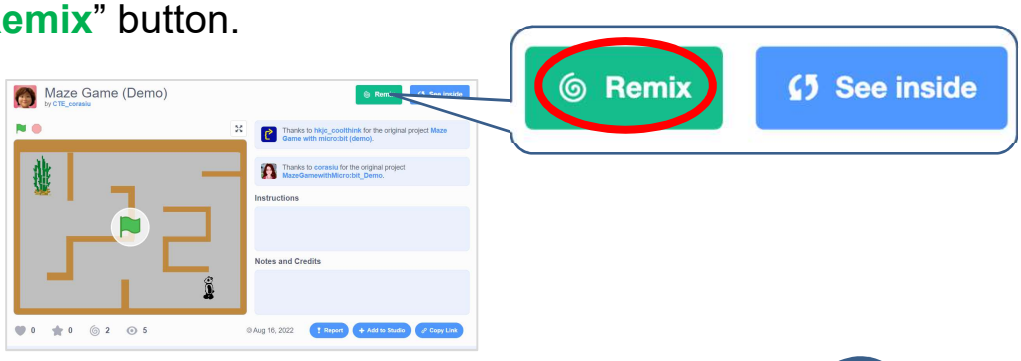
Start Here

1. Sign into your account at scratch.mit.edu.

1

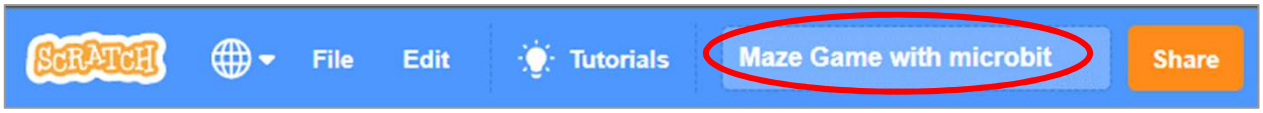
2. Open the Completed Maze Game in previous unit : https://scratch.mit.edu/projects/722154863 as the template for this unit and click on the "**Remix**" button.

2



3

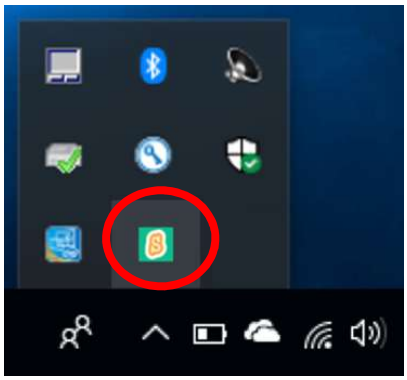
3. Rename the project as **Maze Game with microbit**.



Creating a Maze Game with micro:bit

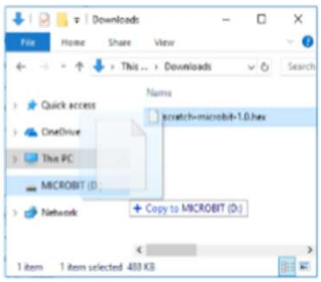
Prepare the Environment

1. To connect your micro:bit to Scratch, you need to download and install the Scratch Link software. You may find all the resources: https://scratch.mit.edu/microbit

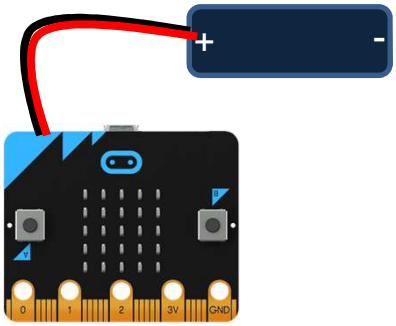


2. Start Scratch Link and make sure it is running.
   It should appear in your toolbar.

3. Download, drag and drop the Scratch micro:bit HEX file to the drive of micro:bit.



4. Connect the micro:bit with power using USB cable or battery box.

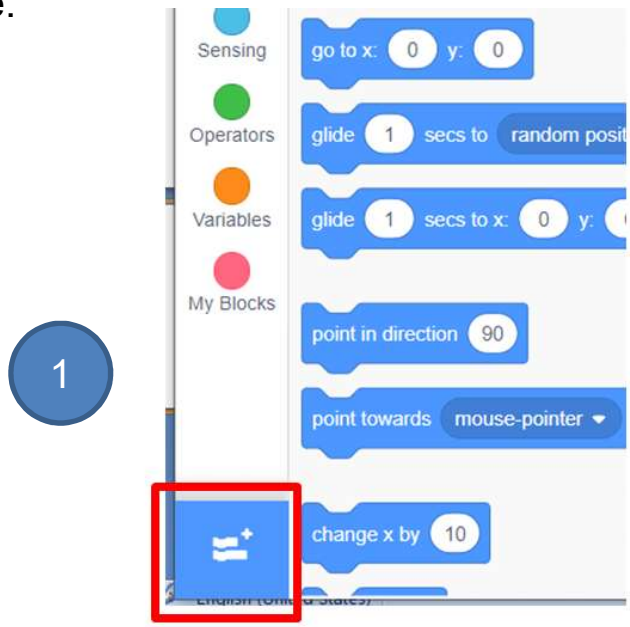


Creating a Maze Game with micro:bit

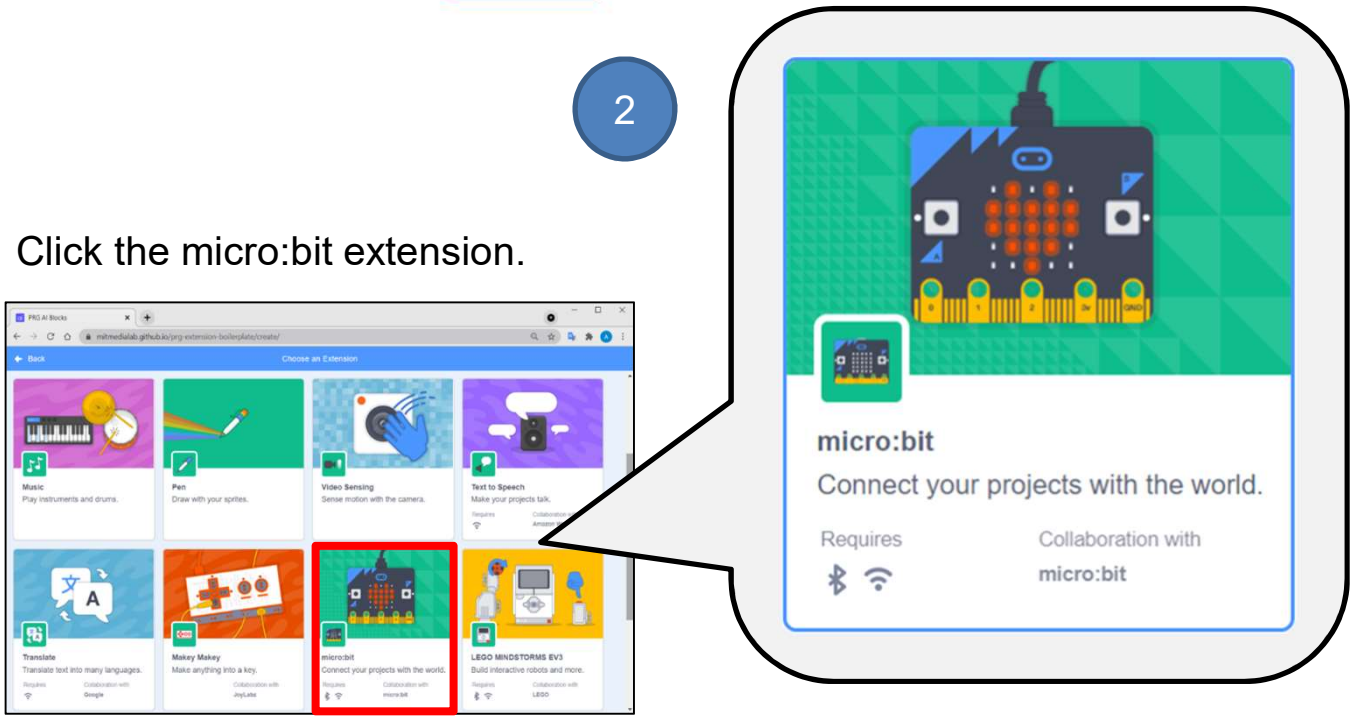
To Code: Add micro:bit Extension

Now go back to your Scratch project.

1. Click the button "**Add Extension**" on the bottom left corner of Scratch coding interface.



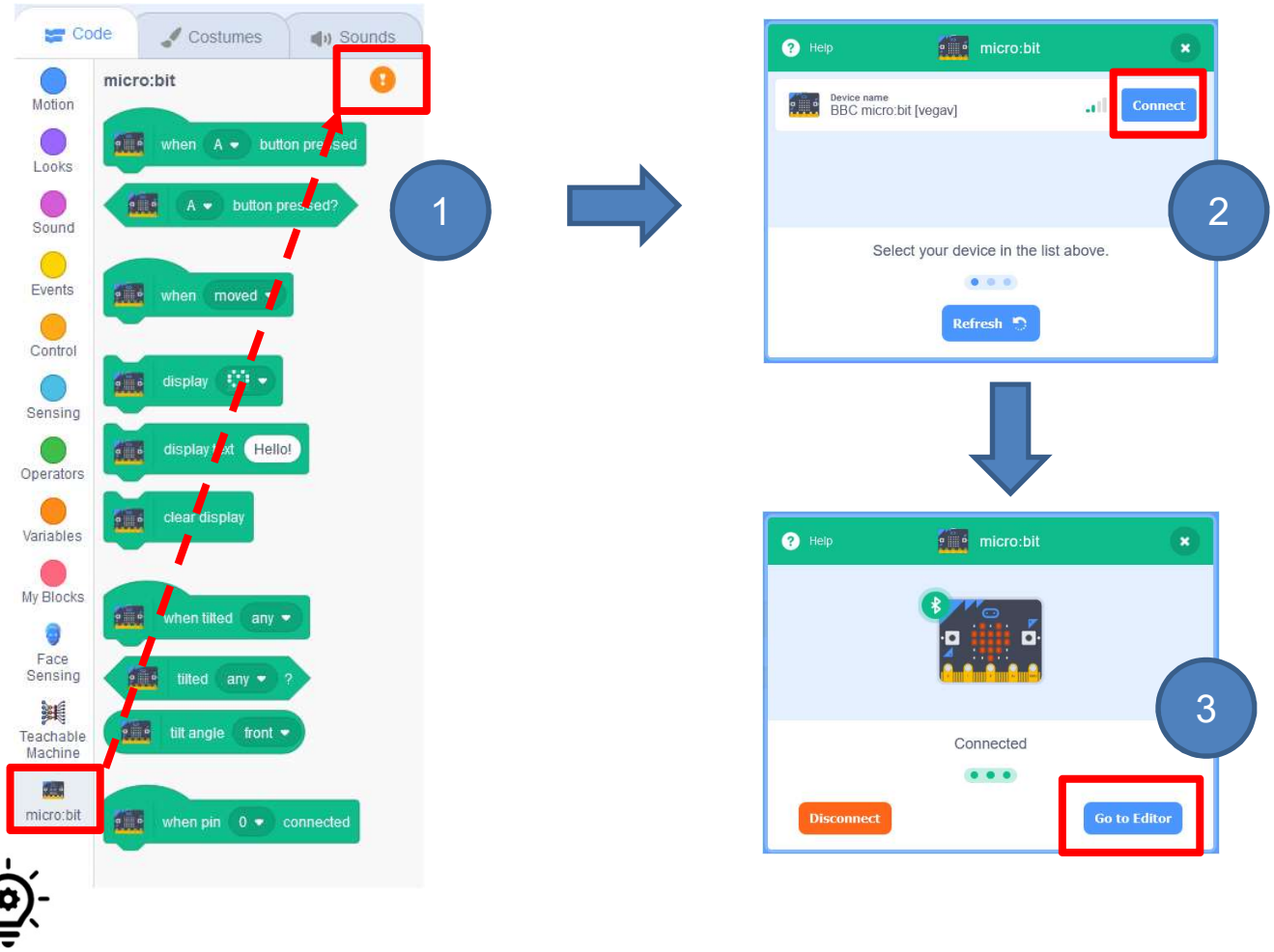
2. Click the micro:bit extension.



Creating a Maze Game with micro:bit

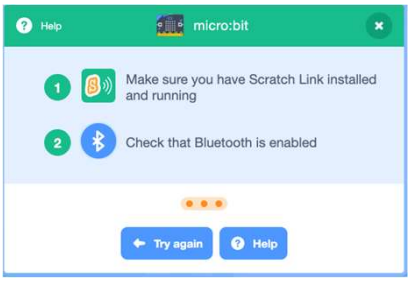
To Code: Connect micro:bit to Scratch

1. In the group "**micro:bit**" of Code tab page, click the icon (orange exclamation mark).
2. Click "**Connect**" button on the corresponding device option to connect the micro:bit.
3. Click "**Go to Editor**" button and return to the coding platform.



Knowledge builds up: Causal Reasoning

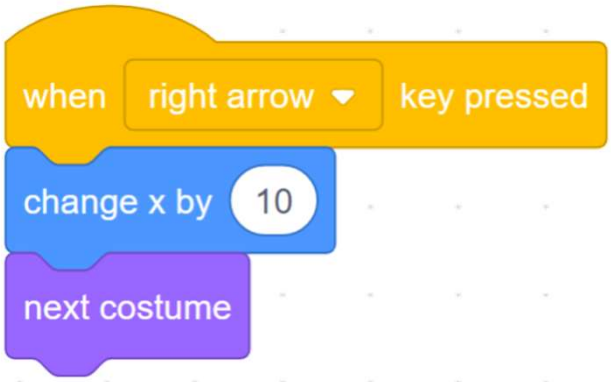
If you do not prepare the environment well or have something missing, e.g. Bluetooth turning off, low micro:bit battery, you may fail to connect micro:bit to Scratch as the screen shown.



Creating a Maze Game with micro:bit

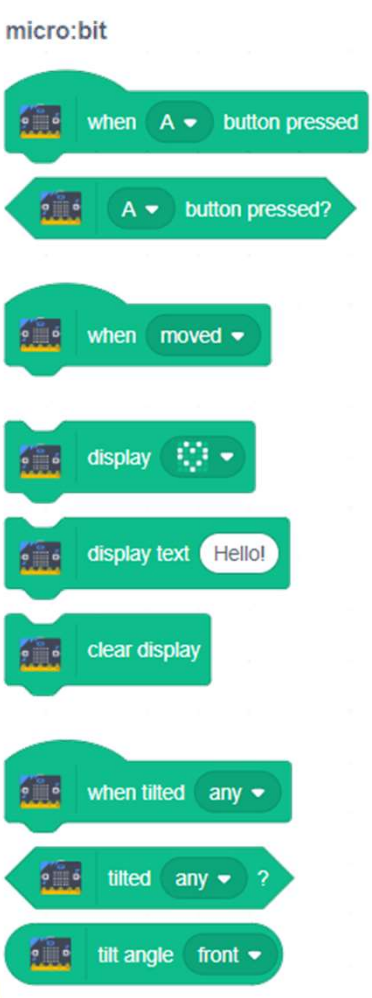
To Think and To Code: Control Panda with micro:bit

Compared with the program that we finished in the previous unit, think about which part of the program needs to be changed?



How can we update the code blocks so we control the Panda sprite with the control of joystick instead of the keyboard?

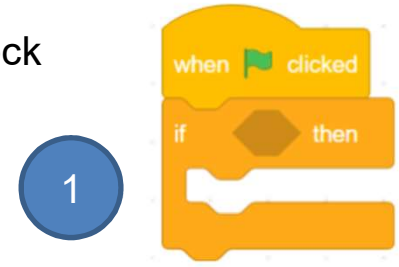
Hint: Look at the micro:bit Drawer.



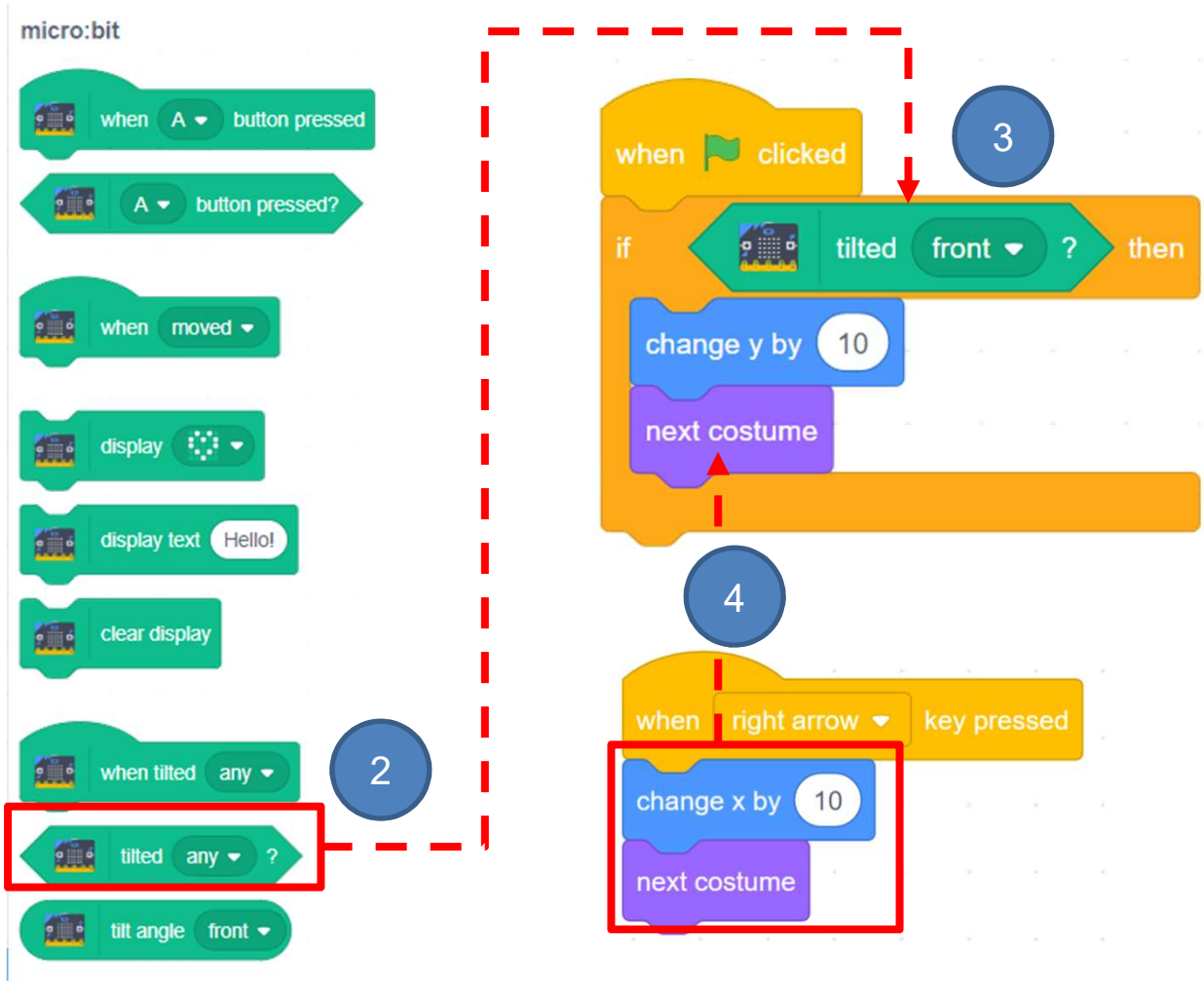
Creating a Maze Game with micro:bit

To Code: Control Panda with micro:bit

1. Drag a "**when green flag clicked**" & an "**if-then**" block and make them together.



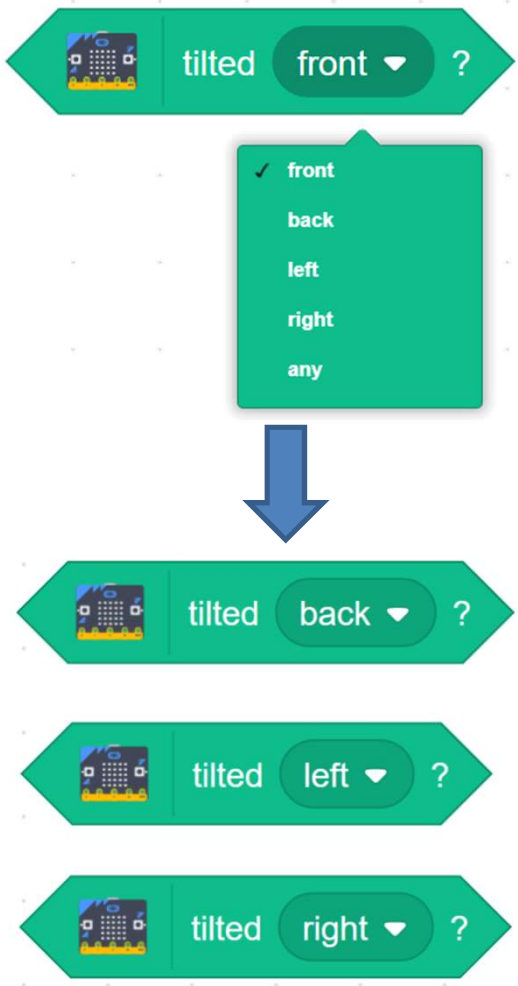
2. From the new added micro:bit drawer, find the "**tilted <any>?**" block.
3. Change "**any**" to "**front**" and insert it into the "**if**" block.
4. Copy the code blocks of movement from "**when key pressed**" event.



Creating a Maze Game with micro:bit

To Code: Control Panda with micro:bit

Duplicate the **"titled front"** block, update and complete the rest of the other 3 directions:



Testing and Debugging

Can you control the sprite with micro:bit? Try more than one time? Which block do you miss?

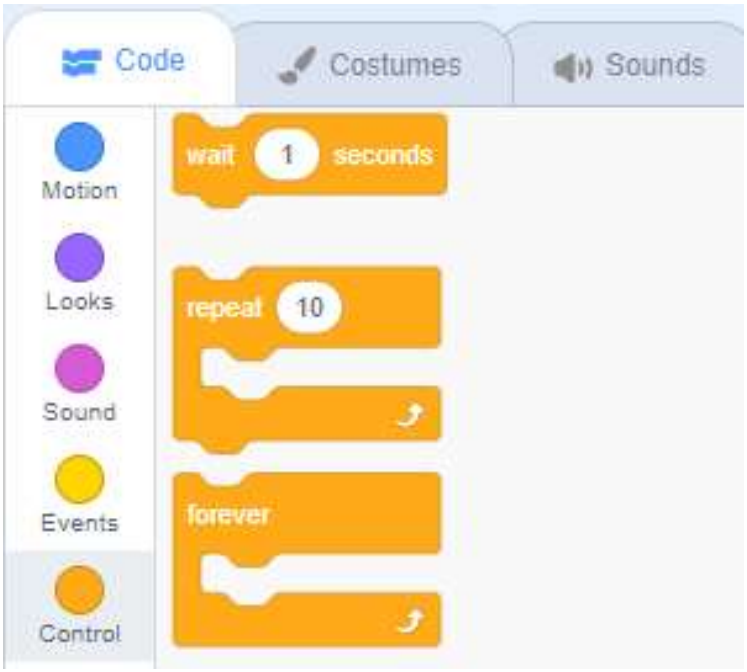
Control Panda with micro:bit

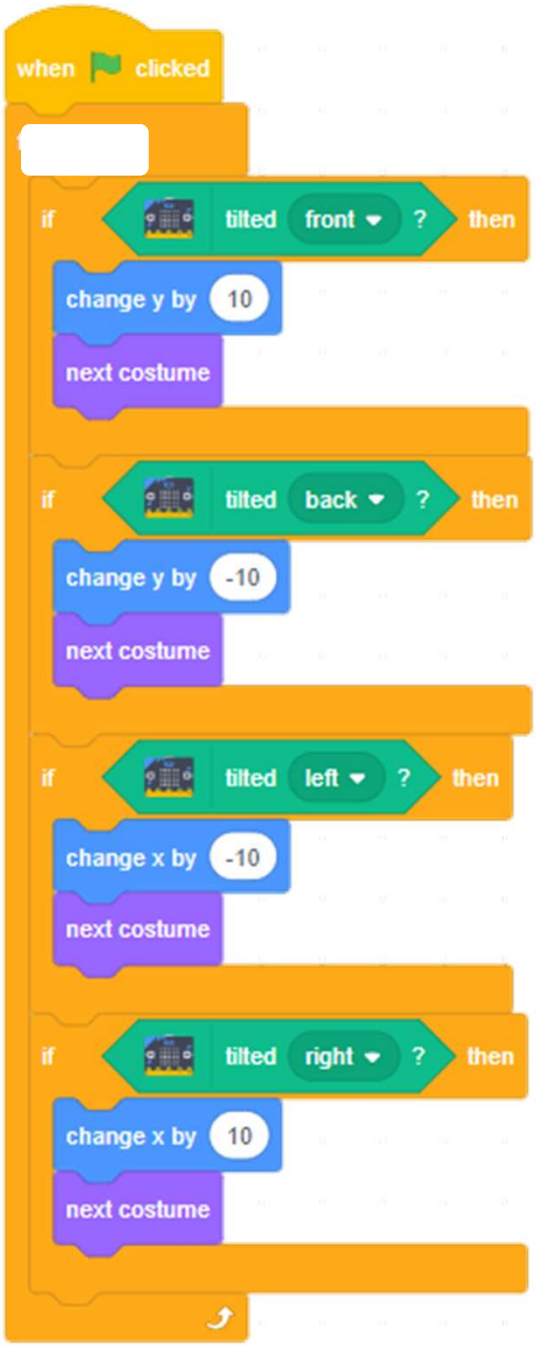
Creating a Maze Game with micro:bit

To Code: Iteration

Which block do you need to **solve** the problem?

Hint: Look for the "Control" drawer.





Testing and Debugging

Try again! Is the joystick too sensitive? What can you do to solve it?

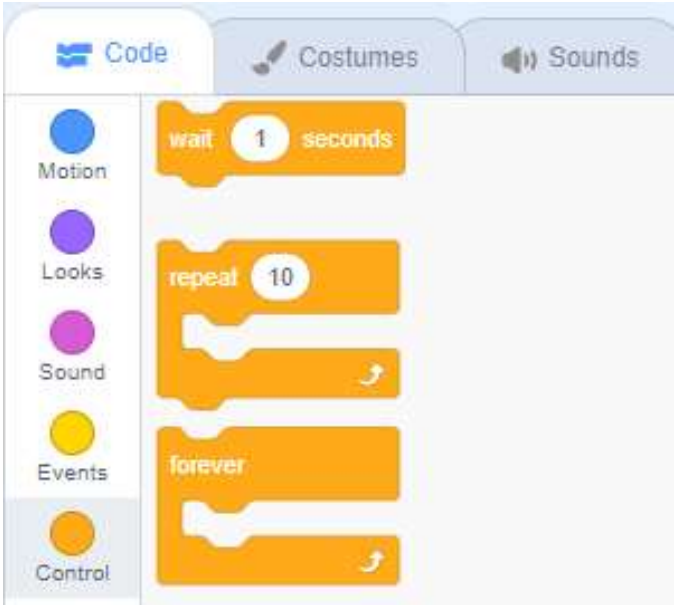
Control Panda with micro:bit → Iteration →

Creating a Maze Game with micro:bit

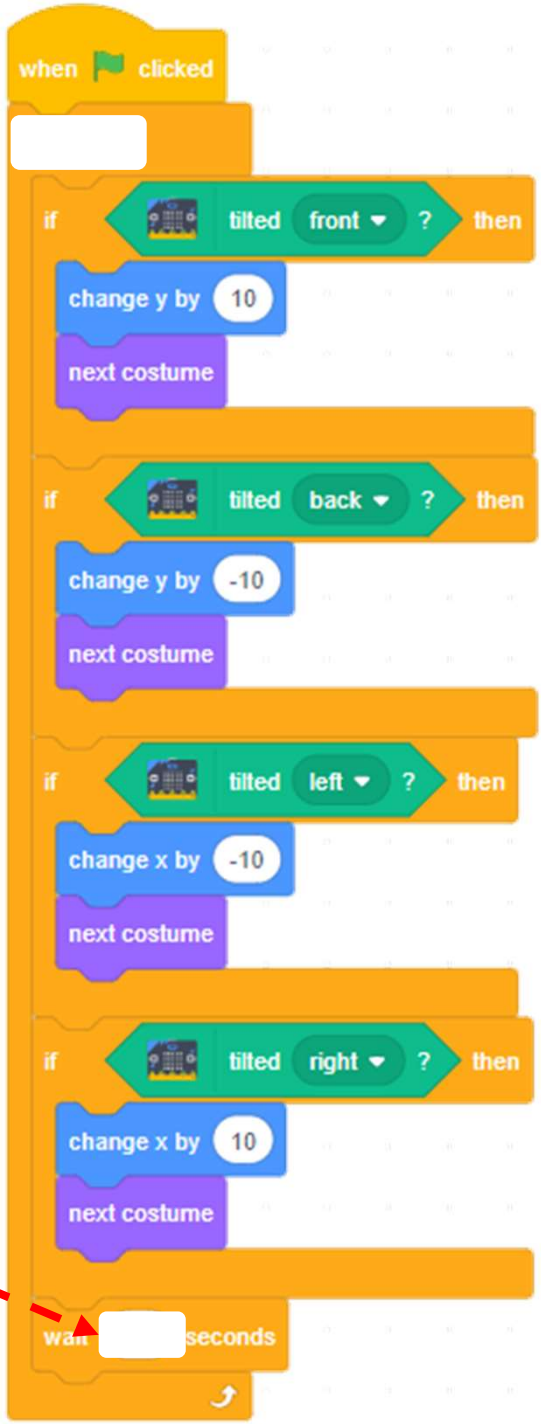
To Code: Add Wait Block

Is the joystick too sensitive? What can you do to solve it?

Add **"wait ___ seconds"** block see if it helps.



How long should we wait? You can input different value like 0.3, 1 second or 3 seconds until you find the appropriate time.



Testing and Debugging

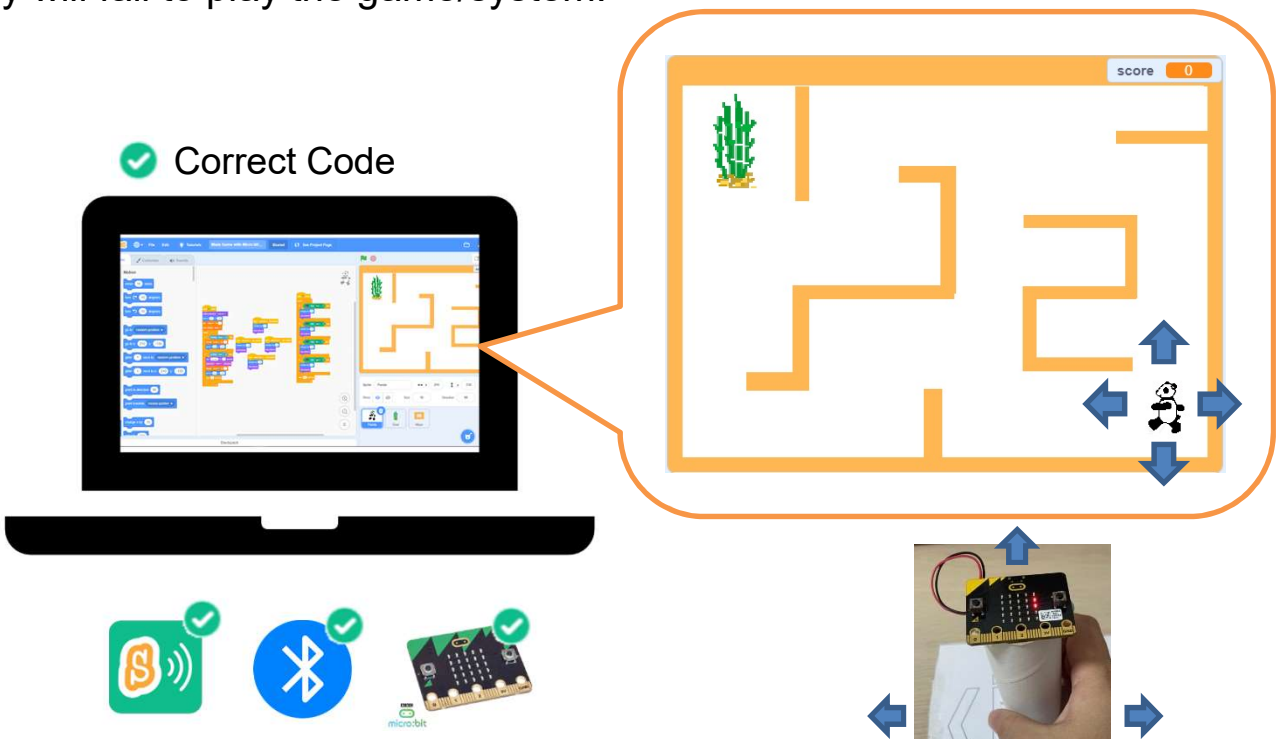
Yes! You did it!
Enjoy the maze game with micro:bit joystick!

Control Panda with micro:bit → Iteration → Add wait block

Creating a Maze Game with micro:bit

Knowledge builds up: Engineering Systems Thinking / Forming a system connected with physical objects

We should prepare the environment well and make everything connected or they will fail to play the game/system.



Integrated Development Environment

Control physical objects

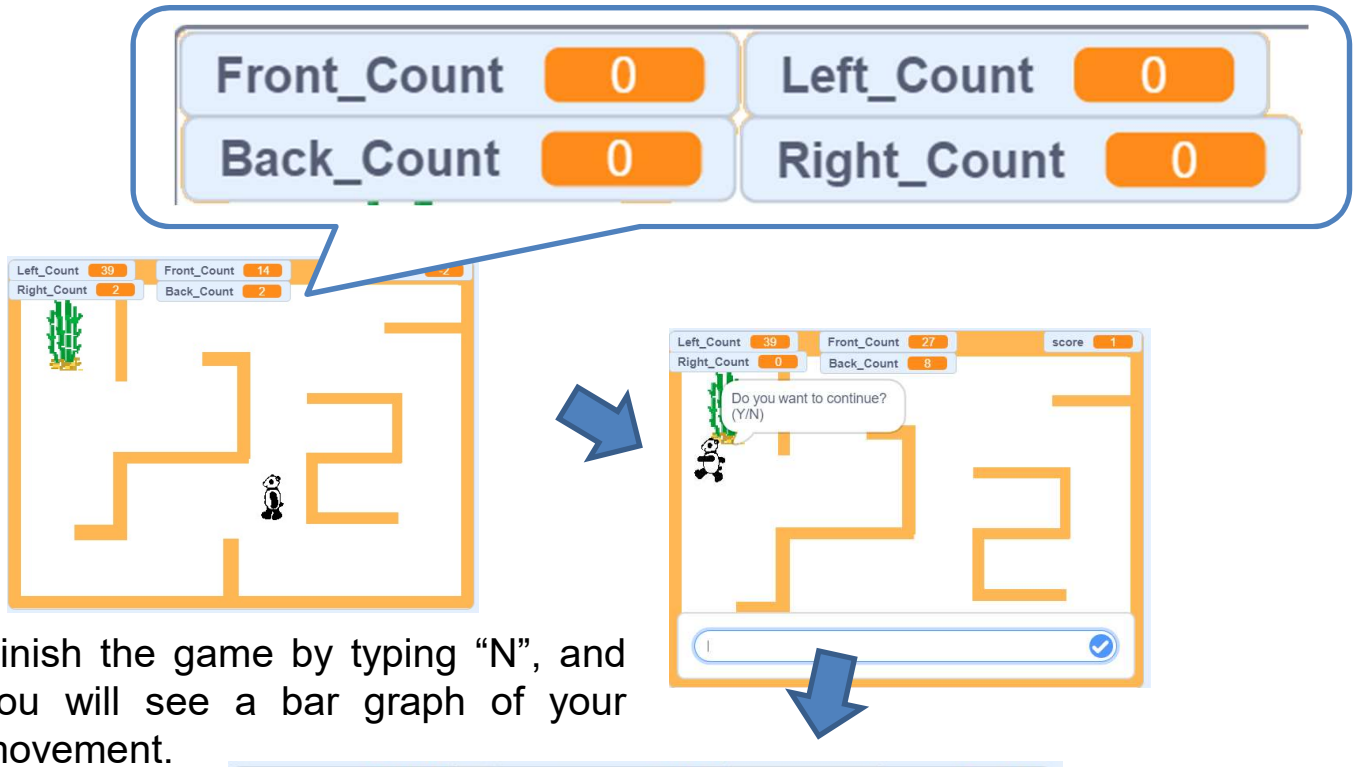
<u>Knowledge builds up: Engineering Systems Thinking / Forming a system connected with physical objects</u>
It is about the thinking of putting interrelated parts together as a whole and enable the physical system work as it is intended.

Creating a Maze Game with micro:bit

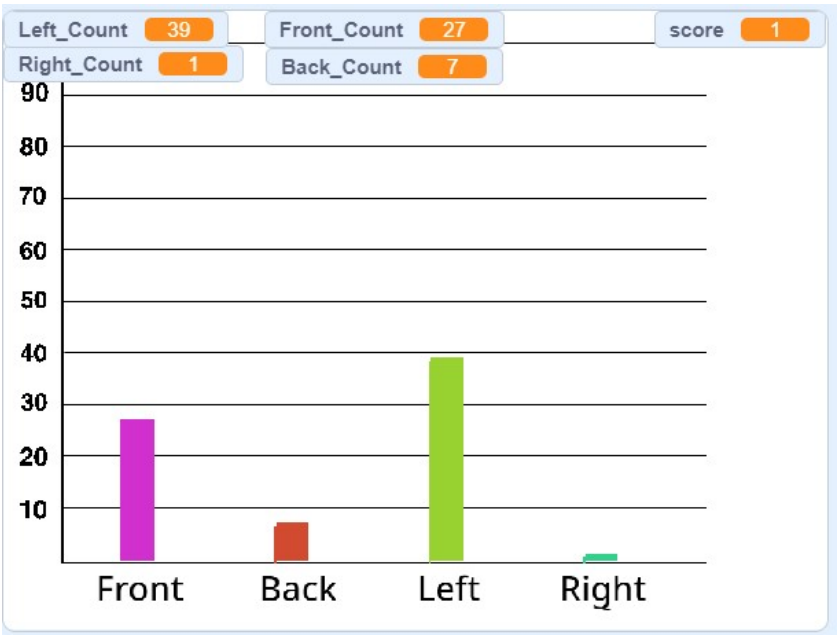
To Play: Maze Game with Data Analysis

Play another Maze Game with data analysis which counts your steps and shows you in bar graph. https://scratch.mit.edu/projects/734745981/.

Other than score, you will see four more variables to count the sprite moving front/back/left/right.



Finish the game by typing “N”, and you will see a bar graph of your movement.

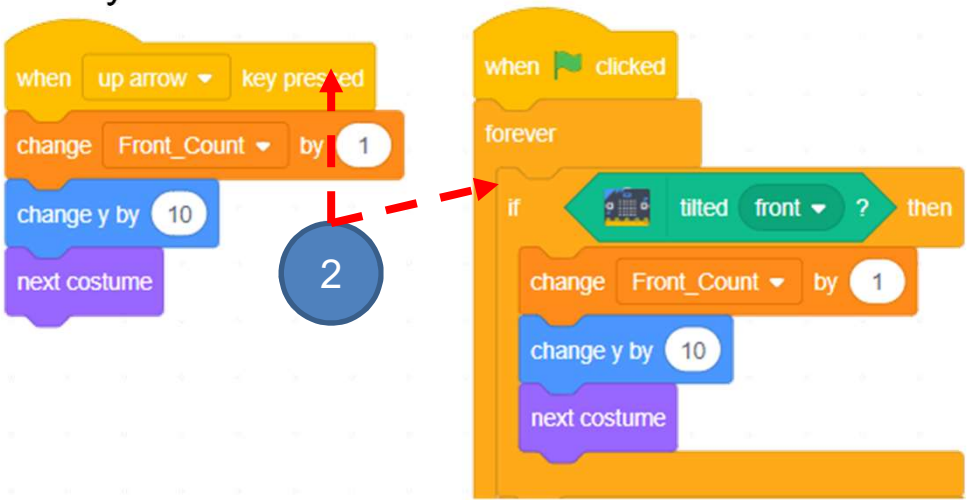


Creating a Maze Game with micro:bit

To Code: Code Comprehension

Click **"See inside"** to see the code.

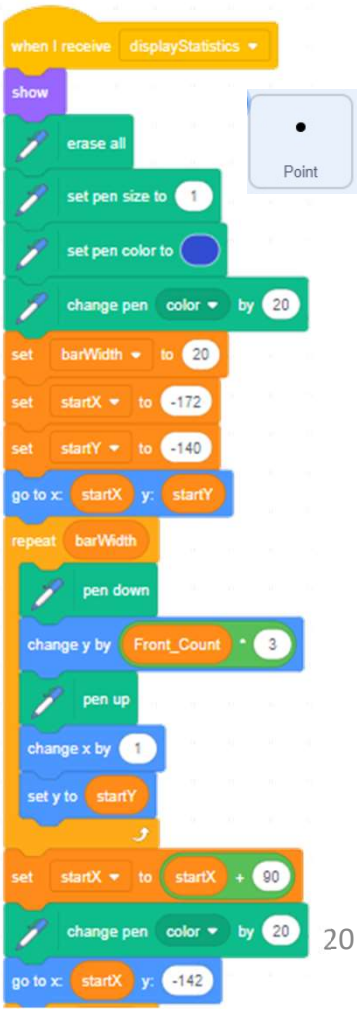
1. There are four variables that counts your steps. It includes different variables to count your steps and shows data in the graph.
2. No matter you control with keyboard or micro:bit, it will count your moves.



3. When you finish the game, it will **"ask"** you if you want to continue.

4

4. By using **"Pen"** feature, there is a **"Point"** Sprite to draw bar graph with those data it counts during the game.



Creating a Maze Game with micro:bit

Conclusion

This unit is a taste of simplified IoT (Sensing – Reasoning – Reacting). With the joystick(micro:bit inside), we make use of the micro:bit build-in sensor (accelerometer) and write the code blocks to interact with the Scratch programming environment and form a system connected with physical objects.

Knowledge Build up – Internet Of Things (IoT)

Internet of Things (IoT): It is the network of physical things that embedded with sensors, apps and related technologies for the purpose of exchanging data with other devices over the internet. It usually use communication technologies such as Bluetooth and WiFi.

By using the Internet of Things, we can coordinate students' learning and activities during the school day.

Do you know any other real-life example with IoT?

_____

Creating a Maze Game with micro:bit

To Create

What do you want to create with the accelerometer of micro:bit you learnt today? Draw your idea in the box below:

micro:bit

Creating a Maze Game with micro:bit

To Reflect: Two Stars and a Wish Worksheet

Name of Project: _____ Name of Creator: _____

Please write down two things that you like about this project.

1

_____

2

_____

What is one thing you would like to add or change to make this project better?

_____

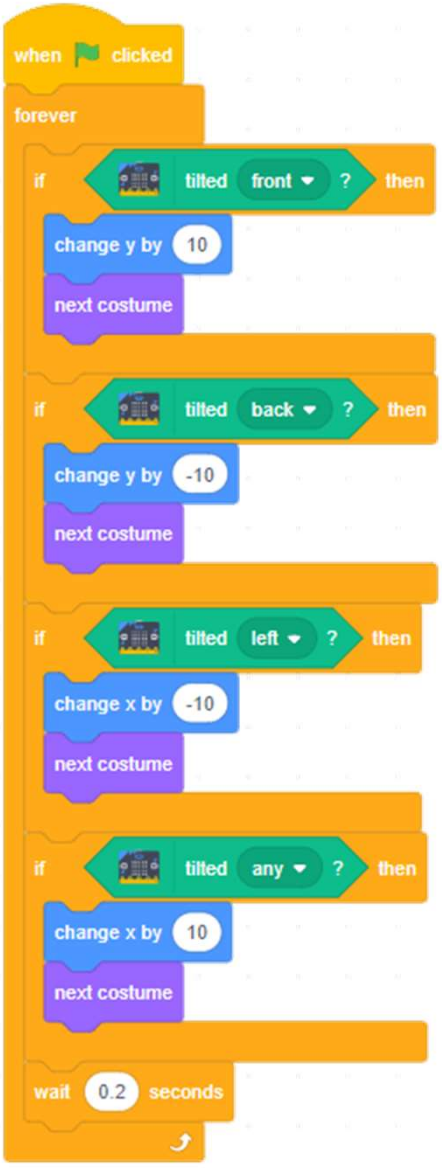
_____

_____

Creating a Maze Game with micro:bit

Review Questions

1. A student tried controlling the sprite with micro:bit but failed to move to the correct direction. Read the following code blocks and try to debug.



A. Remove the forever block.

B. The wait seconds should be updated from “0.2” to “2”.

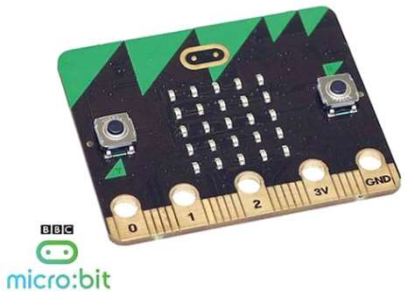
C. The forth if-then statement “tilted any?” should be replaced with “tilted right”.

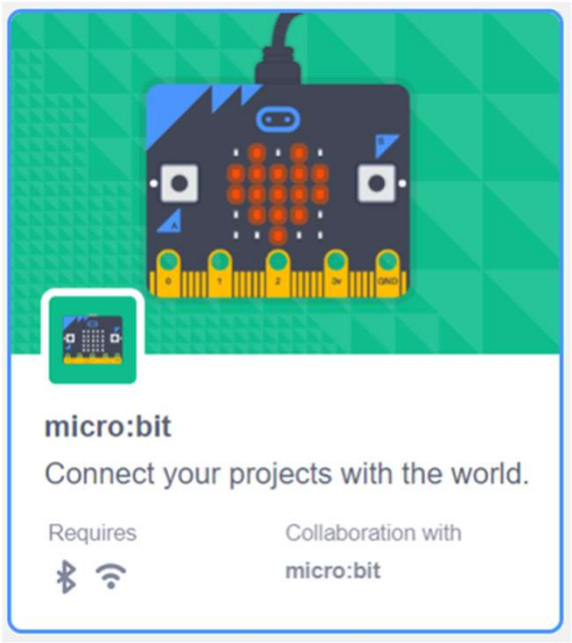
D. Use another micro:bit instead.

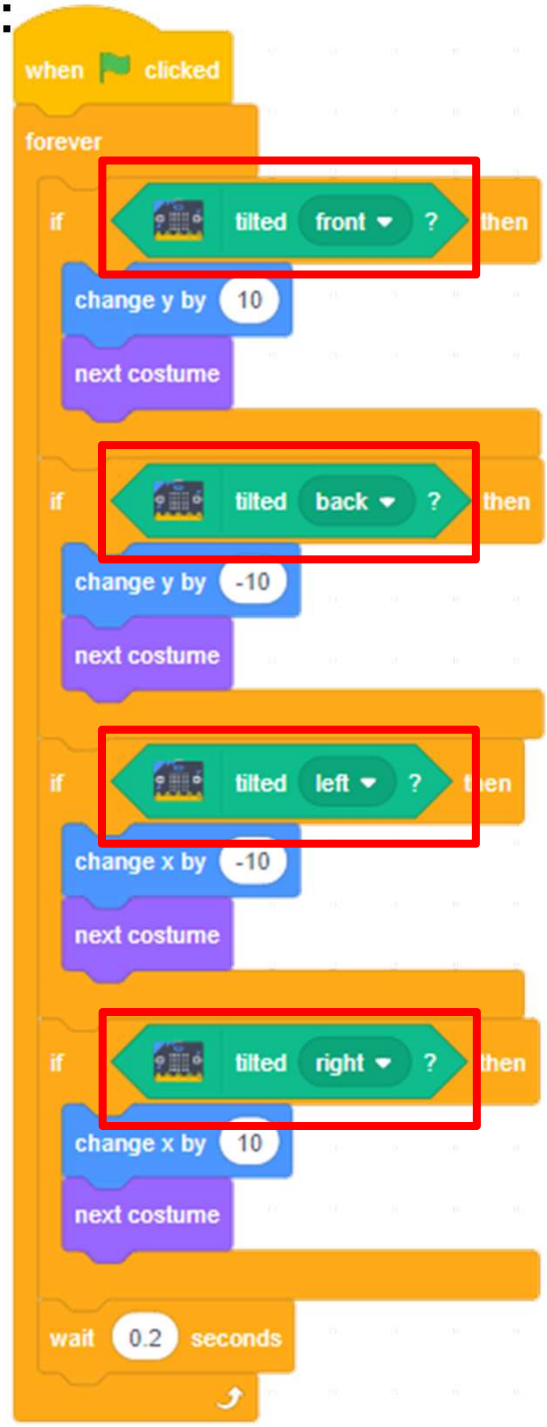
Creating a Maze Game with micro:bit

Revision on Key Features

micro:bit and accelerometer:





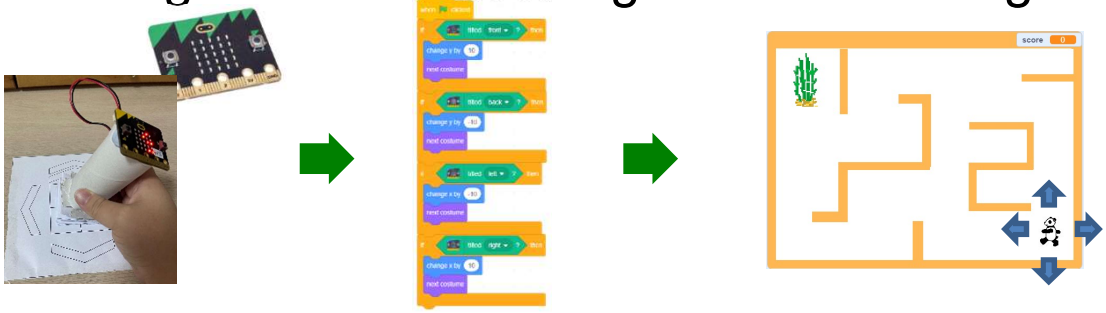


Creating a Maze Game with micro:bit

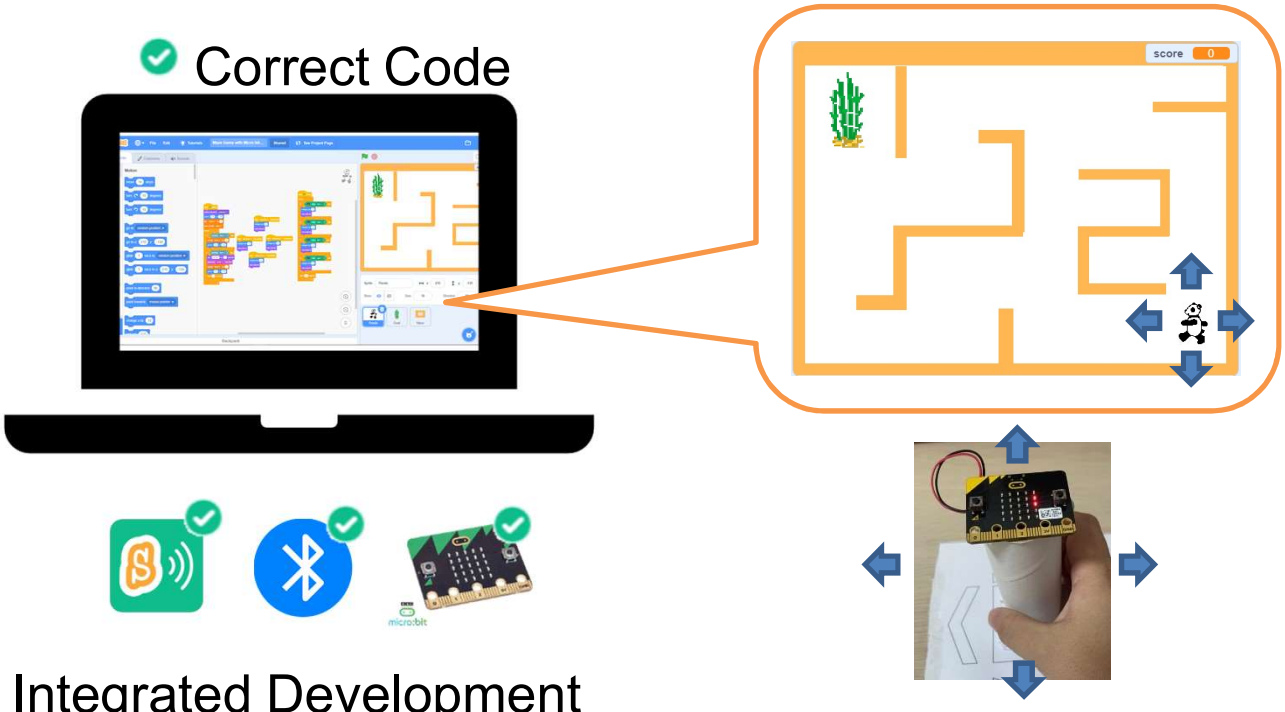
Revision on Key Concepts & Practices

Sensing: Collect data from the sensors.
Reasoning: Judge and make decisions based on the data.
Reacting: Respond according to the result of reasoning.

Sensing → Reasoning → Reacting



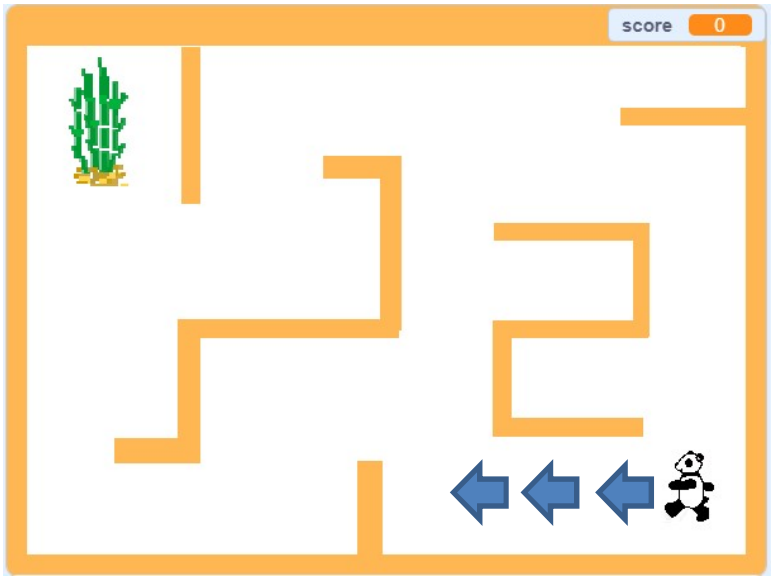
Engineering Systems Thinking / Forming a system connected with physical objects: It is about the thinking of putting interrelated parts together as a whole and enable the physical system work as it is intended.

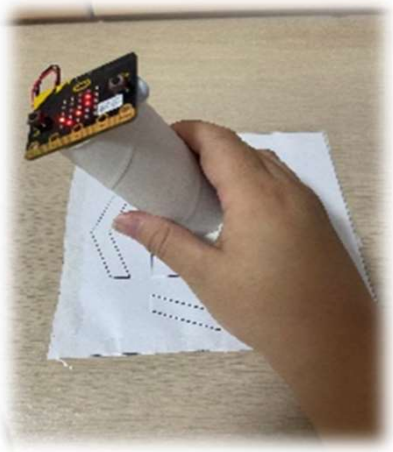


Creating a Maze Game with micro:bit

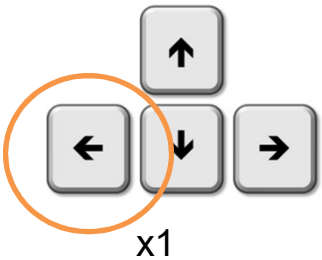
Revision on Key Concepts & Practices

Use the micro:bit's accelerometer (joystick) instead of the keyboard to control the Panda sprite.





Keep Moving



x1
x2
x3…

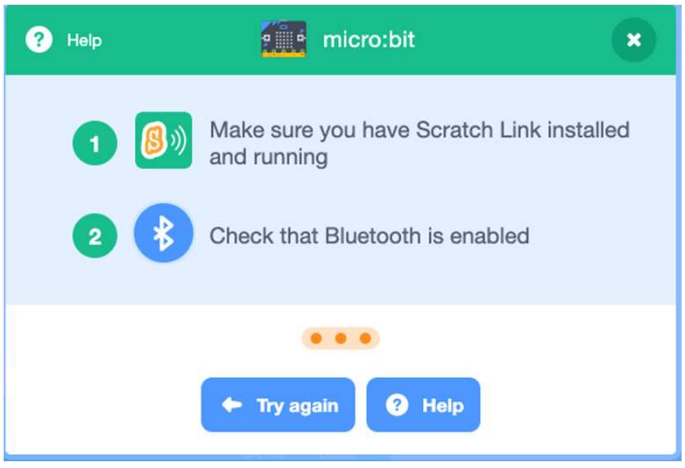
One Click, One Step

Creating a Maze Game with micro:bit

Revision on Key Concepts & Practices

Sequencing: The process of combining things in a specific order.

Casual Reasoning: The process of identifying causality: the cause and its effect.



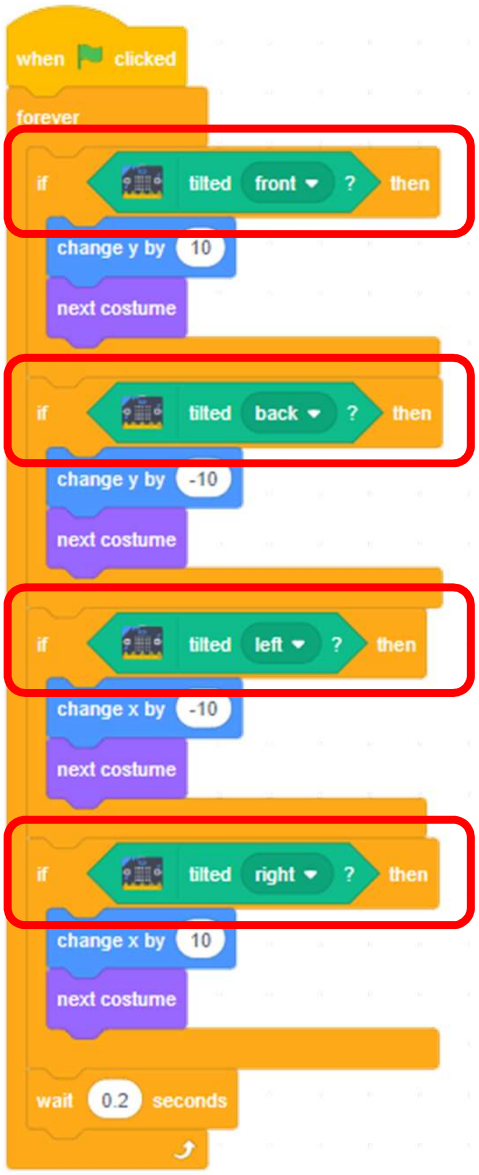
Error Screen

Checking Items

Creating a Maze Game with micro:bit

Revision on Key Concepts & Practices

Branching/Selection: We use conditional statements in programming to enable computers to make decisions. Conditionals always have an if part, which tells the program in the then part what to do when the condition is true.

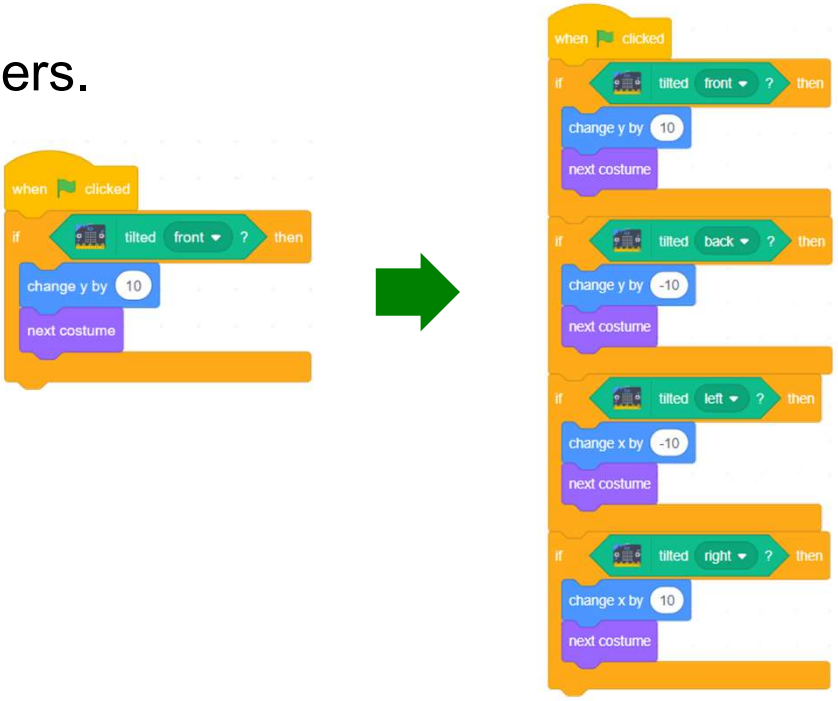


Creating a Maze Game with micro:bit

Revision on Key Concepts & Practices

Variables: In programming, variables are used to store values. It has a name, can only store one value at a time and be updated. For example, we use variables to count front/back/left/right key pressed events.

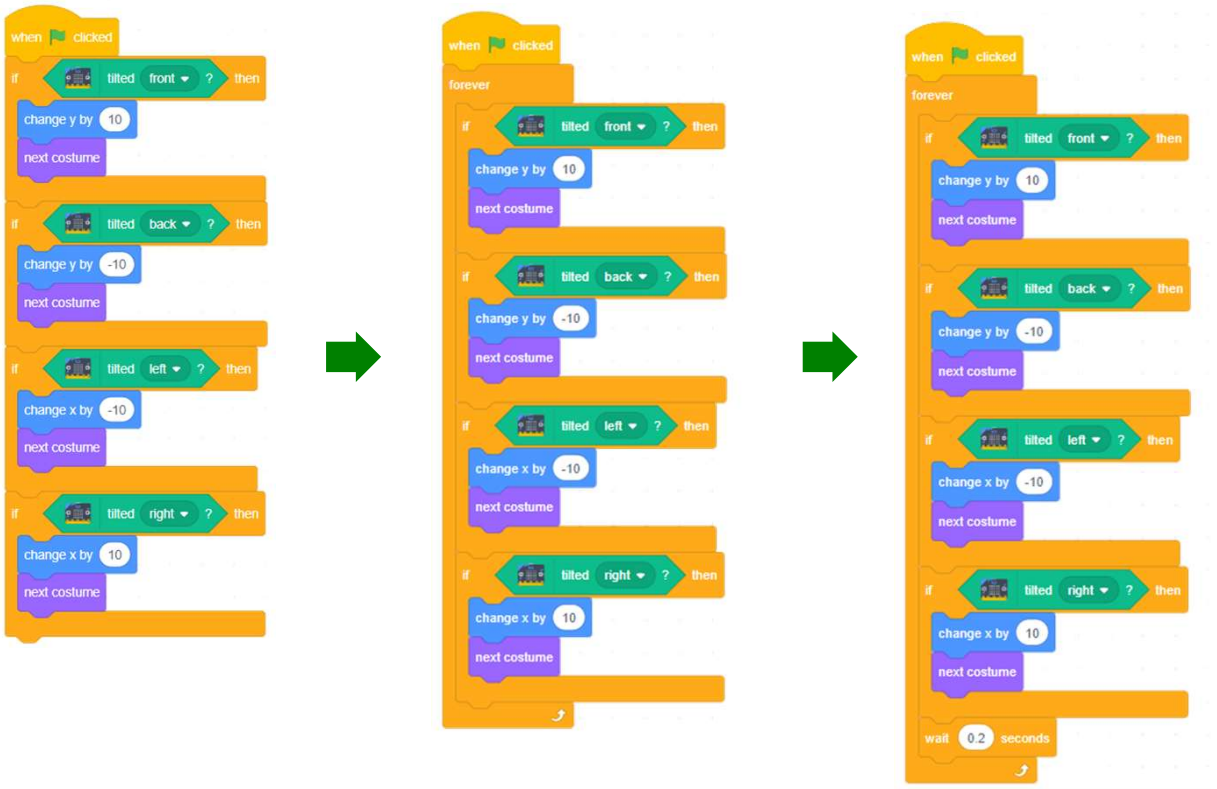
Reuse and Remix programs/codes: The reuse and remix of the works of other programmers are crucial in the online communities of Scratch. We can reuse and remix the codes of one micro:bit tilted event and use it for the others.



Creating a Maze Game with micro:bit

Revision on Key Concepts & Practices

Being incremental and iterative: to work out a sub-task as an iteration, try it out, then work out another sub-task in one more iteration until the whole programming task is completed.



Testing and Debugging: Testing a computer program is the process of checking if it can produce outcomes as designed. Debugging a computer program is the process of finding out ways to revise the program so that the bugs can be removed.

Control Panda with micro:bit → Iteration → Add Wait Block

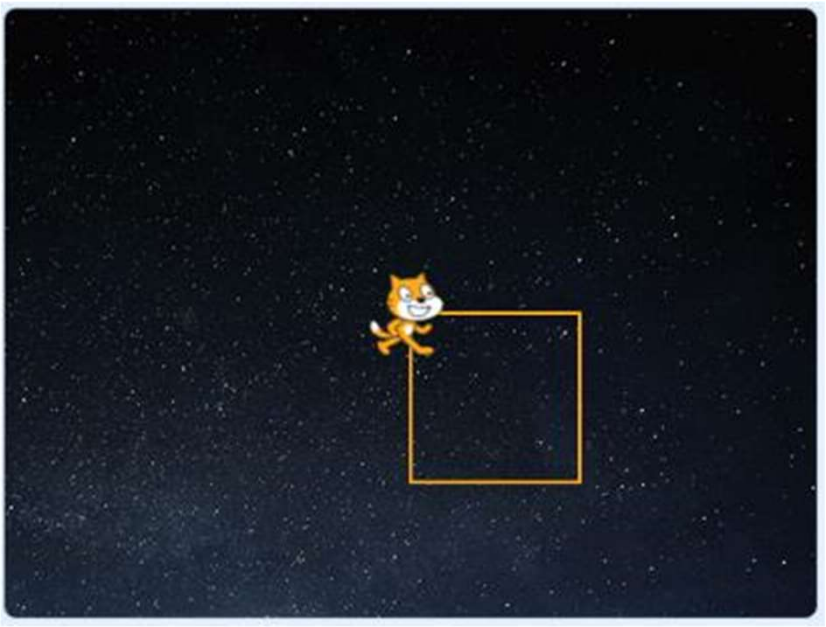
Unit 7: Drawing Shapes in Scratch Student Guide

Content

Drawing Shapes in Scratch

Let's learn to draw shapes in Scratch!

In this unit, you will learn how to draw a line and square using the Pen feature of Scratch.



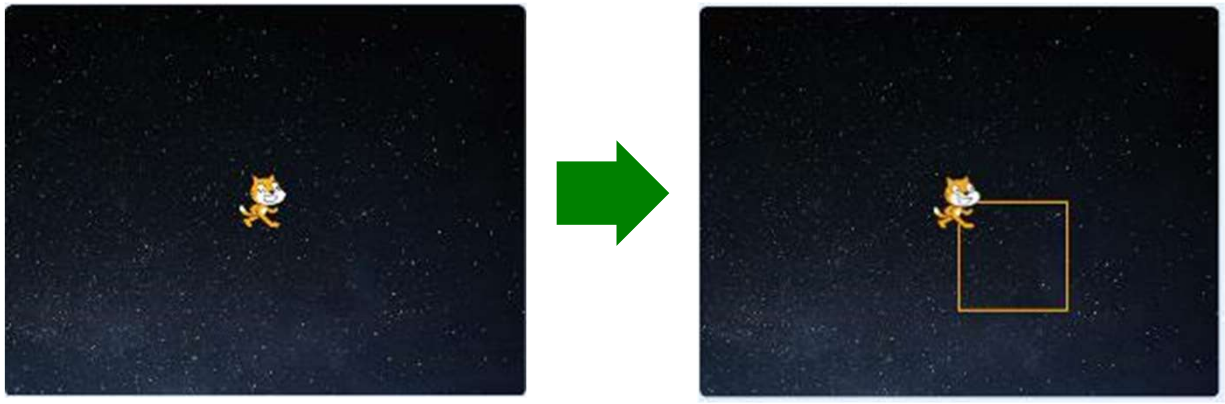
To Play

Open the Scratch Project, Drawing Shapes in Scratch:
https://scratch.mit.edu/projects/737338011/

Click the green flag and see what happen.

Click again to see where the sprite moves.

Did the Scratch cat draw a square?

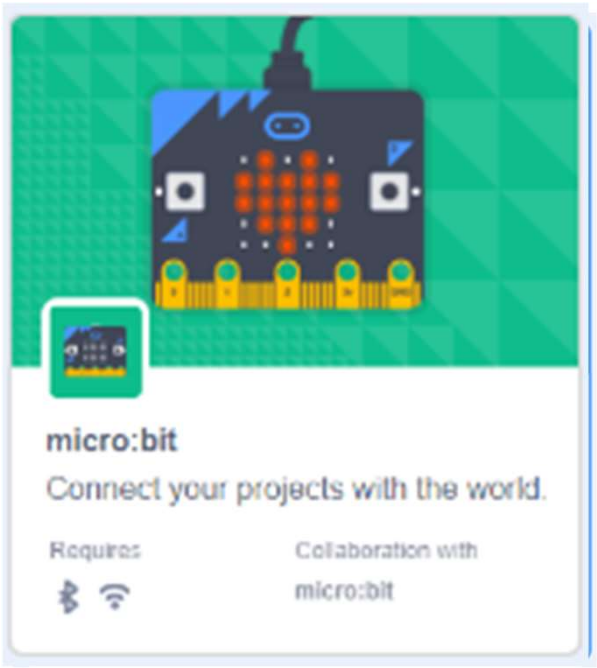


Drawing Shapes in Scratch

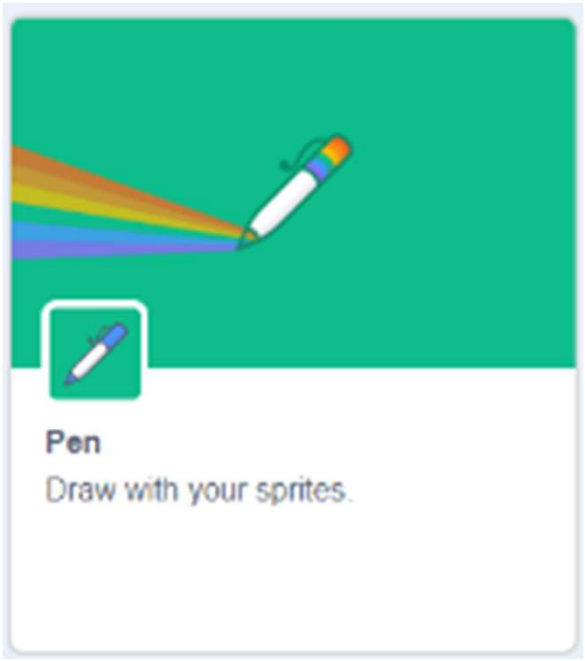
To Think

1. Do you know which feature will we use to draw a square in this unit? Tick the box (✓).

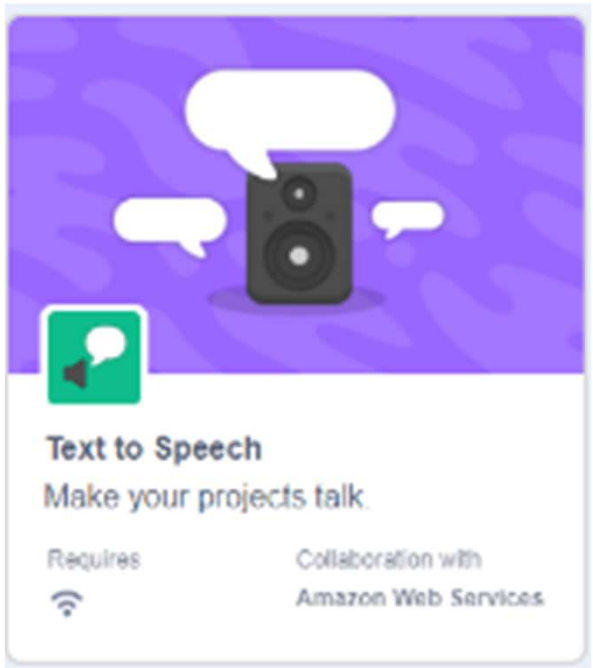
❑ Micro:bit



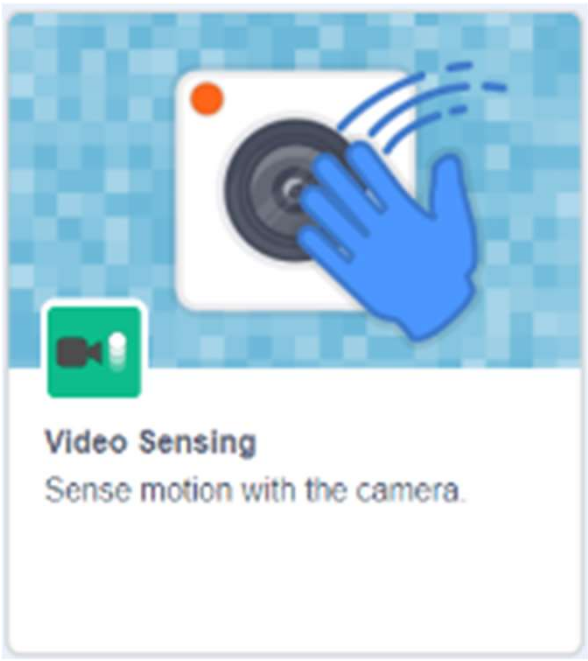
❑ Pen



❑ Text to Speech



❑ Video Sensing

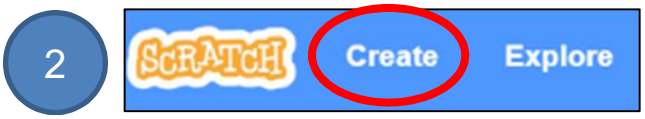


Drawing Shapes in Scratch

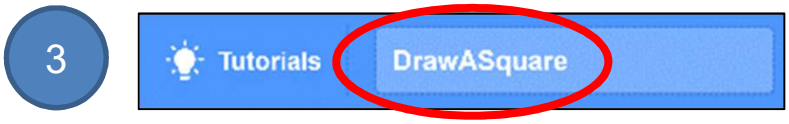
Start Here

1. Sign into your account at https://scratch.mit.edu/

2. Go to "Create" to start a new project.

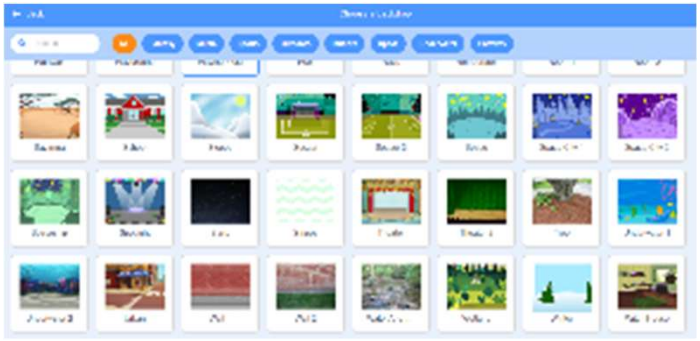


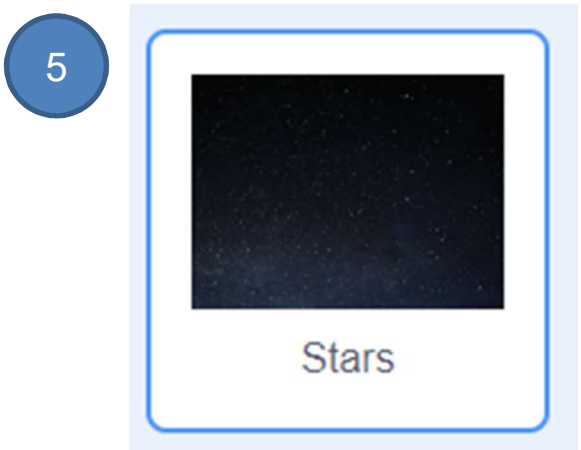
3. Name it "DrawASquare".

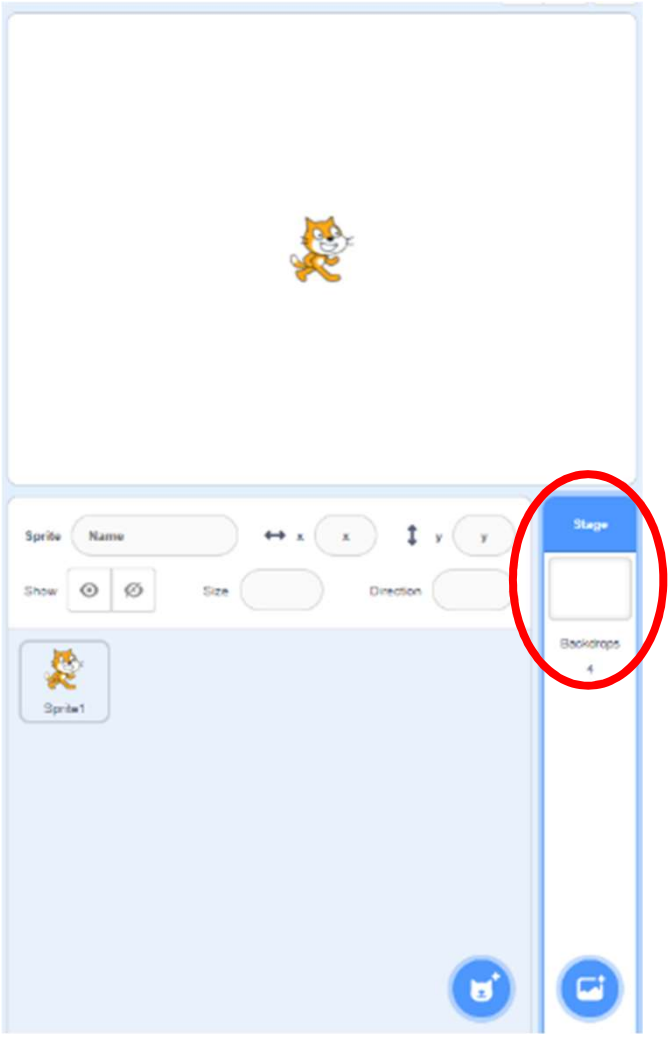


4. You can change the backdrop for your project.

5. For example, you may choose "Stars".





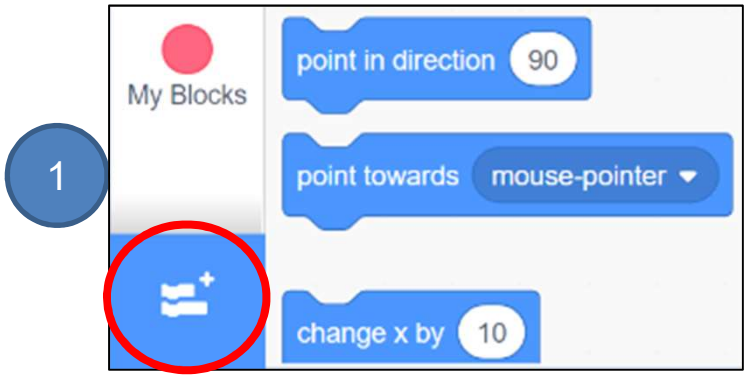


Drawing Shapes in Scratch

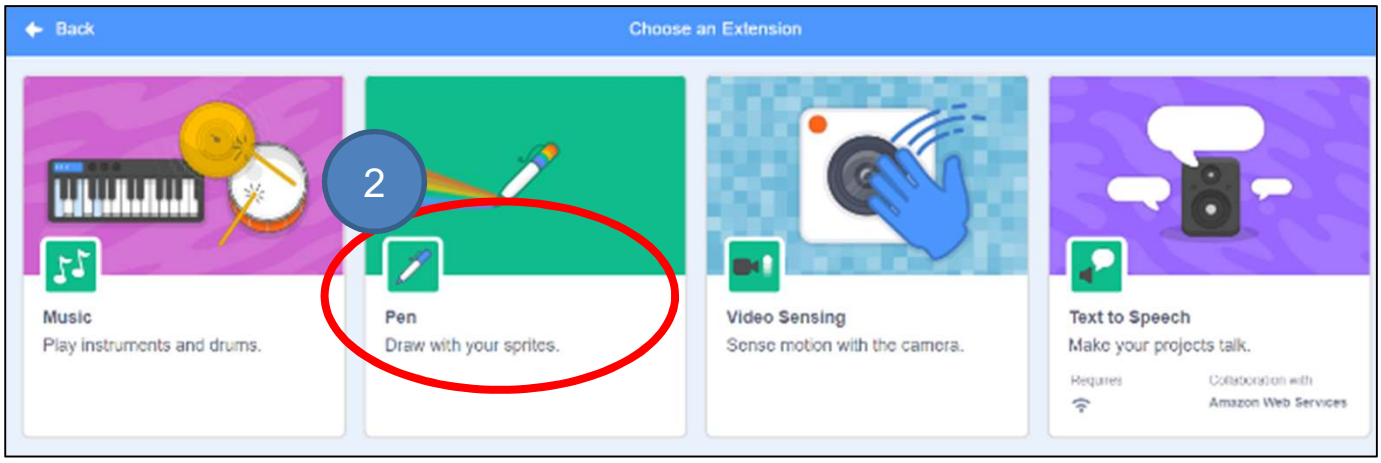
To Code: Add Pen Feature

Before you start drawing, you need to add the Pen component to your Scratch project.

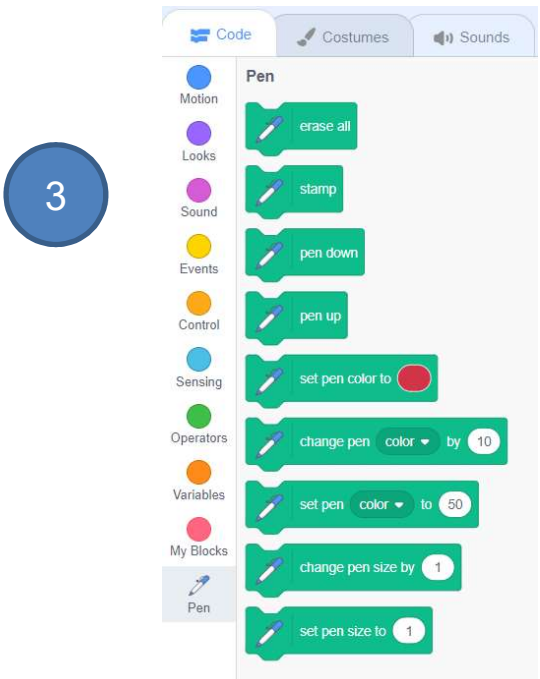
1. Click on the "Add Extension" icon at the bottom left of the page.



2. Select the "Pen" extension.



3. Go to the "Pen" drawer under "Code" tab to see a list of blocks that you can use.

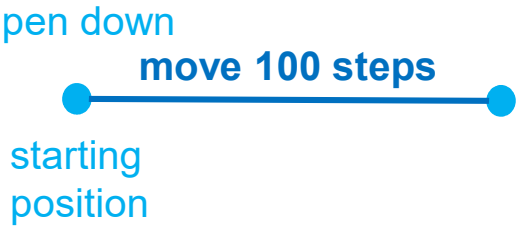


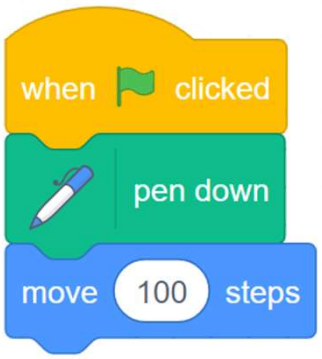
Drawing Shapes in Scratch

See Appendix P.28

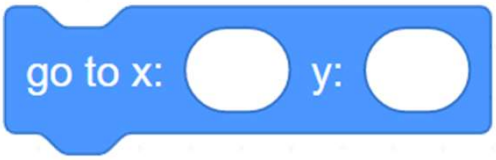
To Code: Draw a Line

With the Pen feature, by moving the sprite, you can draw a line.

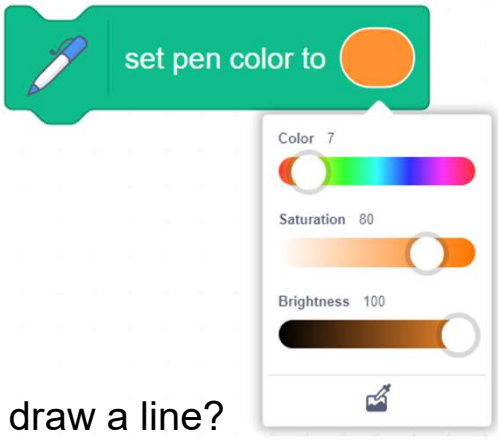




1. Run the code blocks on the right, can you draw a line?
2. Click the green flag again, what is the difference this time?
   How would you solve it?
   Hint: Where is the starting position of the cat?
   What should you do with the marks everytime you draw?



3. Enhancement: Change the pen color to your favourite color.
   You may also change the pen size as well.



Testing and Debugging

Click on the green flag, did you successfully draw a line?

Drawing Shapes in Scratch

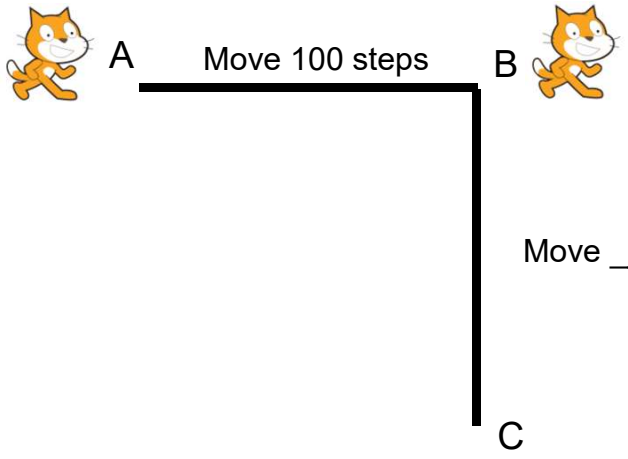
To Think and To Code: Draw a Square

After you drew a line, how can you draw a square?

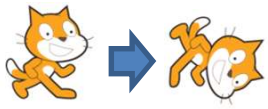
See Appendix P.29

Before we code, can you continue to draw a square below?

Let's say if the Scratch Cat moves 100 steps from A to B, then how long it moves from B to C?



To move from B to C, the Scratch Cat needs to turn its body down:



Go back to Scratch. In **"Motion"** drawer, drag out a **"turn (right)_____degrees"** block.

Hint: Try 30/60/90 degrees for the suitable one with testing.

After you find the right degrees, you may extend the line to a square!

Testing and Debugging

After completing a small step of programming, you can click the green flag and test it to see if your idea works. Then continue programming until you draw a square.

Draw a Line → Draw a Square →

Drawing Shapes in Scratch

To Think and To Code: Use Repeat Block

Do you see some blocks are repeating?

How many times do these blocks repeat to draw a square?

See Appendix P.30

Can you find an alternative way to code with repeat block?
Hint: Drag a "repeat" block from the Control drawer.

Testing and Debugging

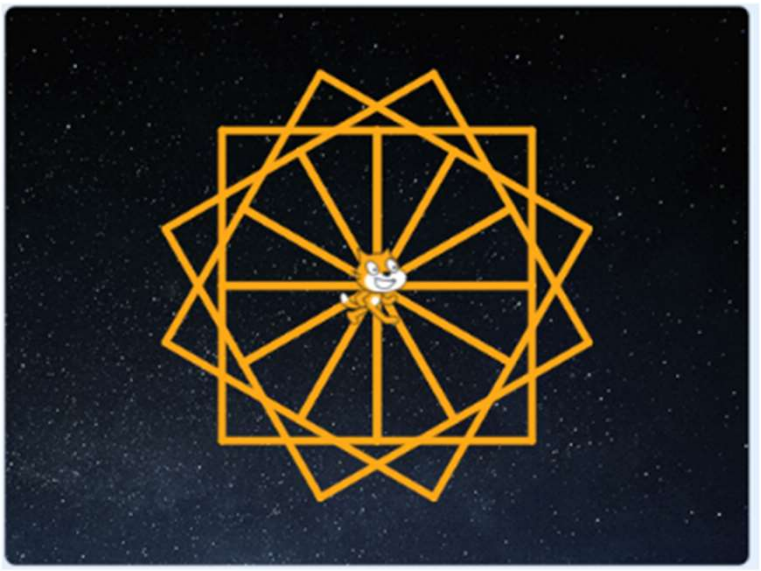
Click on the green flag to test!

Are you able to draw a square using the repeat block?

Draw a Line → Draw a Square → Use Repeat Block

Drawing Shapes in Scratch

In this lesson, you will learn how to make a snowflake with multiple squares.

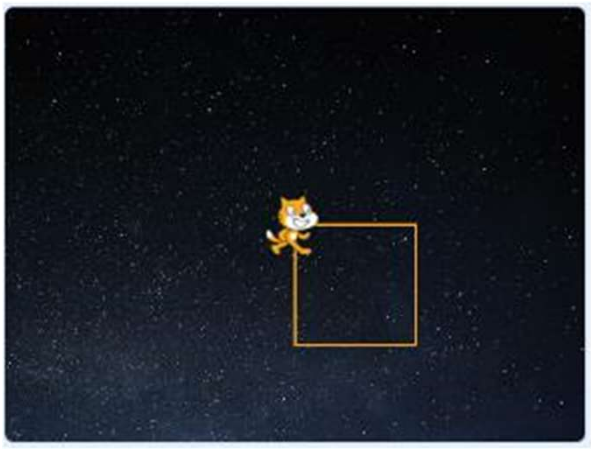


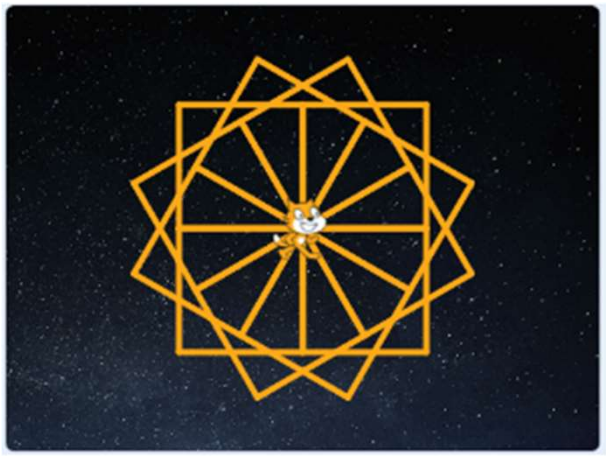
To Play

Open another Scratch Project: https://scratch.mit.edu/projects/737402437

Click the green flag and see what will happen.

Do you know the relationship between a square and a snowflake?

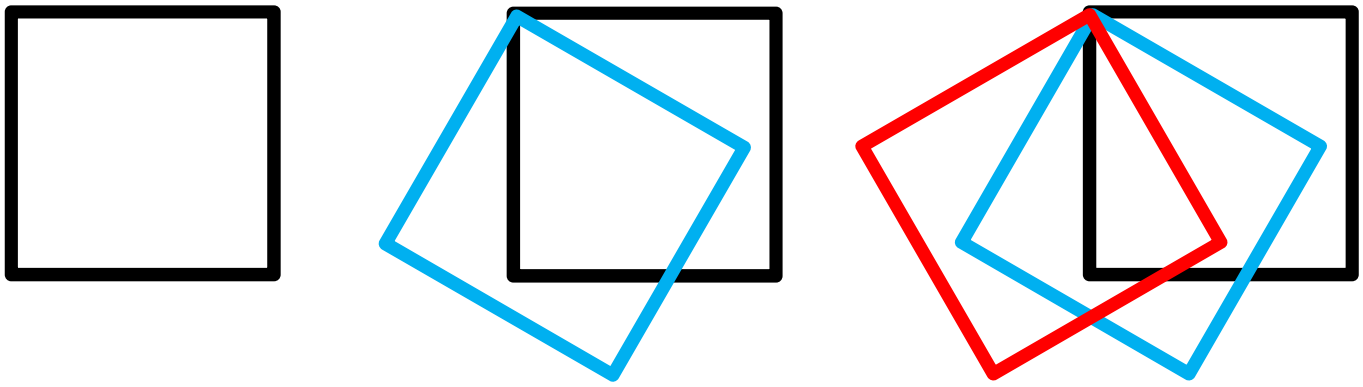




Drawing Shapes in Scratch

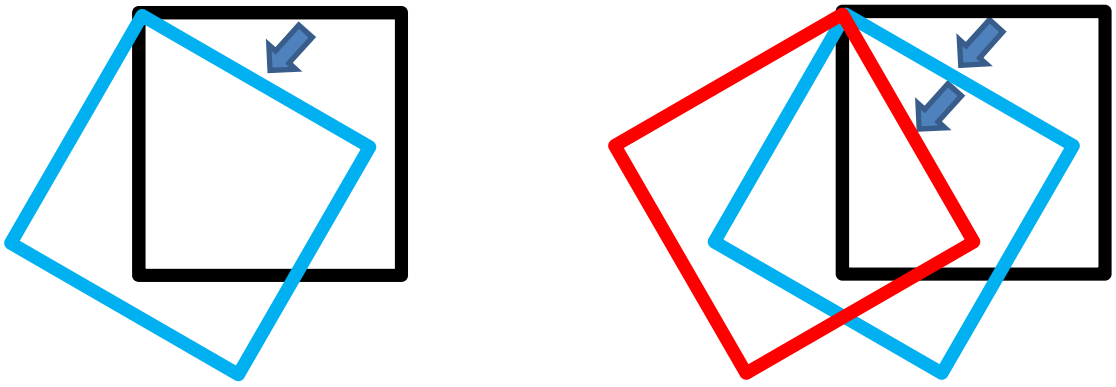
To Think: A Snowflake = Multiple Squares

Let's break down the snowflake to see how it can be made.
Fill in the blank to count how many square(s) are in these shapes:



1 _____ _____

Draw one square, turn some degrees and then draw another one over and over to make these shapes.

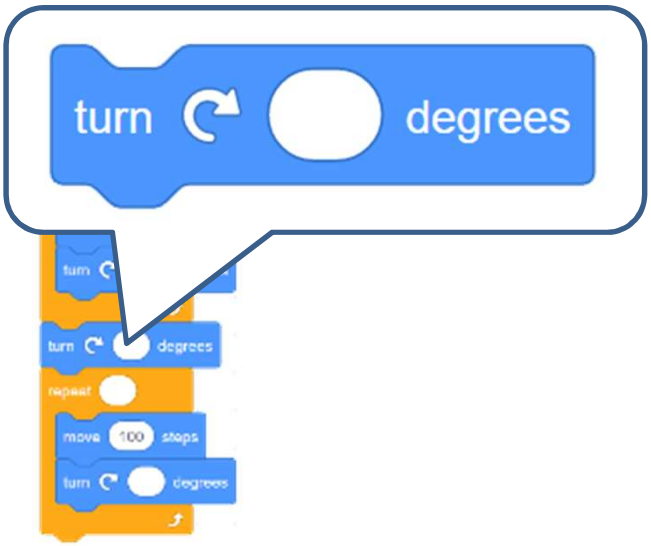


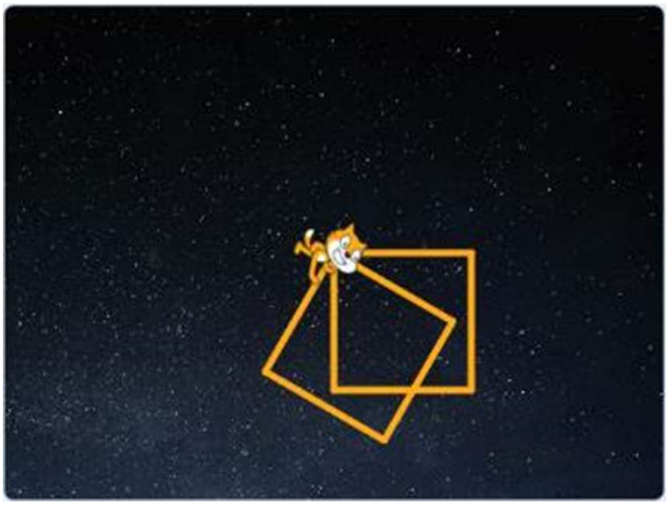
Drawing Shapes in Scratch

To Think and To Code: Draw Multiple Squares

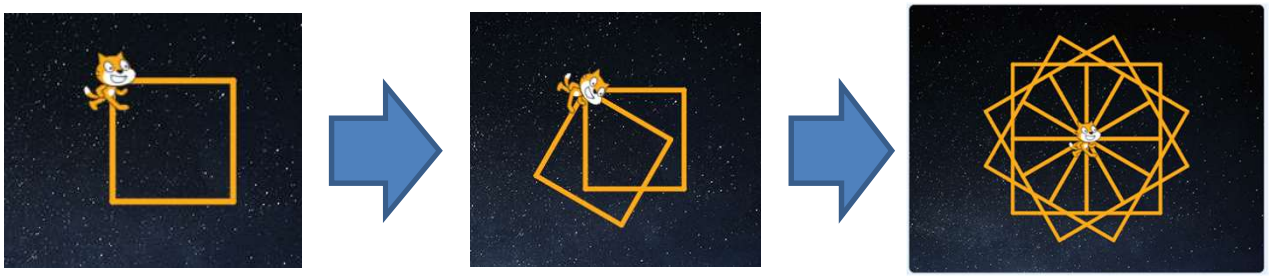
Time to try in Scratch! Save the project as a copy called **DrawASnowflake**.

After drawing the first square, how many degrees do you think it should turn to draw another square? Hint: Try 30 degrees.





Repeat this step (for 12 times) until you can draw a snowflake.



Testing and Debugging

Click the green flag to test! Can you draw multiple squares (a snowflake)?

Drawing Shapes in Scratch

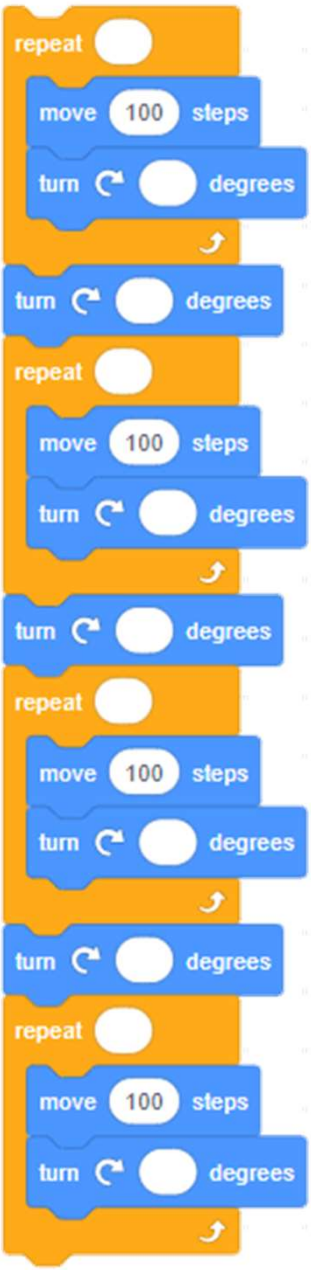
To Think and To Code: Draw Multiple Squares

How long are your code blocks now?
Do you think they are too lengthy?

See Appendix P.31

Based on what you have learnt, how can we make it shorter and clearer?

Hint: Try **"Repeat"** block.



Let's think:
Can we change the **"Turn___Degrees"** and **"repeat___"** to draw different snowflakes?

Testing and Debugging

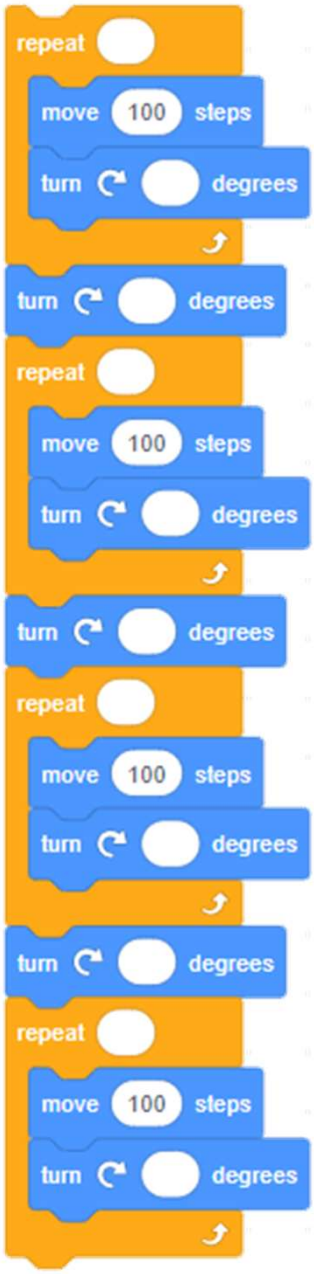
Test again! Can you draw the same shapes with shorter codes?

Drawing Multiple Squares

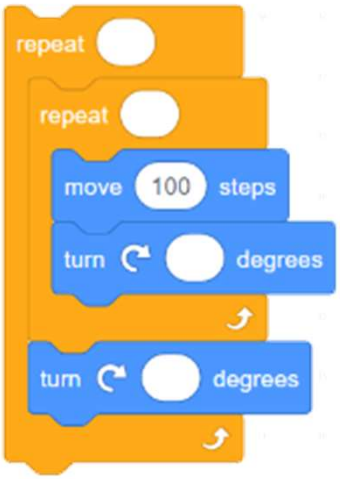
Drawing Shapes in Scratch

To Think

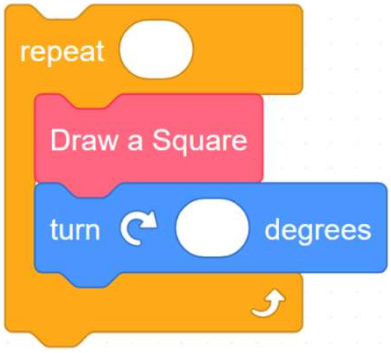
Check if you use the following way to create two or more squares with shorter blocks:



Is this clear enough?



What if we can just tell computer to "Draw a Square"?



Drawing Shapes in Scratch

To Think

Unplugged Activity: Drinking Water

Drinking water is simple. Can you write down the steps of “drinking water”?

A

B

Drinking Water
Take out the water bottle, _____ _____

Usually, when you want to drink water in the classroom, how would you ask the teacher? Try these two different ways to do so:

A. Teacher, can I “drink water”?
B. Teacher, can I (all the steps of “drinking water”)?

Which way is clear and better?

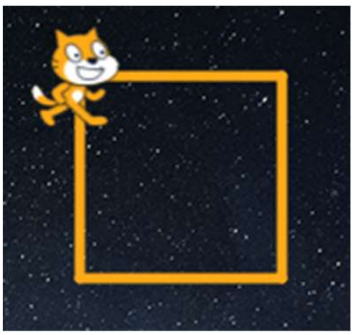
In the same way, how should we tell computer to “Draw a Square”? Recall the code blocks you used to draw the square in Scratch. What are the detailed steps?

Drawing Shapes in Scratch

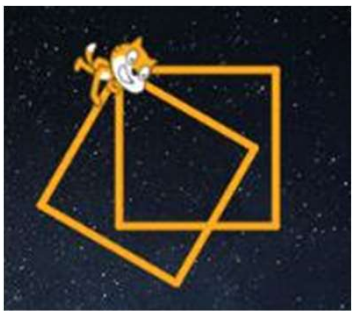
To Think

Can you write down the steps that we draw a line/square/multiple squares?

Draw a **Line**:
Move _____ steps



Draw a **Square**:
Repeat the steps of “Draw a Line” > Turn _____ Degrees > Repeat _____ Times



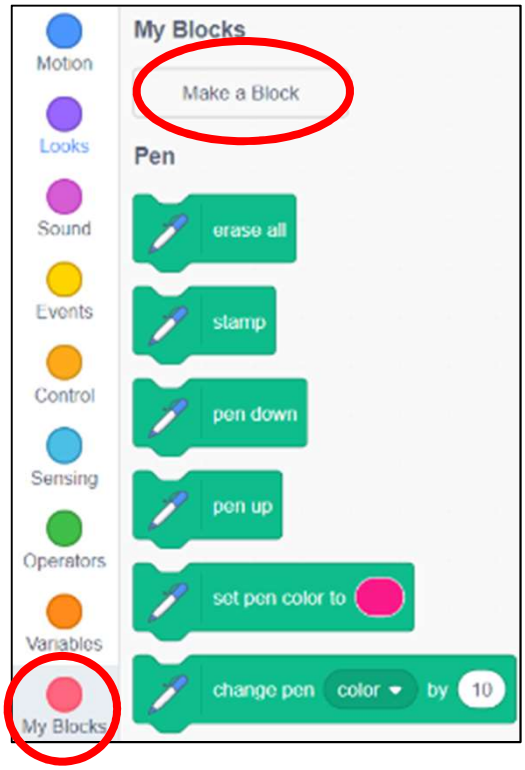
Draw **Multiple Squares**:
Repeat the steps of “Draw a _____” > Turn _____ Degrees > Repeat _____ Times

Drawing Shapes in Scratch

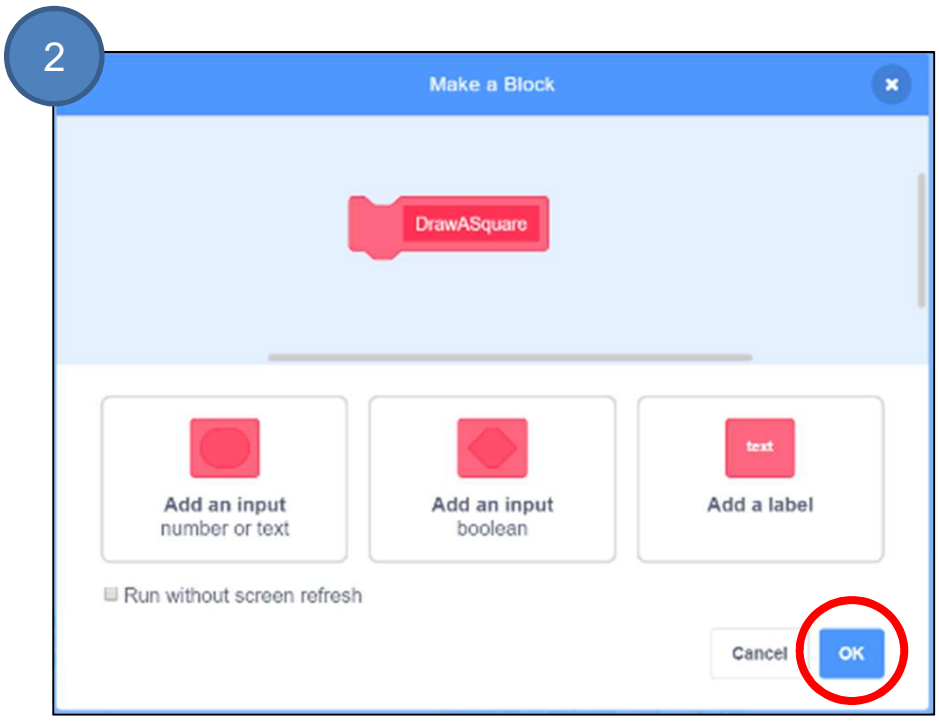
To Think and To Code: Create "DrawASquare" Block

Let's **make** our own custom block!

1. Click on the **"Make a Block"** button in the **"My Blocks"** drawer.



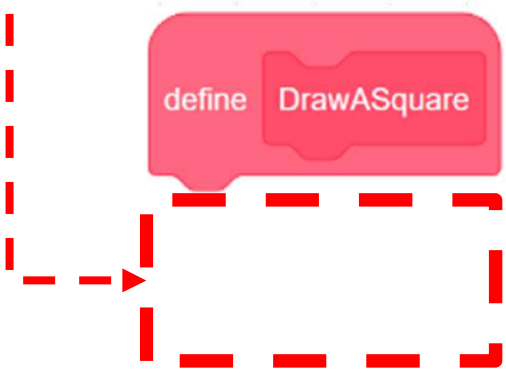
2. Type **"DrawASquare"** as the name of the block and then **"OK"**.



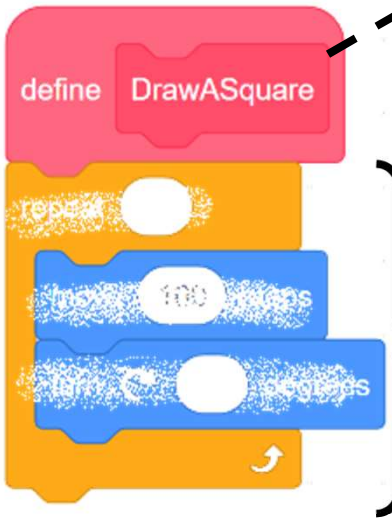
Drawing Shapes in Scratch

To Think and To Code: Create "DrawASquare" Block

After creating the own custom block, **DrawASquare**, you will now see a define DrawASqaure block.
Think about which blocks that draw a square and snap them to it. That means to **"define DrawASquare"**.



Knowledge builds up: Abstraction and Modularization



Abstraction is the process of identifying key information that is relevant in a given context and forgetting those details in that context.
For example, the name the custom block (procedure) "DrawASquare" captures its abstract key meaning of what to do later in the module.

Modularization is the process of organizing code for the task it executes and always try to keep the code reusable. It can be written once and be used in multiple places in the program. For example, we can use the "DrawASquare" module repeatedly to draw squares.

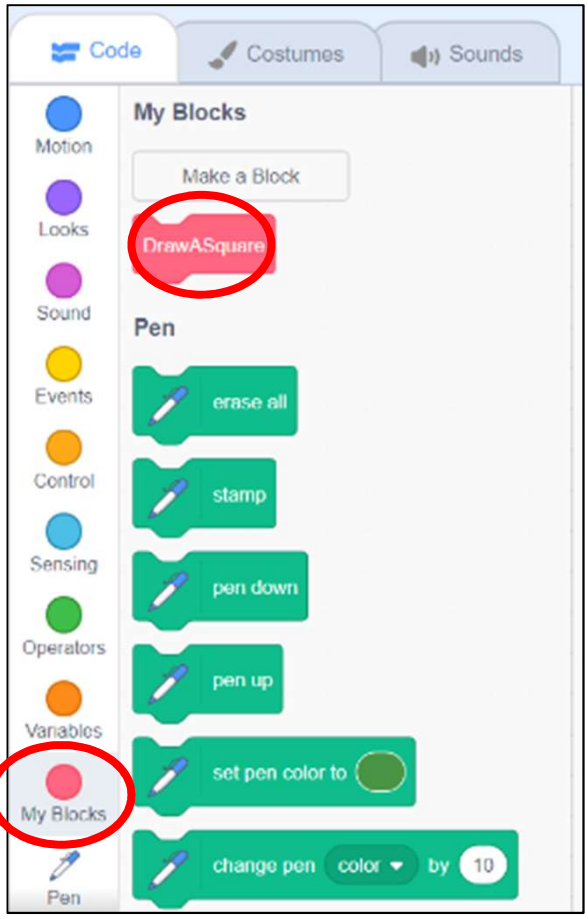
Drawing Shapes in Scratch

To Think and To Code: Create "DrawASquare" Block

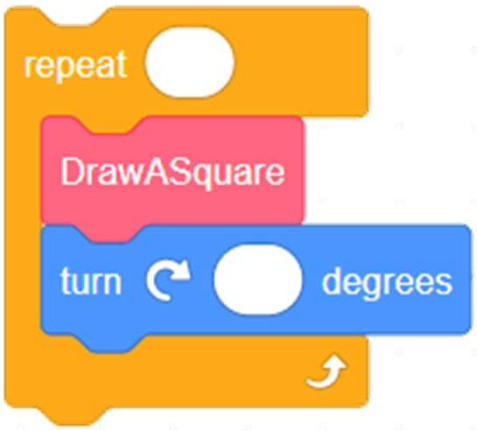
See Appendix P.32

Creating a custom block does not mean that it will run by itself; therefore we need to call it.

1. In the **"My Blocks"** drawer, find and drag the **"DrawASquare"** block out.



2. Then, with our custom block, **draw multiple squares.**



Testing and Debugging

When you finish, you can test it at once! Can you draw a snowflake with our custom block?

Drawing Multiple Squares → Create "DrawASquare" Block →

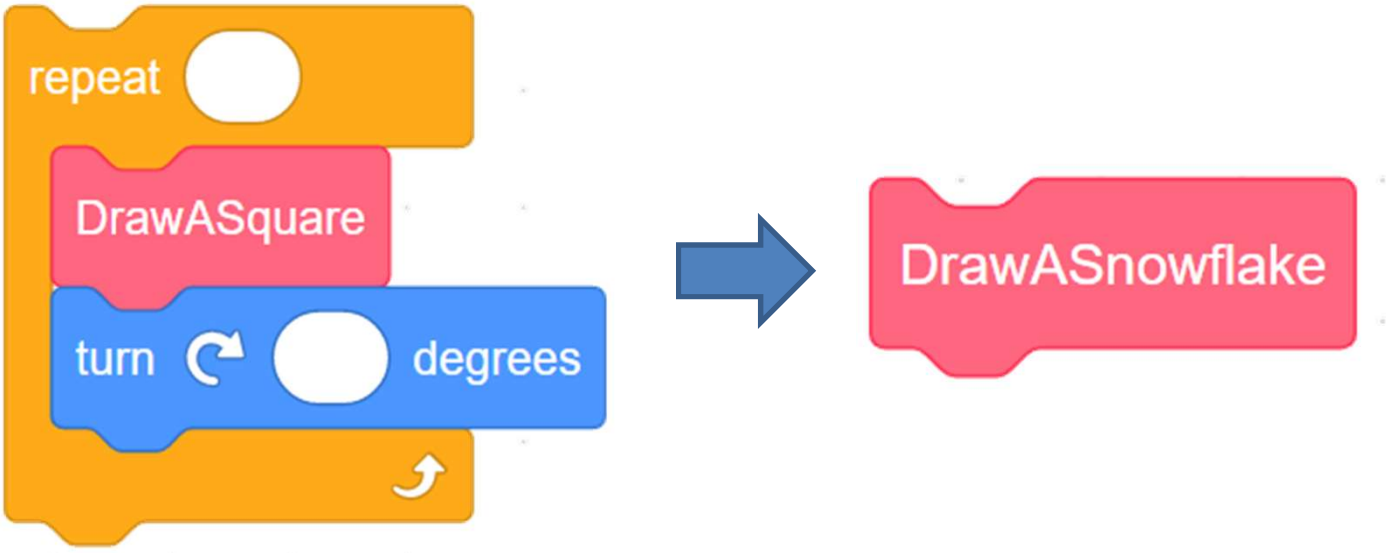
Drawing Shapes in Scratch

To Think and To Code: Create "DrawASnowflake" Block

See Appendix P.33

Finally, what if we want computer to "Draw a Snowflake"?

Review your code blocks and create another block named "**DrawASnowflake**" to do so.



Testing and Debugging

Click the green flag and test to see if the snowflake is still the same.

Drawing Multiple Squares → Create "DrawASquare" Block → Create "DrawASnowflake" Block

Drawing Shapes in Scratch

To Reflect: Two Stars and a Wish Worksheet

Name of Project: _____ Name of Creator: _____

Please write down two things that you like about this project.

1

_____

2

_____

What is one thing you would like to add or change to make this project better?

_____

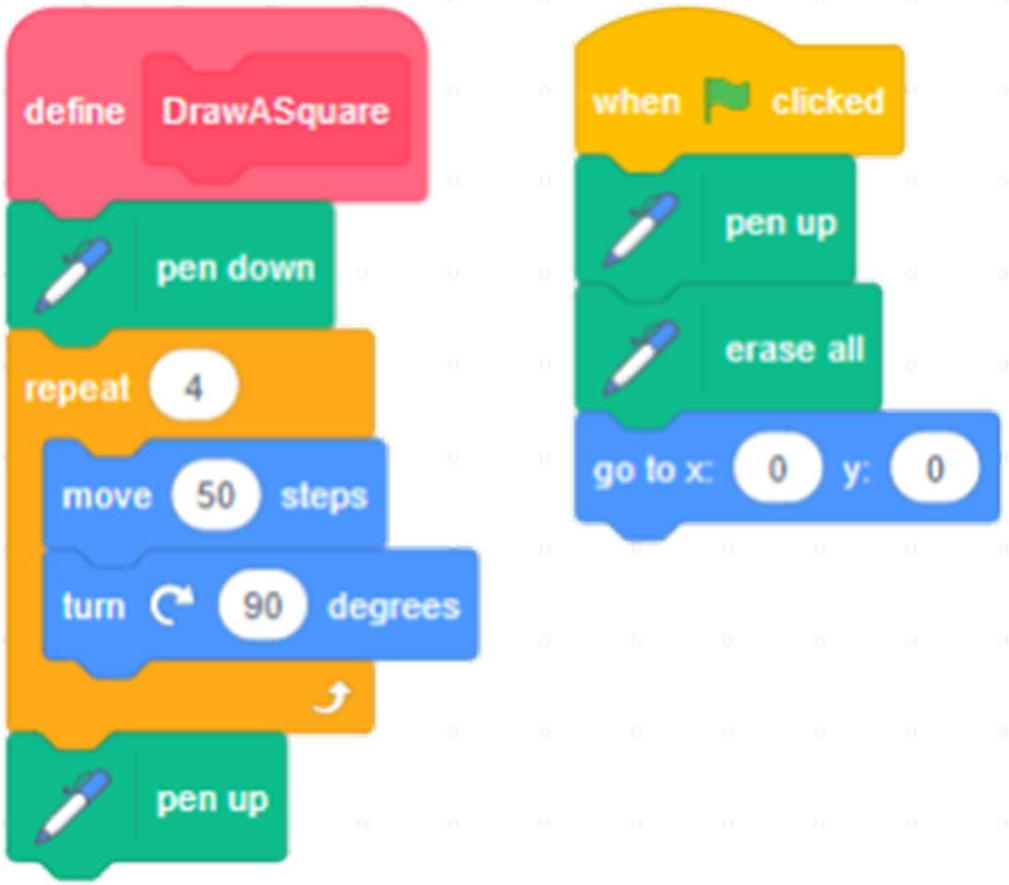
_____

_____

Drawing Shapes in Scratch

Review Questions

1. A student makes a project with the blocks above. What happens when the user clicks the green flag?

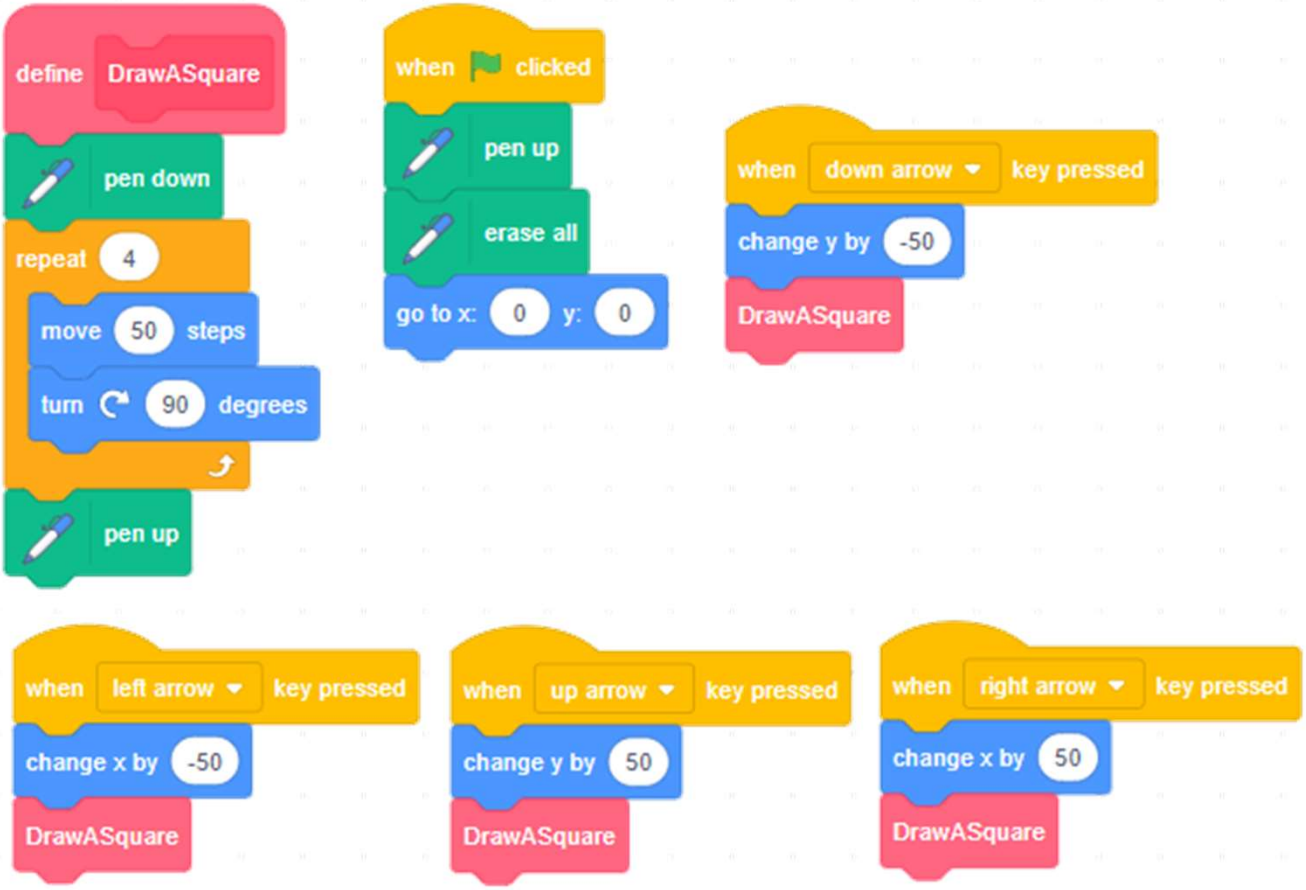


A. The sprite draws a snowflake.
B. The sprite draws four squares.
C. The sprite draws one square.
D. The stage is cleared and the sprite moves to the centre of the stage.

Drawing Shapes in Scratch

Review Questions

2. What happens if the user presses the right arrow 3 times?

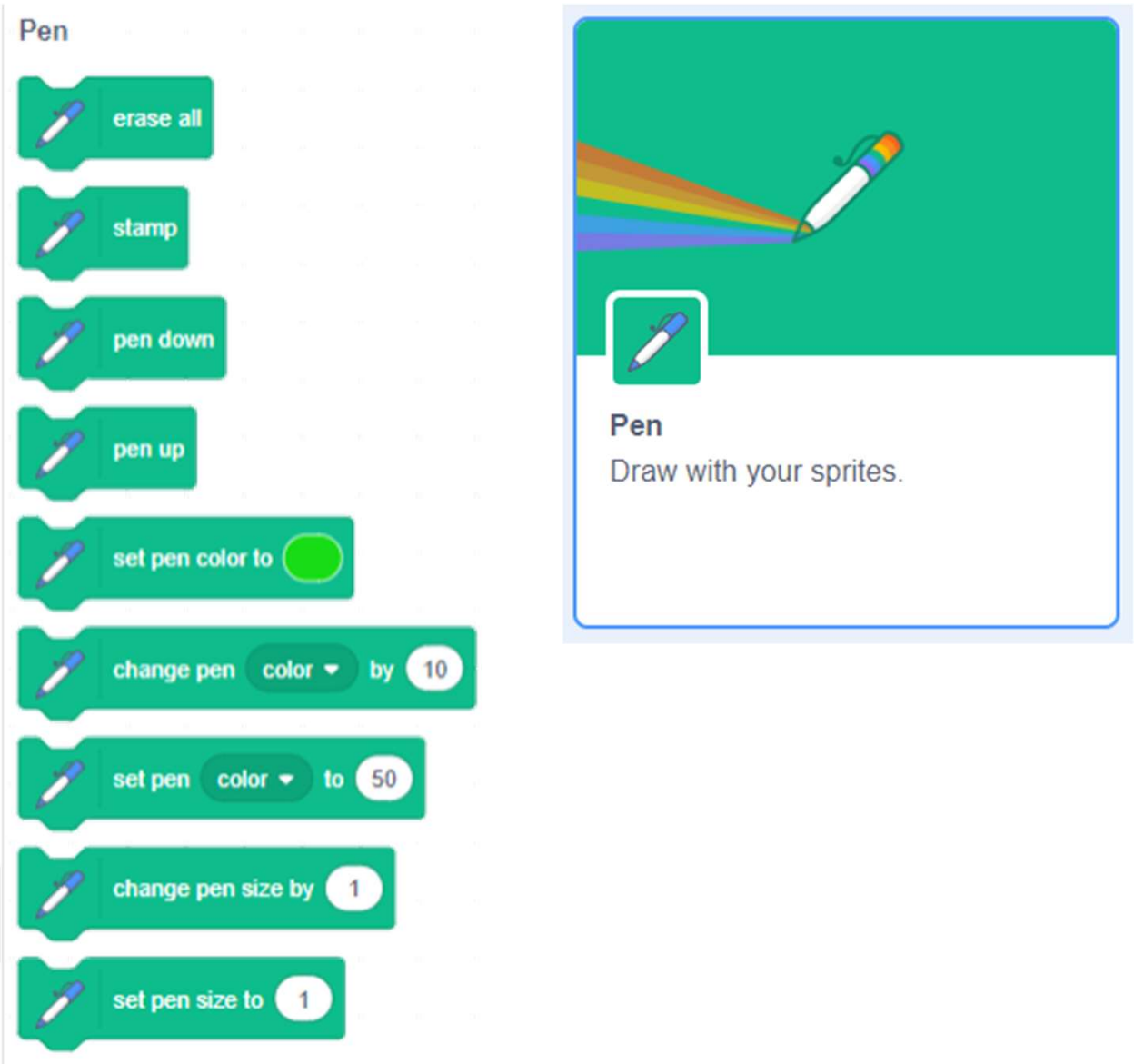


A. The sprite moves to the right 30 steps.
B. The sprite moves to the centre of the stage.
C. The sprite draws 3 squares that are beside each other on the stage.
D. The sprite draws 3 squares that are above each other on the stage.

Drawing Shapes in Scratch

Revision on Key Features

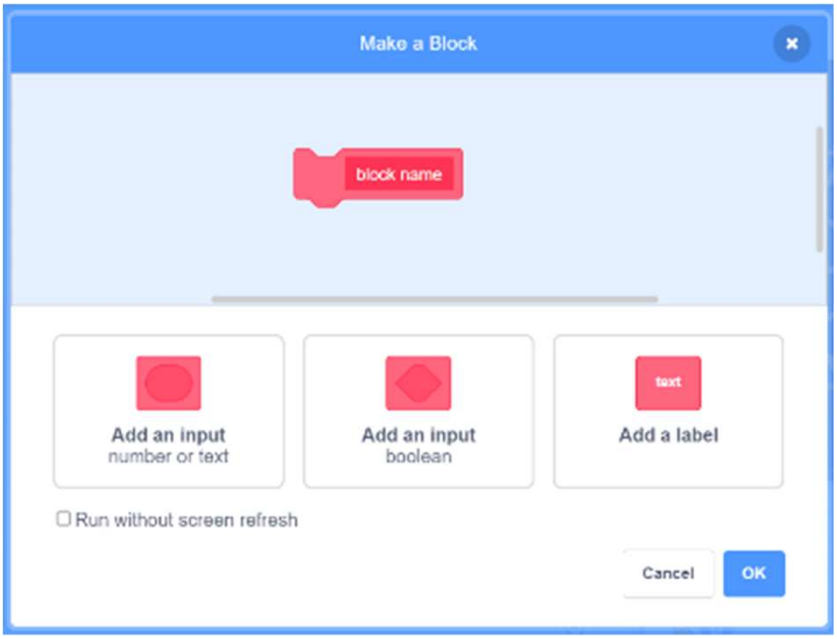
Pen: Turn the sprite (Scratch Cat) into a pen and draw on the stage freely.

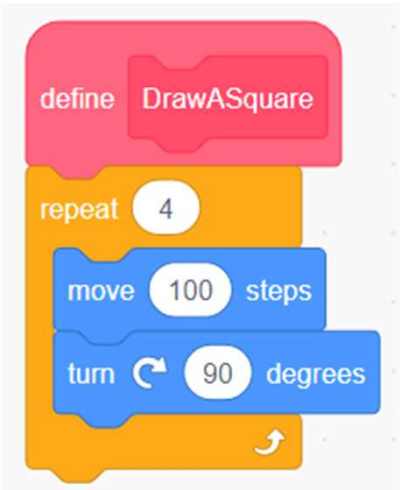


Drawing Shapes in Scratch

Revision on Key Features

My Blocks: create code blocks to modularize and abstract sequences of commands.

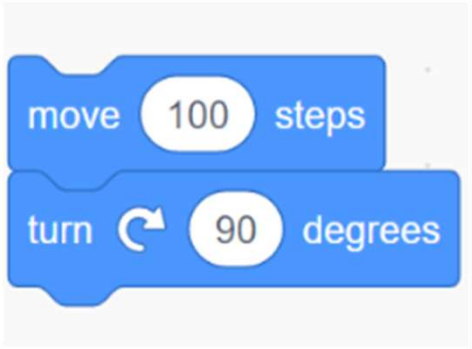




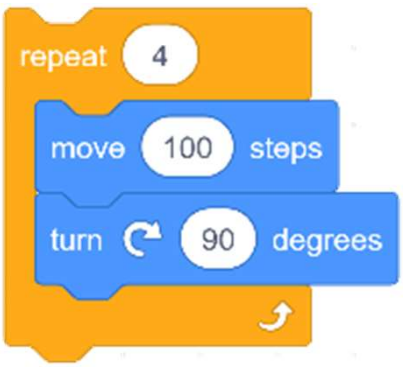
Drawing Shapes in Scratch

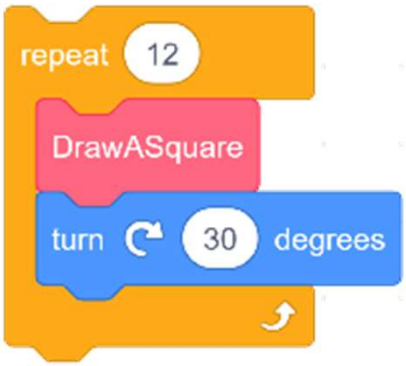
Revision on Key Concepts & Practices

Sequences: A series of steps that are executed by the computer one after the other in an order.



Iteration: Iteration is repeating a process in order to produce a sequence of outcomes. "Repeat ____" blocks can trigger iteration in Scratch.





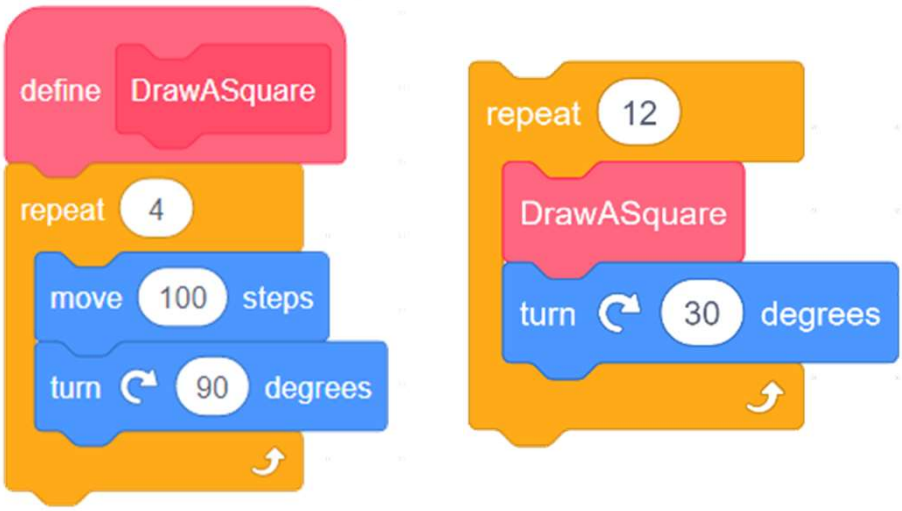
Drawing Shapes in Scratch

Revision on Key Concepts & Practices

Abstraction and Modularization:

In computer programming, Abstraction is the process of identifying key information that is relevant in a given context and forgetting those details in that context. For example, the name the custom block (procedure) “DrawASquare” capture its abstract key meaning of what to do later in the module.

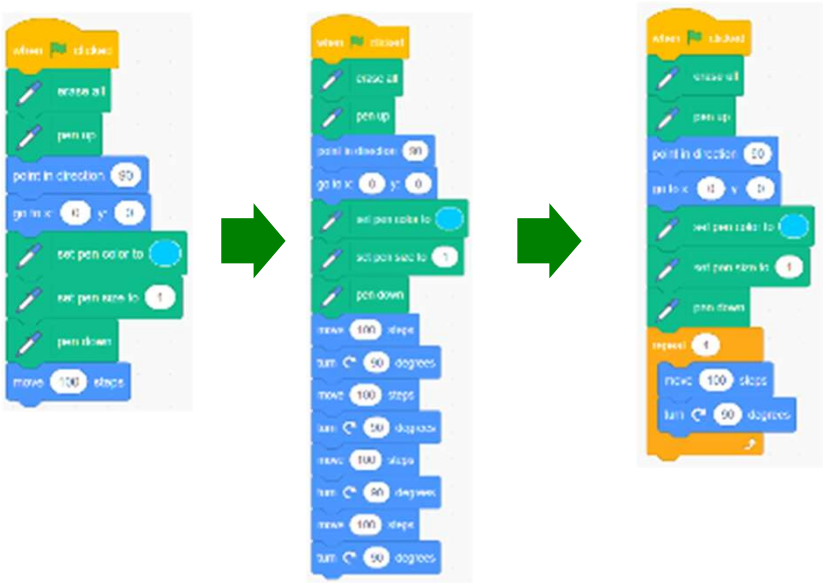
In computer programming, Modularization is the process of organizing code for the task it executes and always try to keep the code reusable. For example, we can use the “DrawASquare” module repeatedly to draw squares.



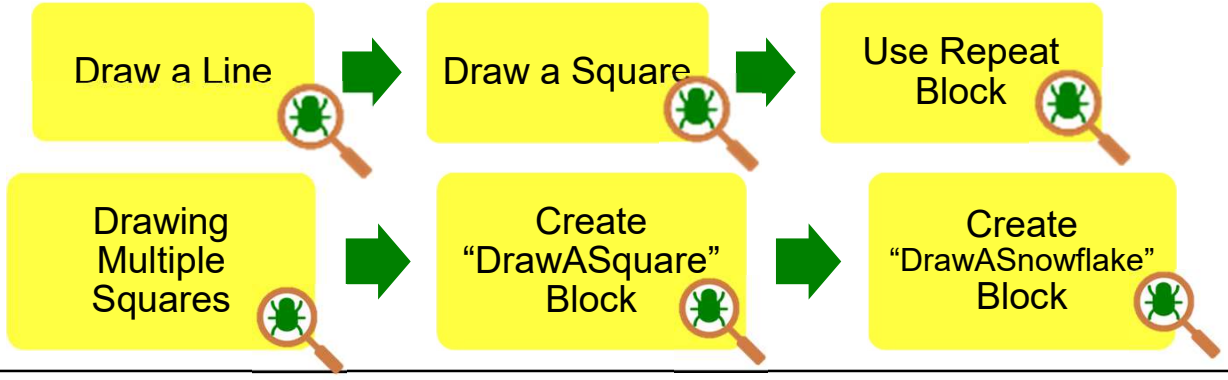
Drawing Shapes in Scratch

Revision on Key Concepts & Practices

Being incremental and iterative: to work out a sub-task as an iteration, try it out, then work out another sub-task in one more iteration until the whole programming task is completed.



Testing and Debugging: Testing a computer program is the process of checking if it can produce outcomes as designed. Debugging a computer program is the process of finding out ways to revise the program so that the bugs can be removed.



Appendix

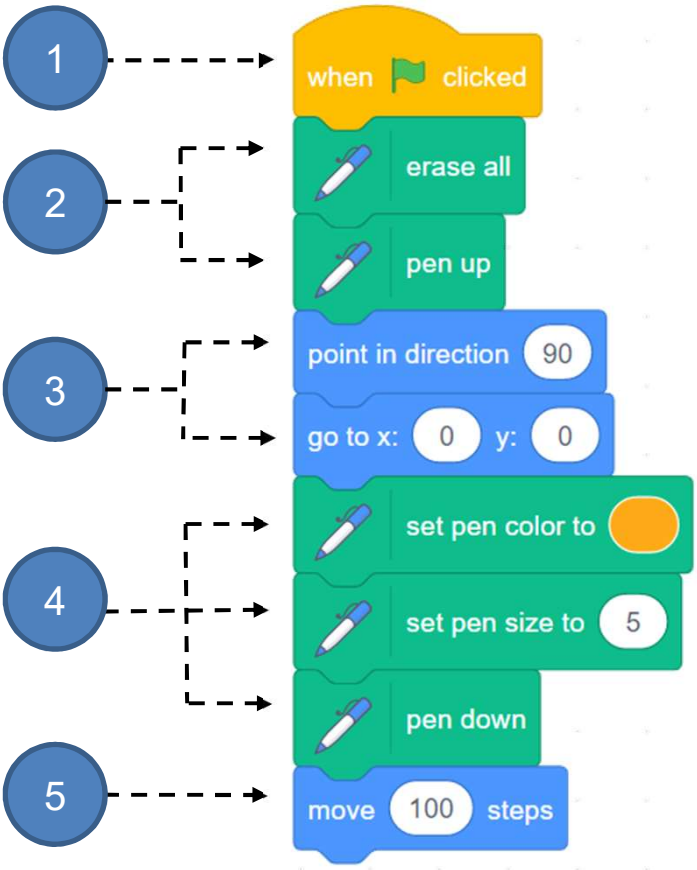
Operation Manual

Drawing Shapes in Scratch

To Code: Draw a Line

See Student Guide P.5

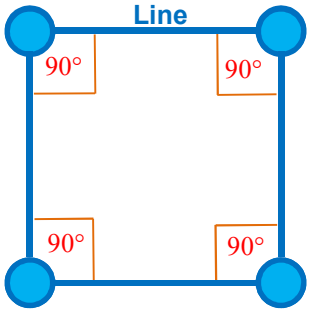
1. Drag the **"when green flag clicked"** block out from the **Events** drawer.
2. To clear the screen, drag out the **"erase all"** and **" pen up "** blocks from the **Pen** drawer and snap them to the **" when green flag clicked "** block. The **" pen up "** block prevents the sprite from drawing while its moving.
3. To get the sprite to its starting position, drag out the **"point in direction 90 "** and **" go to x:0 y:0 "** blocks from the **"Motion"** drawer and snap to the above blocks.
4. Drag out the **"set pen color to", " set pen size to"** and **"pen down"** blocks from the **Pen** drawer. Snap them to the above blocks to get the sprite ready to draw.
5. Add a **"move 100 steps"** block from the **"Motion"** drawer to draw the line.



Drawing Shapes in Scratch

To Think and To Code: Draw a Square

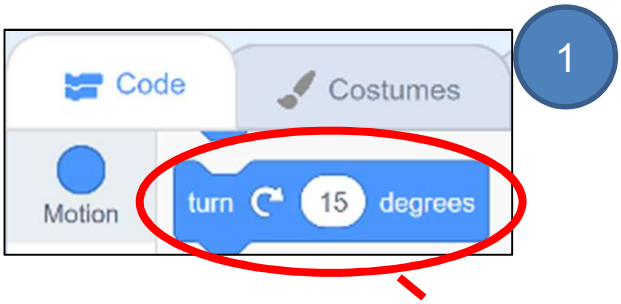
A square is created by joining four lines at right angles (90 degrees).



See Student Guide P.6

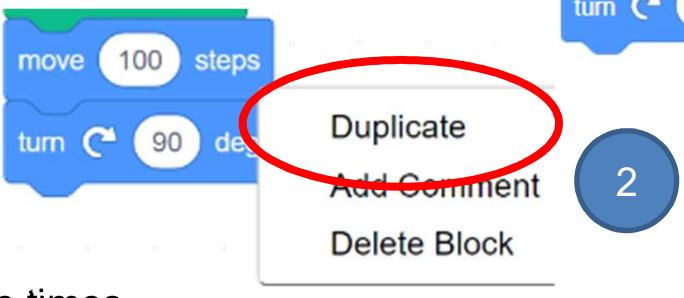
To draw a square, you need to turn 90 degrees before you start to draw the next line.

1. Snap a **turn right 90 degrees** block from the **Motion** drawer to the **move 100 steps** block.



Change to 90 degrees

2. Right-click on the **move 100 steps** block to duplicate the **move 100 steps** and **turn right 90 degrees** blocks. Snap the duplicate blocks below the blocks created in step 1.



3. Repeat **step 2** two more times.

Drawing Shapes in Scratch

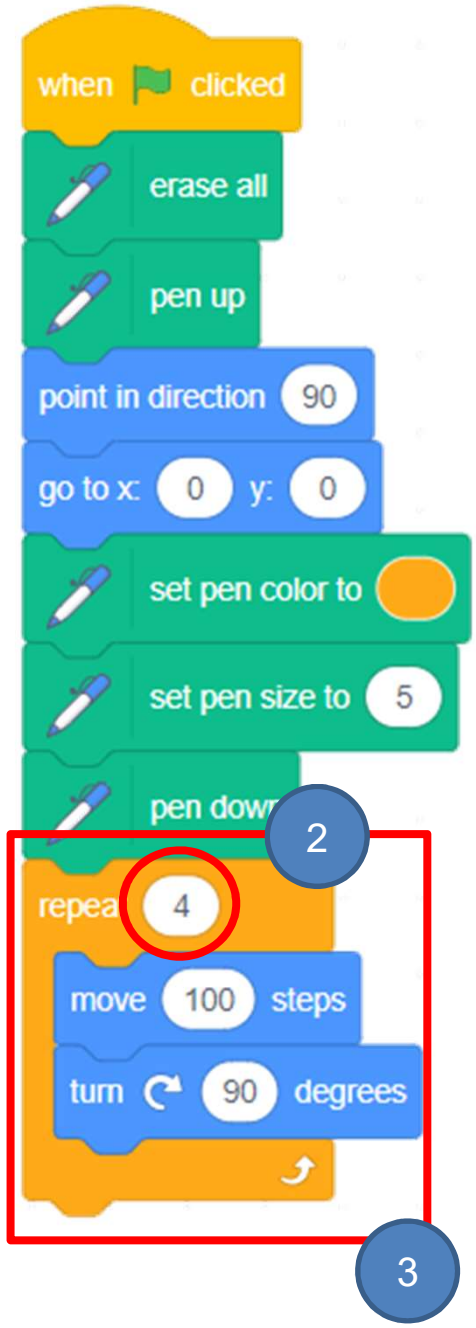
To Think and To Code: Use Repeat Block

See Student Guide P.7

1. Remove the three sets of duplicate blocks: **move 100 steps** and **turn right 90 degrees**.

2. Drag a **repeat** block from the **Control** drawer and change the number to “4”.

3. Snap the remaining **move 100 steps** and **turn right 90 degrees** blocks into the **repeat** block.



Drawing Shapes in Scratch

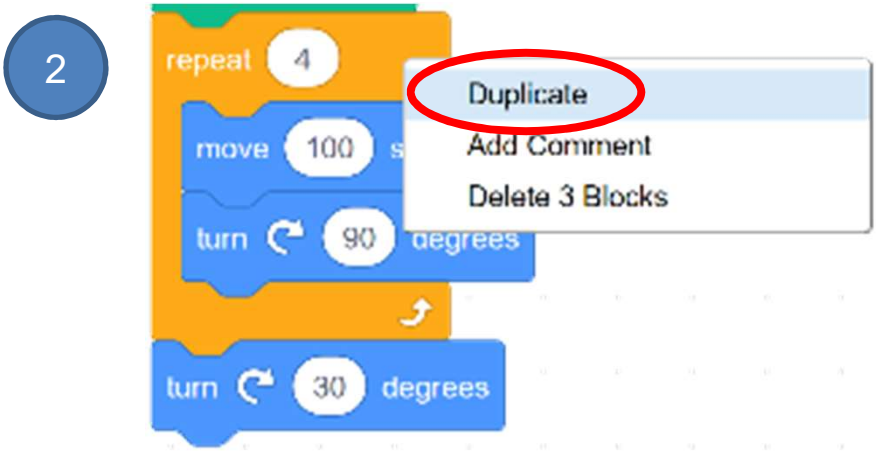
To Think and To Code: Draw Multiple Squares

To draw a snowflake with multiple squares, you need to:

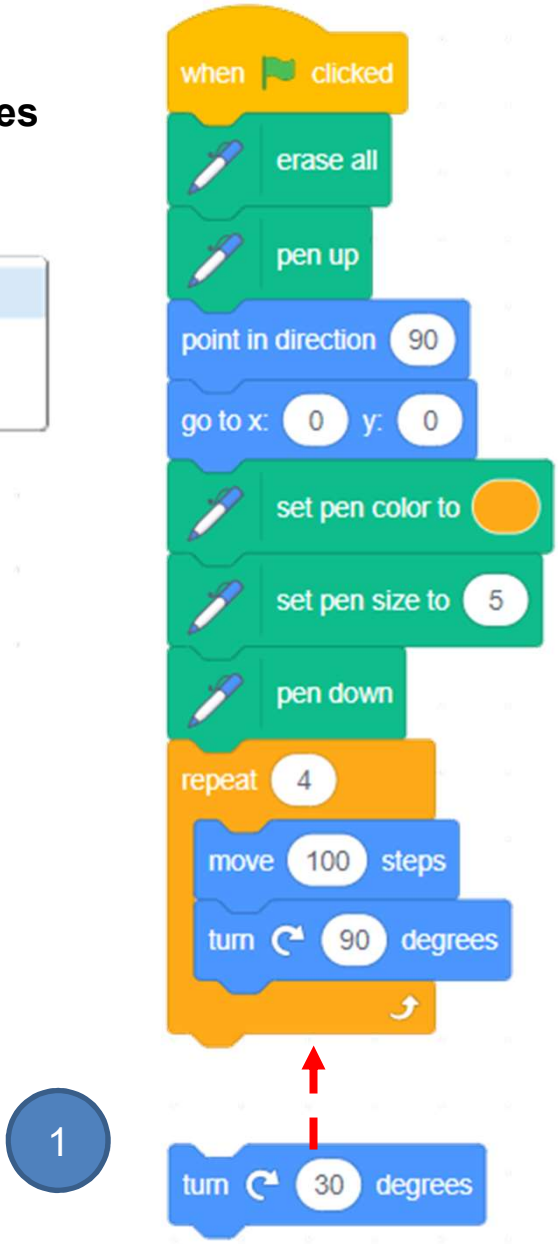
See Student Guide P.11

1. Drag the **turn right 15 degrees** block from the **Motion** drawer. Snap it below the **repeat** block and change it to **30** degrees.

2. Duplicate the **repeat** and **turn right 30 degrees** blocks.



3. Snap the duplicate blocks to the **turn right 30 degrees** block.



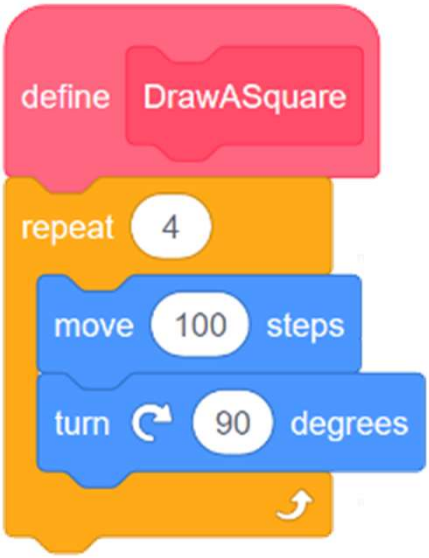
Repeat the steps until you make a snowflake.

Drawing Shapes in Scratch

To Think and To Code: Create "DrawASquare" Block

See Student Guide P.17

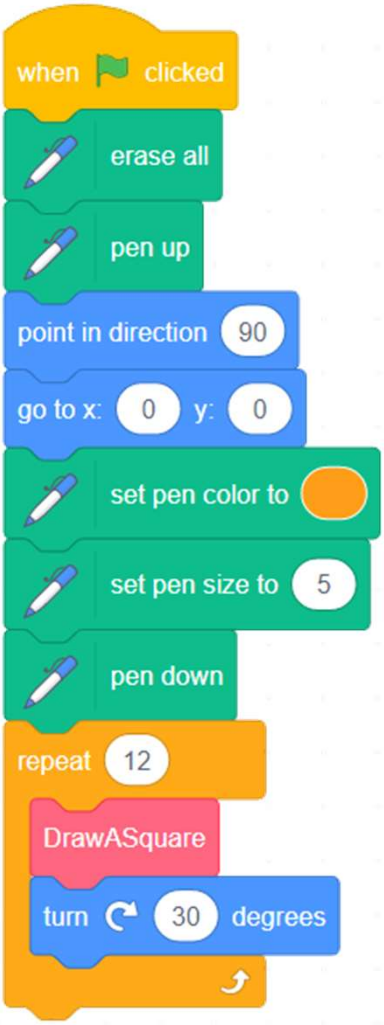
Drag the blocks that draw a square and snap them to the DrawASquare block.



1. And you need to draw 12 squares to form a snowflake. So, drag a **repeat** block from the **Control** drawer and snap the **DrawASquare** and **turn right 30 degrees** blocks in. Remember to change the number to **12** to draw 12 squares.

2. Before you draw a new square, the sprite needs to turn 30 degrees. So, snap the **turn right 30 degrees** block in the **DrawASquare** block.

3. Drag the **repeat** block set to the end of the when green flag clicked blocks.

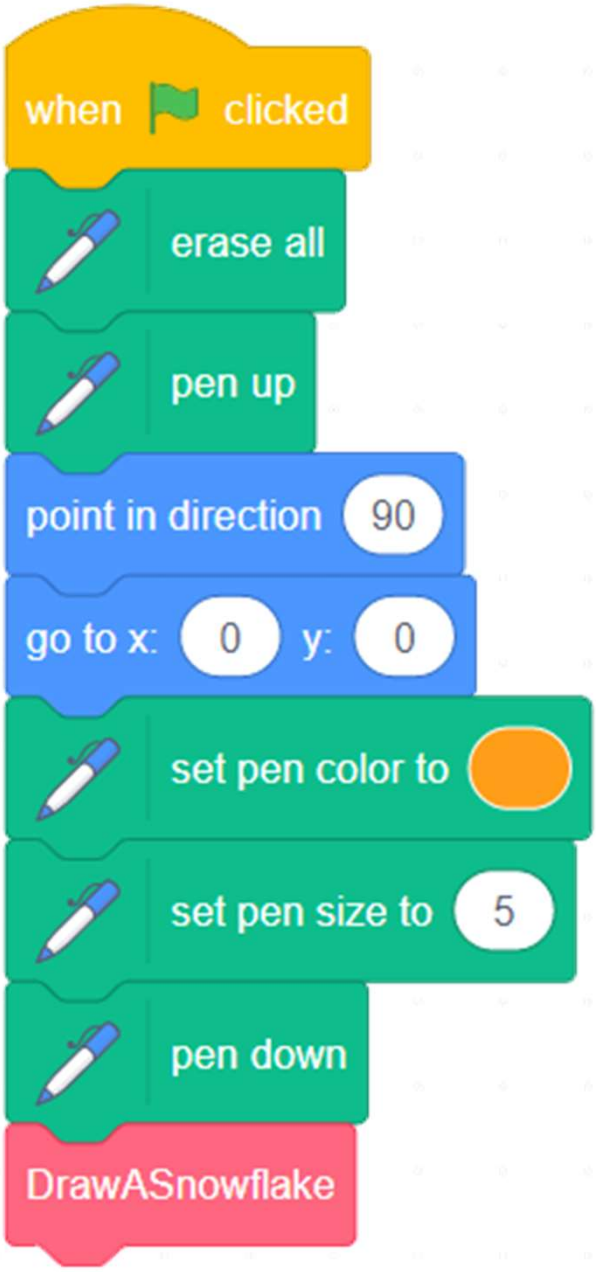


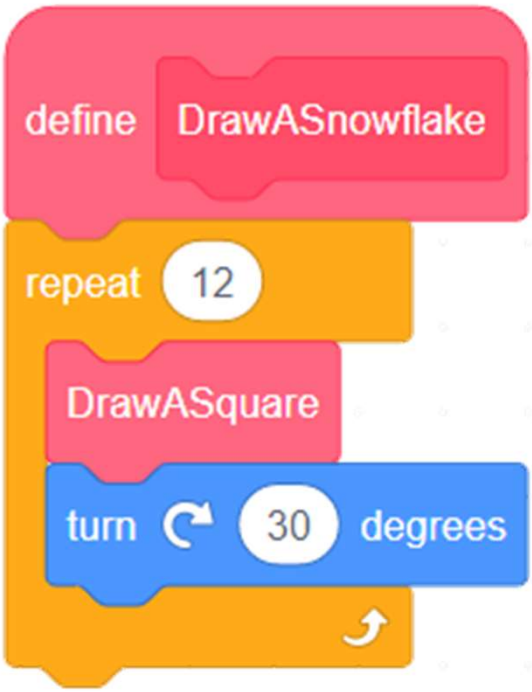
Drawing Shapes in Scratch

See Student Guide P.18

To Think and To Code: Create "DrawASnowflake" Block

Create and use the "DrawASnowflake" block to make the program code clearer.



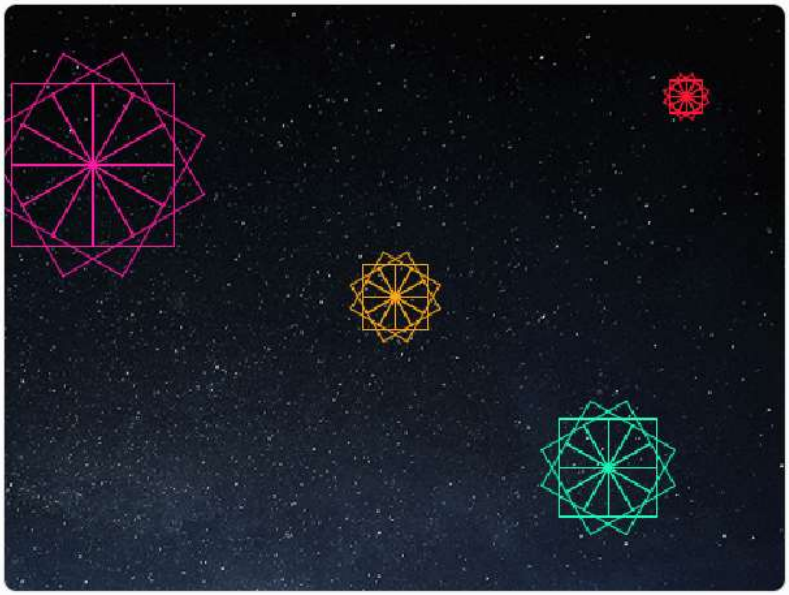


Unit 8: Designing Patterns in Scratch
(Extension Unit)
Student Guide

Content

Designing Patterns in Scratch

In this unit, you will learn how to make more snowflakes in different locations on the stage. Also, you will change the size of your snowflakes in your art project.



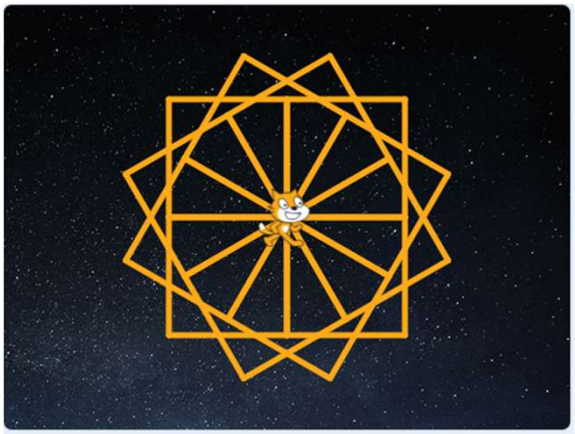
To Play

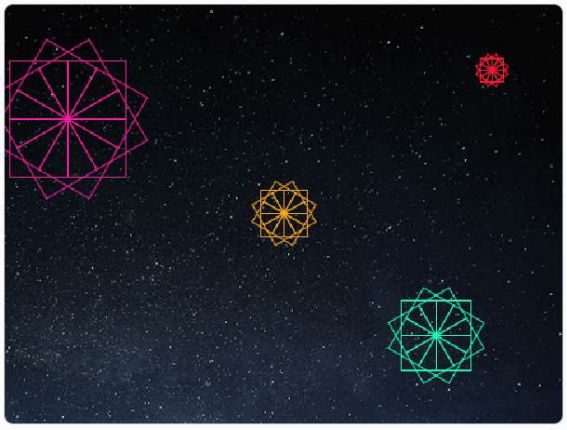
Open the Scratch Project https://scratch.mit.edu/projects/737985288/

Click the green flag and see what will happen.

Click again to see if those snowflakes go to the same way.

Did you see the Scratch cat this time?

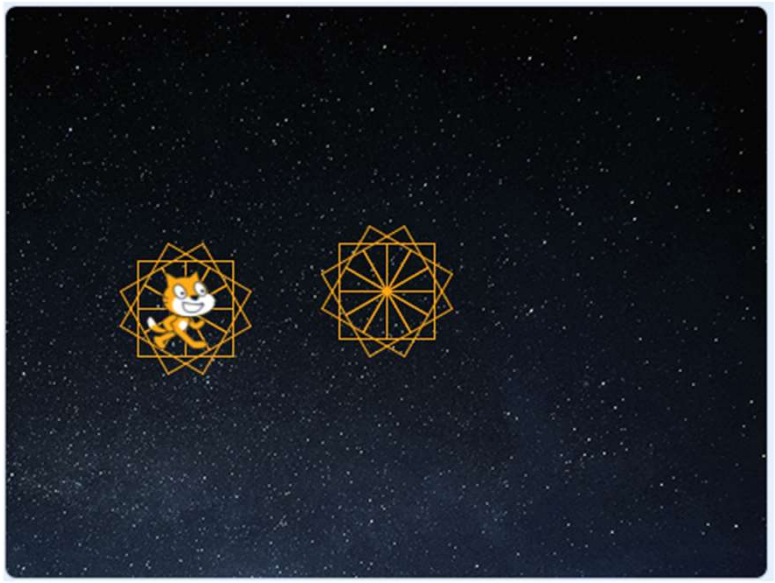




Designing Patterns in Scratch

To Think

Do you have any ideas for how to make multiple snowflakes in different locations on the stage?



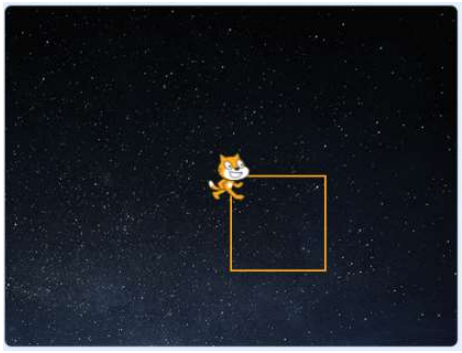
1. Sign into your account at scratch.mit.edu. Go to "My Stuff" under your name at the right top of the screen and open your Unit 7 "DrawASnowflake" project.

2. Select "Save as a copy" from the "File" menu. Then change the name to "DrawMultipleSnowflakes" and save your project.

Designing Patterns in Scratch

To Think: Draw a Square

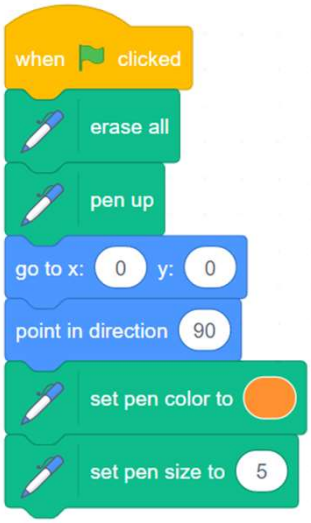
Review what you learned in the previous unit, do you remember how to draw a square?



A. Turn 90 degrees	B. Draw a Line

When Green Flag Clicked

Preparation:
1. Erase all
2. Pen up
3. Starting position
4. Point in direction 90
5. Set pen color and size

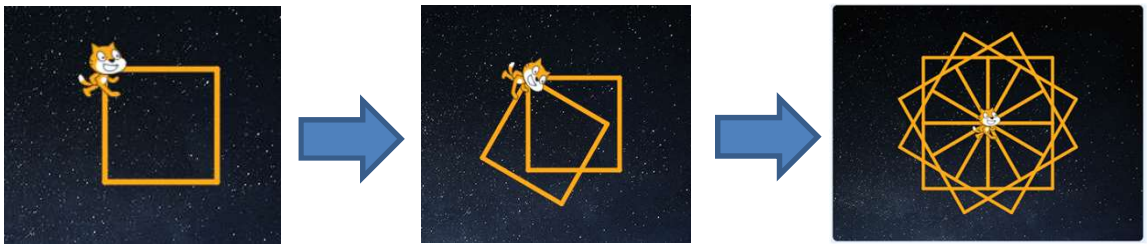


Repeat 4 Times

Designing Patterns in Scratch

To Think: Draw a Snowflake

Try to extend “Draw a Square” to “Draw a Snowflake”.



A. Turn 30 degrees	B. Draw a Square

When Green Flag Clicked

Preparation:
1. Erase all
2. Pen up
3. Starting position
4. Point in direction 90
5. Set pen color and size

Repeat 12 Times

Designing Patterns in Scratch

To Think: Draw More Snowflakes

How to draw more snowflakes?

A. Go to Random Position	B. Draw a Snowflake	C. Change Pen Color

When Green Flag Clicked

Preparation:
1. Erase all
2. Pen up
3. Starting position
4. Point in direction 90
5. Set pen color and size

Repeat 4 Times

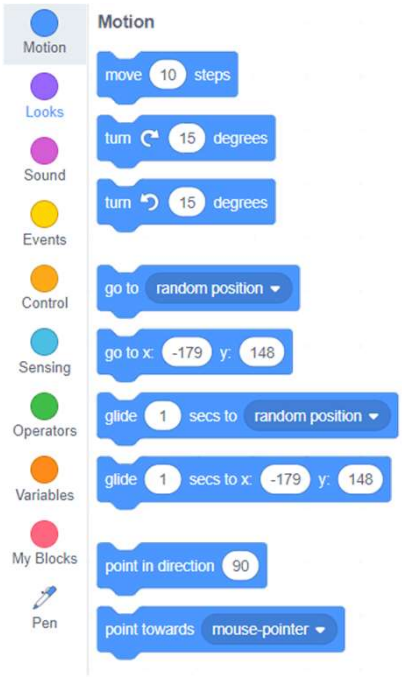
Designing Patterns in Scratch

To Code: Draw More Snowflakes

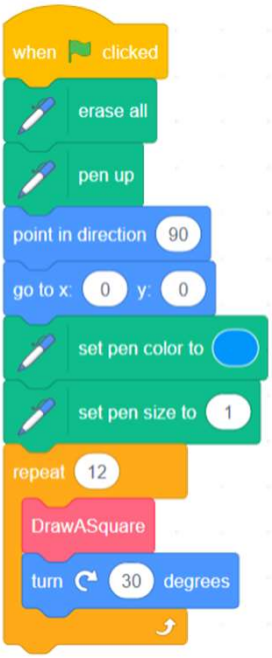
After drawing one snowflake, the sprite should move to a new position.
Code to do it!

See Appendix P.21

Hint: Check out the "Motion" drawer.



How would you draw a second snowflake once the sprite has moved to a new position?



Hint: Repeat the blocks to add more snowflakes.

Testing and Debugging

Click on the green flag and see if the snowflakes are drawn in different positions. Then look at those blocks. Can you make them shorter?

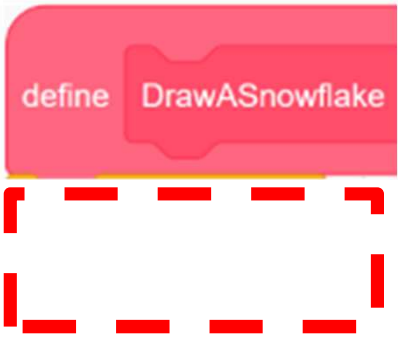
Draw More Snowflakes

Designing Patterns in Scratch

To Think and To Code: Use DrawASnowflake Block

See Appendix P.22

You should have some repeated steps to draw
Review the "DrawASnowFlake" block that we learnt in the previous unit.

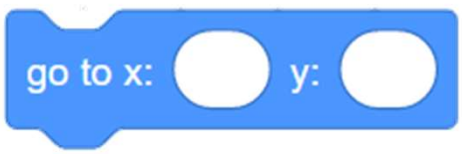


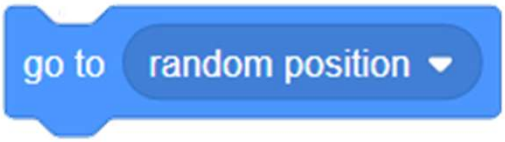
Can you use the "DrawASnowFlake" block to simplify the coding and draw more snowflakes this time?

To draw multiple snowflakes in different locations on the stage, you need to move the sprite to different positions.

Where do you want to go? Certain X / Y or random position?

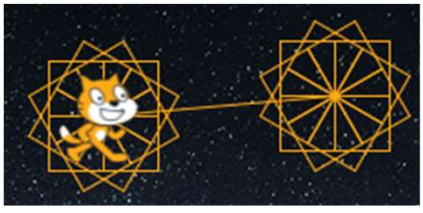
Hint:





Testing and Debugging

Does the simplified blocks do the same thing?
Do you see there is a line between two snowflakes? Why does it happen?



Draw More Snowflakes → Use DrawASnowflake block →

Designing Patterns in Scratch

To Think and To Code: Pen Up & Down

Why is there a line between two snowflakes?

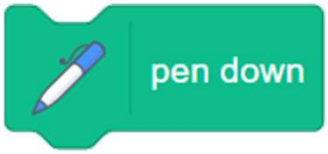
See Appendix P.23

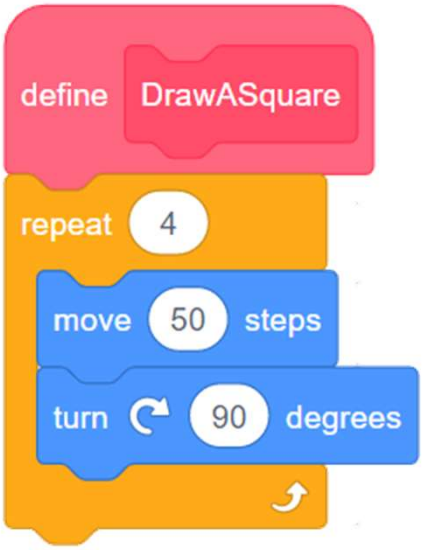
Does the pen always keep “pen down” ?

When drawing a square, keep “pen down”; after drawing, keep “pen up” .

Go to the “Pen” drawer, which block(s) can help you out?
Then update the “DrawASqaure” block.

Hint:





Testing and Debugging

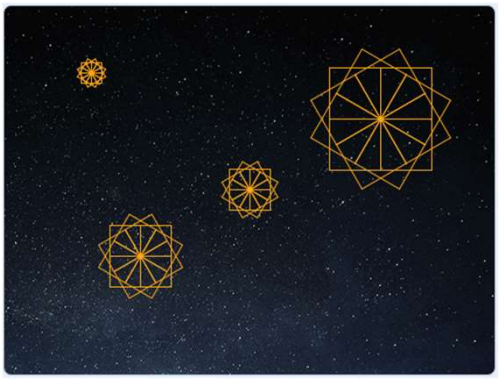
Click to test it! After drawing a snowflake, does the line disappear this time?

Draw More Snowflakes → Use DrawASnowflake block → Pen Up & Down

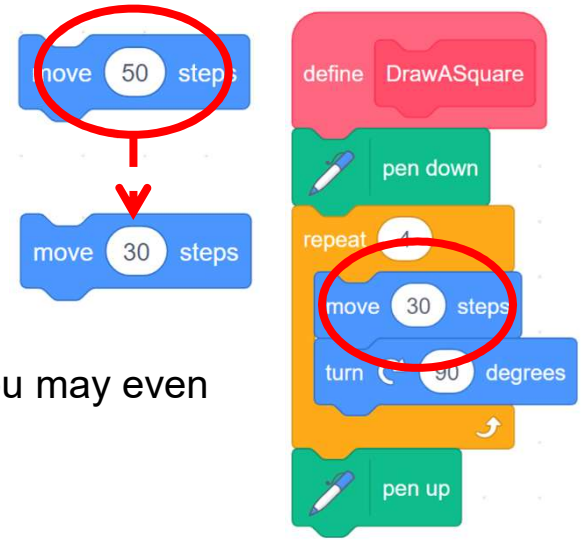
Designing Patterns in Scratch

To Think and To Code: Draw Different Sized Snowflakes

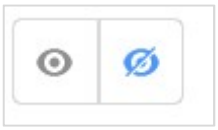
Think about how to make snowflakes of different sizes in different locations?



To make various sizes of snowflakes, we will need to change the size of the squares that make up the snowflakes.



1. In the "define DrawASquare" block", change the number of steps from 50 steps to "move (e.g. 30) steps".
2. Click on the green flag and see what is drawn.
3. If you find the pattern is too small to see, you may even hide the sprite.



Testing and Debugging

If you change the square size by changing the number of "move ( ) steps" block, this will affect the sizes of all snowflakes.

Draw Different Sized Snowflakes

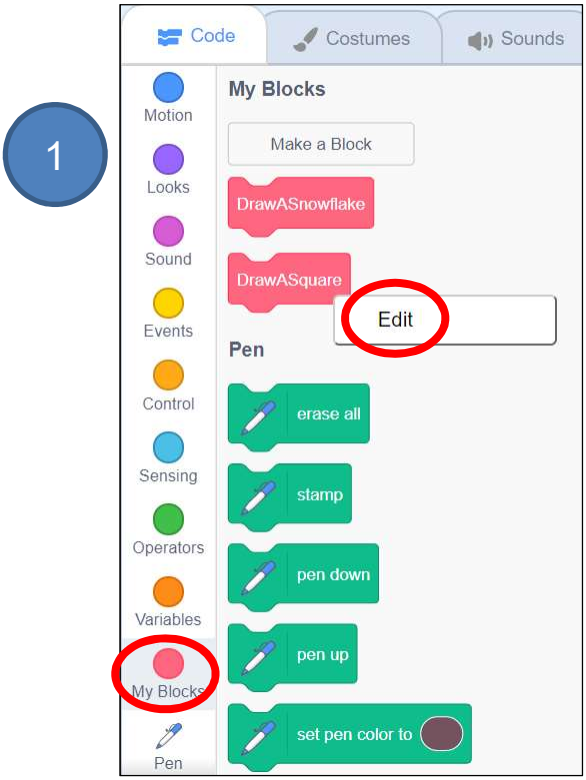
Designing Patterns in Scratch

To Think and To Code: Add an Input (number or text) for Custom Blocks

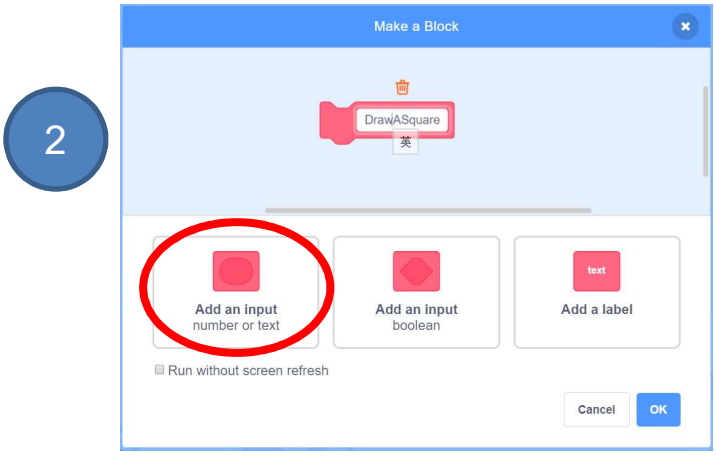
How can you draw each snowflake in a different size? Do you have any ideas? You can create an input parameter for your custom block to create various sizes of squares and snowflakes.

To create an input parameter for your "DrawASquare" block, you need to:

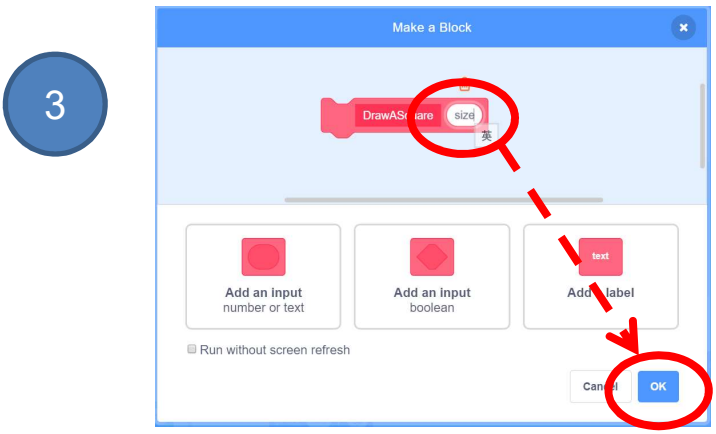
1. Right click the "DrawASquare" block in the "My Blocks" drawer. Click on "Edit".



2. Then click " Add an input ".



3. Type "size" to name the input parameter, then click the "OK" button to save the "DrawASquare" block.

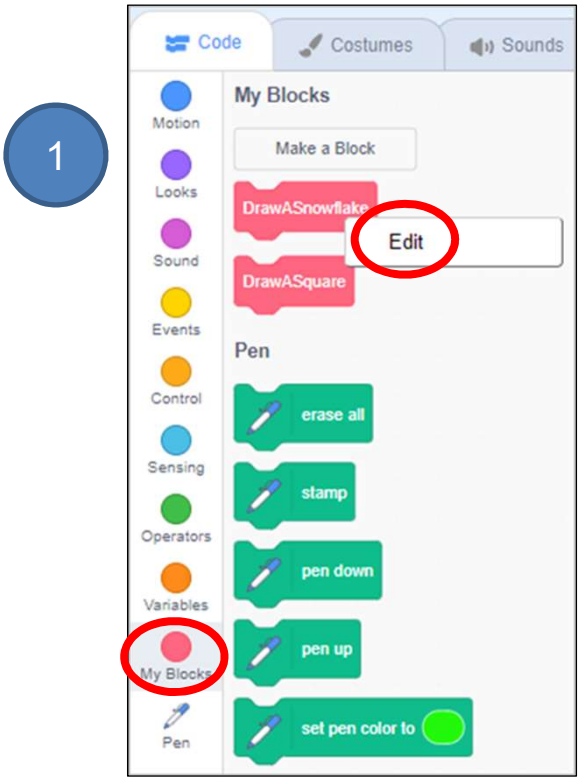


Designing Patterns in Scratch

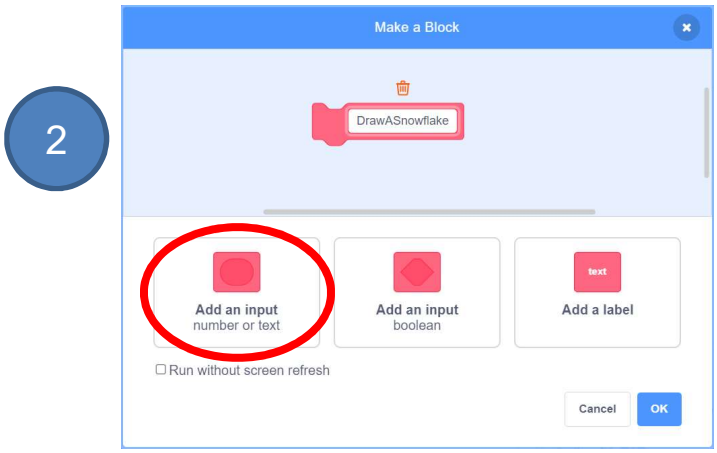
To Think and To Code: Add an Input (number or text) for Custom Blocks

Let's do the same with our "DrawASnowflake" block by adding an input parameter.

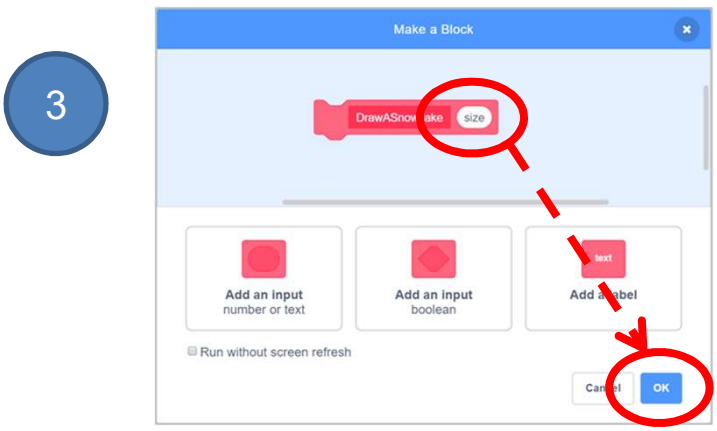
1. Right click the "DrawASnowflake" block in the "My Blocks" drawer. Click on "Edit".



2. Then click "Add an input ".



3. Type "size" to name the input parameter, then click the "OK" button to save the "DrawASnowflake" block.



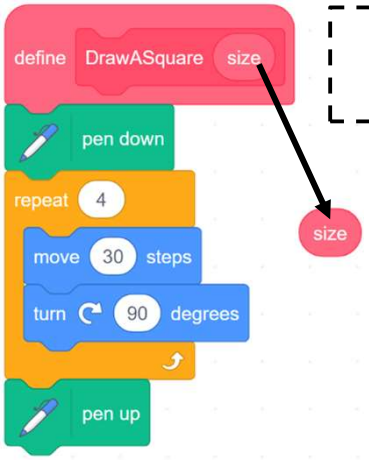
Designing Patterns in Scratch

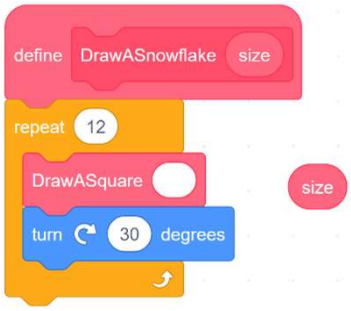
To Think and To Code: Add an Input (number or text) for Custom Blocks

Set the size for "DrawASquare" and "DrawASnowflake" blocks.

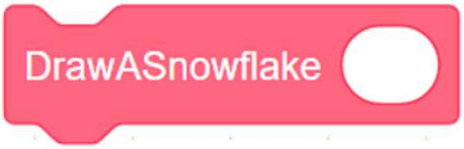
See Appendix P.24

1. Drag " size " from " DrawASquare (size) " out. Where should we put the " size " block? Which block should be replaced?





2. You should also update the " DrawASnowflake (size) " too.
3. Choose different size values for the " DrawASnowflake ( ) " block when you use it to draw different sized snowflakes.



Testing and Debugging

Click on the green flag and see what is drawn.
Now, can you add three more snowflakes to your project with different sizes and locations?

Draw Different Sized Snowflakes → Add an Input (number or text) for Custom Blocks

Designing Patterns in Scratch

To Reflect: Two Stars and a Wish Worksheet

Name of Project: _____ Name of Creator: _____

Please write down two things that you like about this project.

 1

_____

 2

_____

What is one thing you would like to add or change to make this project better?

_____

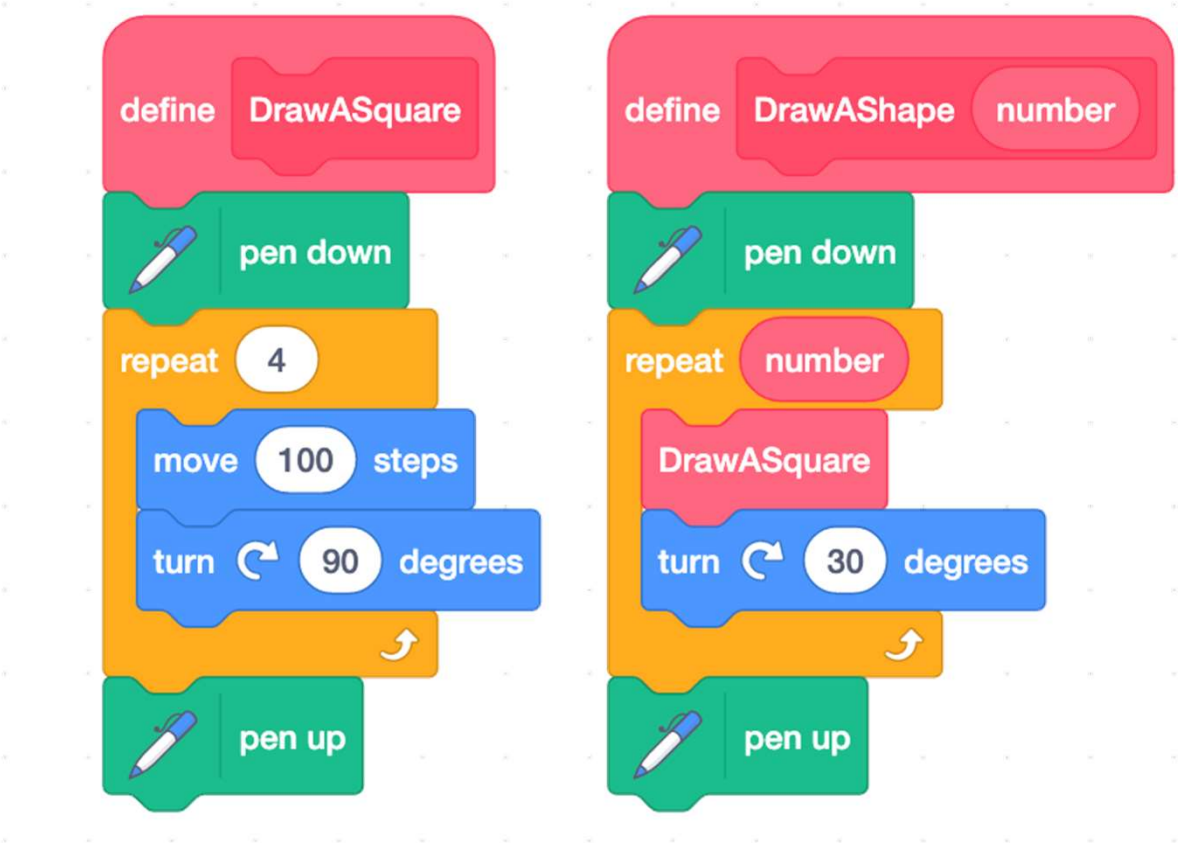
_____

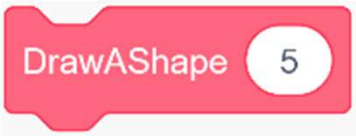
_____

Designing Patterns in Scratch

Review Questions

1. A student makes a project with the following custom blocks.



If the student calls 

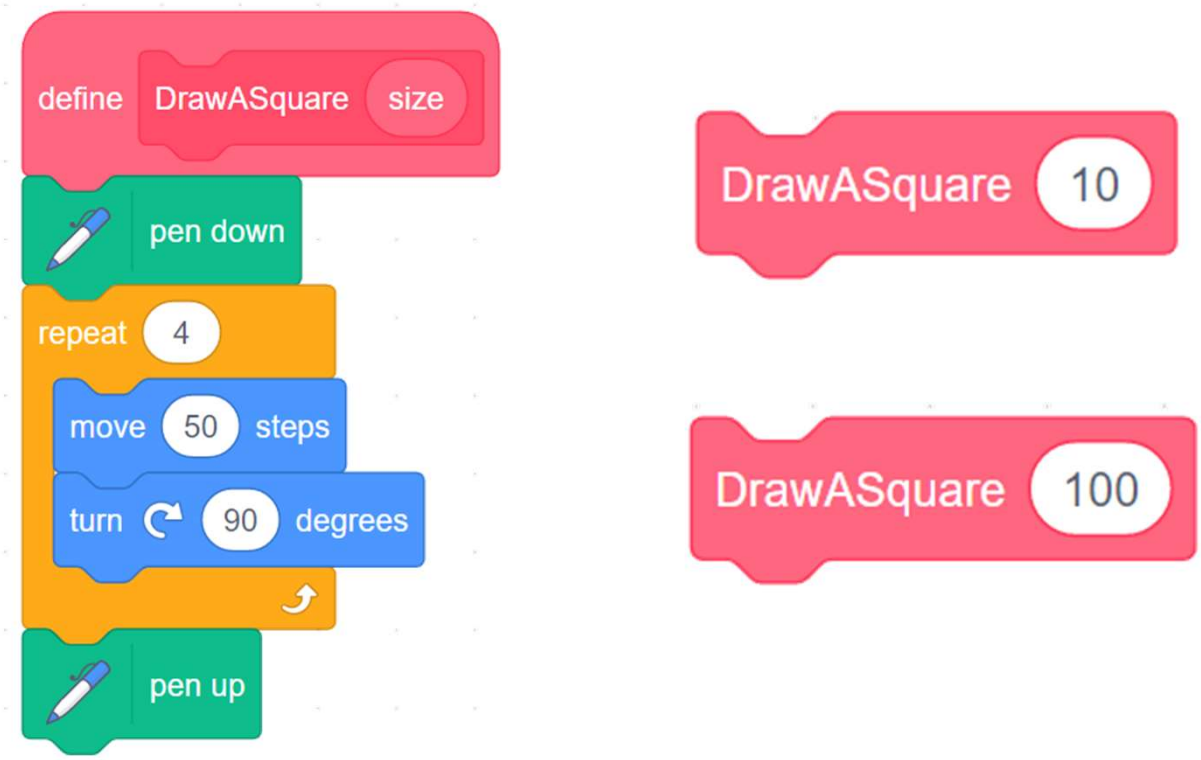
, what happens?

A. A snowflake with a size of 5 is drawn.
B. 5 snowflakes are drawn.
C. A shape with 5 rotated squares is drawn.
D. Nothing is drawn because the pen is up.

Designing Patterns in Scratch

Review Questions

2. A student failed to draw squares with different sizes no matter he called DrawASquare (10) or DrawASquare (100). Can you help debugging?

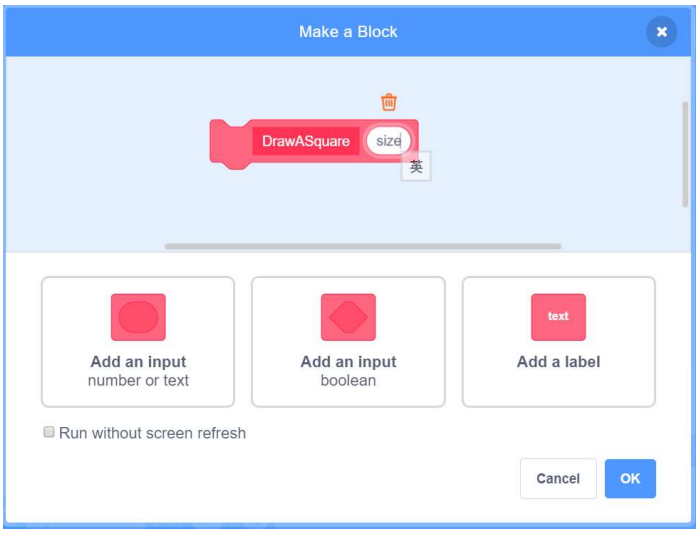


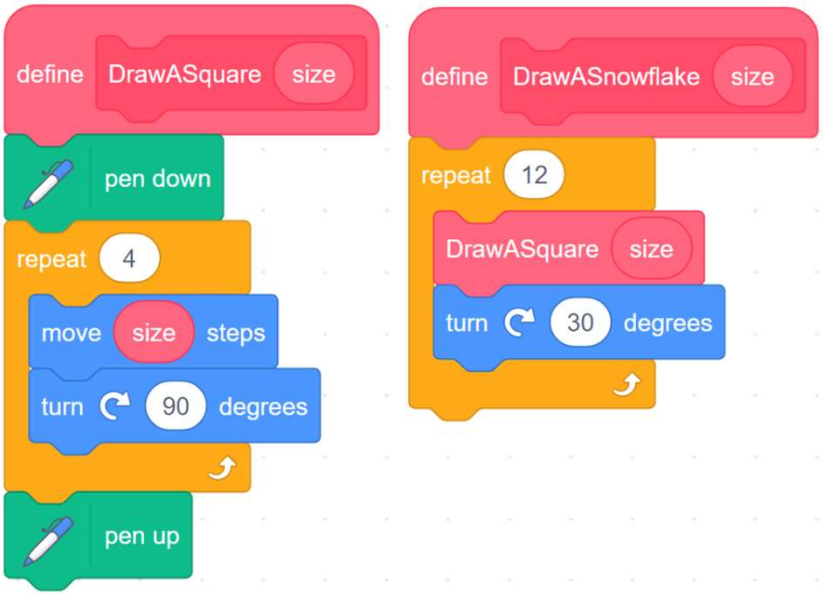
A. Rename the block from "**DrawASqaure**" to "**DrawASnowflake**".
B. Remove the "**size**" parameter inside the "**DrawASqaure**" block.
C. Repeat "**size**" but not "**4**".
D. Replace "**50**" in the "**move 50 steps**" block with the "**size**" parameter.

Designing Patterns in Scratch

Revision on Key Features

My Blocks with Parameter: With the passing parameter “size”, we can create different sized squares and snowflakes.

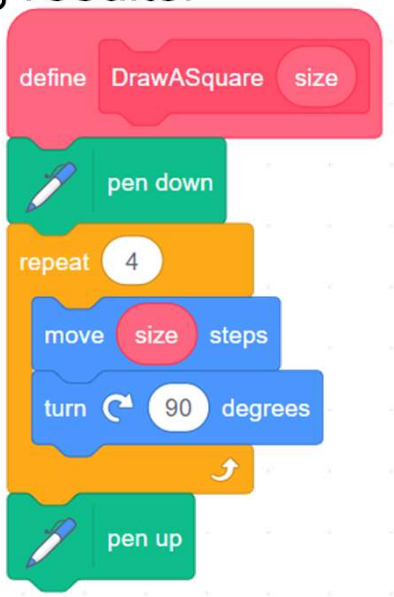




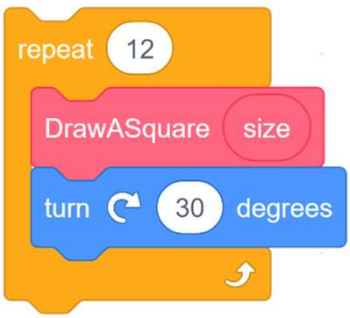
Designing Patterns in Scratch

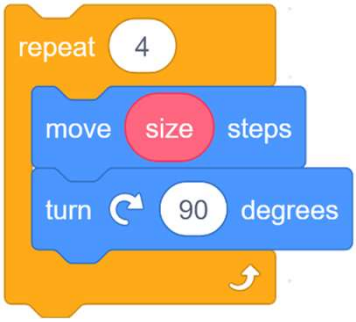
Revision on Key Concepts & Practices

Sequence: It is the order in which the programming statements are executed. A wrong order would lead to incorrect programming results.



Iteration: Iteration is repeating a process in order to generate a sequence of outcomes. “Repeat___” block can trigger iteration in Scratch.





Designing Patterns in Scratch

Revision on Key Concepts & Practices

Abstraction and modularization: In computer programming, abstraction is the process of identifying key information that is relevant in a given context and forgetting those details in that context. For example, the name the custom block (procedure) “DrawASnowflake” capture its abstract key meaning of what to do later in the module.

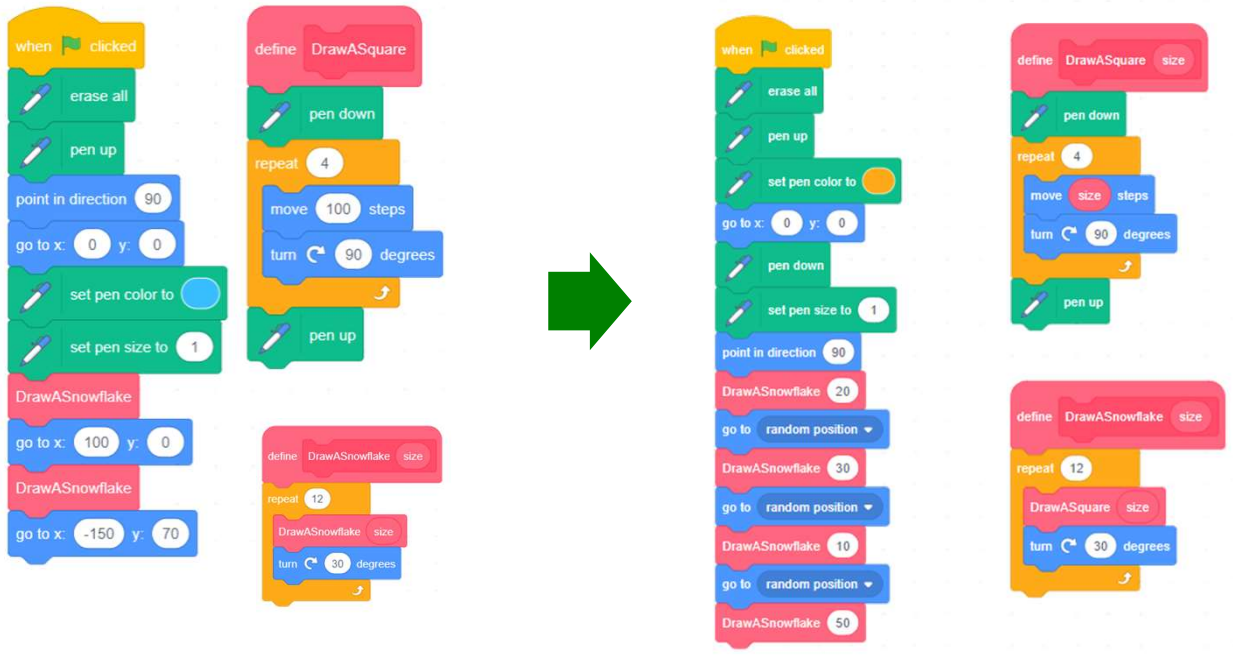
In computer programming, modularization is the process of organizing code for the task it executes and always try to keep the code reusable. For example, we can use the “DrawASnowflake” module repeatedly to draw snowflakes.

Also, we add a variable, "size" which we call a "parameter", to the custom blocks. This allows us to slightly change the modules. With the parameter “size”, we can create different sized squares and snowflakes.

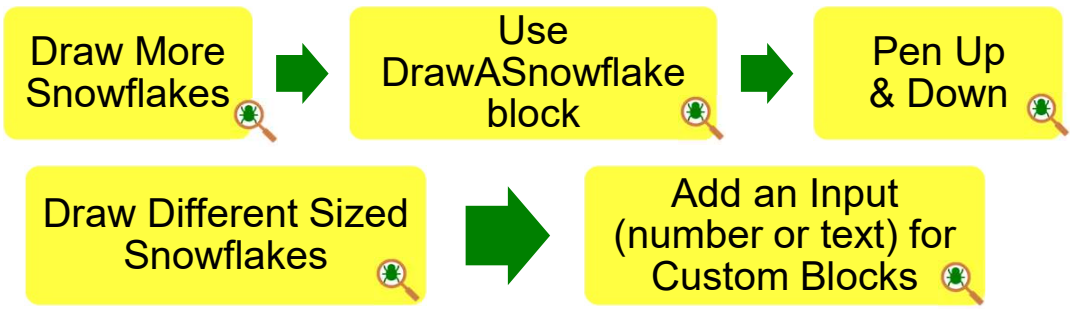
Designing Patterns in Scratch

Revision on Key Concepts & Practices

Being incremental and iterative: to work out a sub-task as an iteration, try it out, then work out another sub-task in one more iteration until the whole programming task is completed.



Testing and Debugging: Testing a computer program is the process of checking if it can produce results as designed. Debugging a computer program is the process of finding out ways to revise the program so that the bugs can be removed.



Appendix

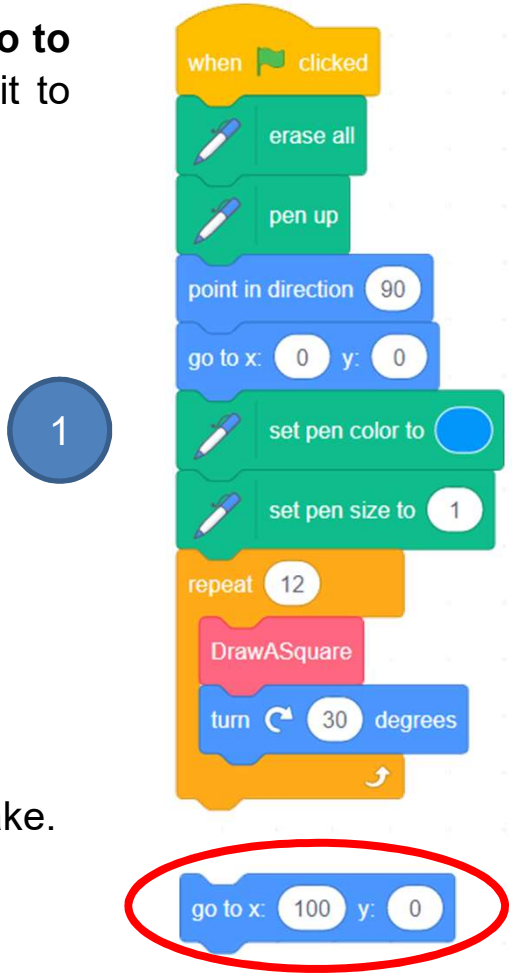
Operation Manual

Designing Patterns in Scratch

See Student Guide P.6

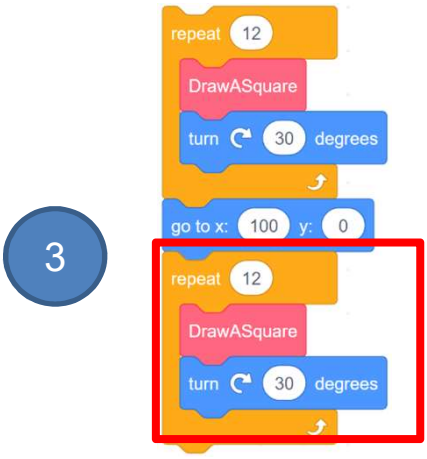
To Code: Draw More Snowflakes

1. To move the sprite to a new position, drag a **go to x:0 y:0** block from the **Motion** drawer. Snap it to the above blocks and change **x to 100.**



2. Duplicate the **repeat** block that draws a snowflake.

3. Snap the duplicated **repeat** block to the **go to x:100 y:0** block.

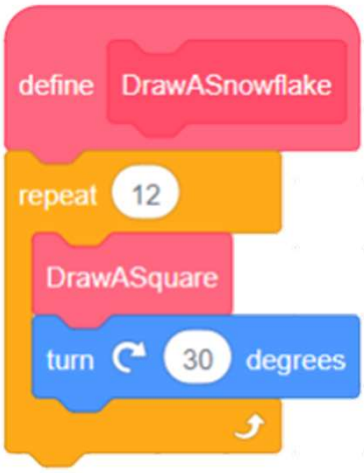


Designing Patterns in Scratch

To Think and To Code: Use DrawASnowflake block

See Student Guide P.7

Revision: Drag the blocks that draw a snowflake and snap them to the **define DrawASnowflake** block.

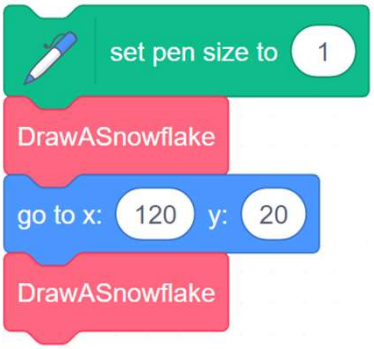


To draw multiple snowflakes in different locations on the stage, you need to move the sprite to different positions.

1. Drag a **go to x: y:** block from the **Motion** drawer. Snap it to the existing blocks.

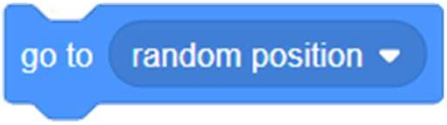
2. Change the x and y values of the **go to x: y:** block to different values of your choosing to change the position of the snowflake.

3. Drag a **DrawASnowflake** block and snap it to the **go to x: y:** block.



If you want to add more snowflakes, just repeat step 1 to 3.

You may also try **go to random position** instead of to certain position.



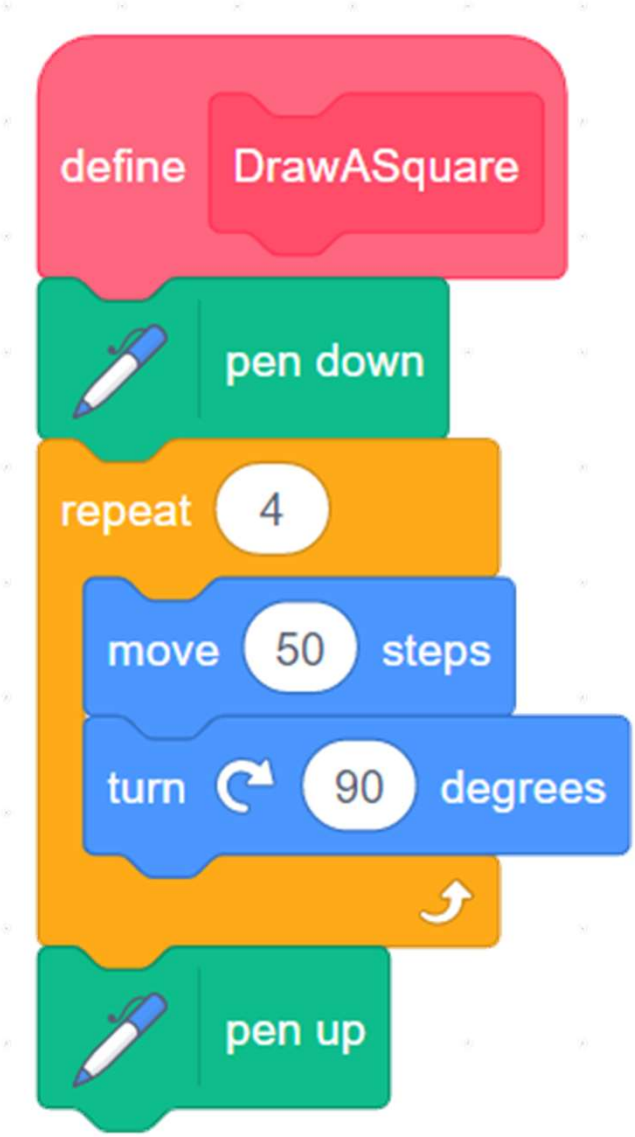
Designing Patterns in Scratch

To Think and To Code: Pen Up & Down

See Student Guide P.8

When excuting the "DrawASquare" block, to draw a square, keep “pen down”; after drawing, keep “pen up” .

If you think the pattern is too big/small , you may also update the move (e.g. 50) steps.



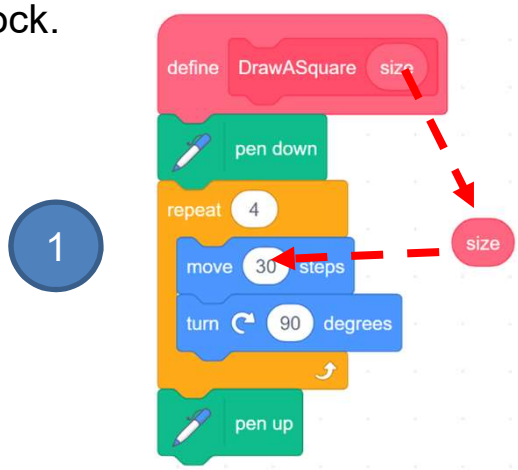
Designing Patterns in Scratch

To Think and To Code: Add an Input (number or text) for Custom Blocks

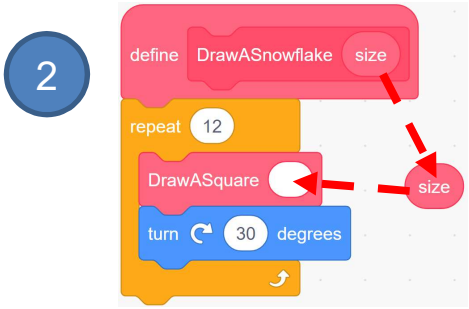
See Student Guide P.12

Set the size for **DrawASquare** and **DrawASnowflake** blocks.

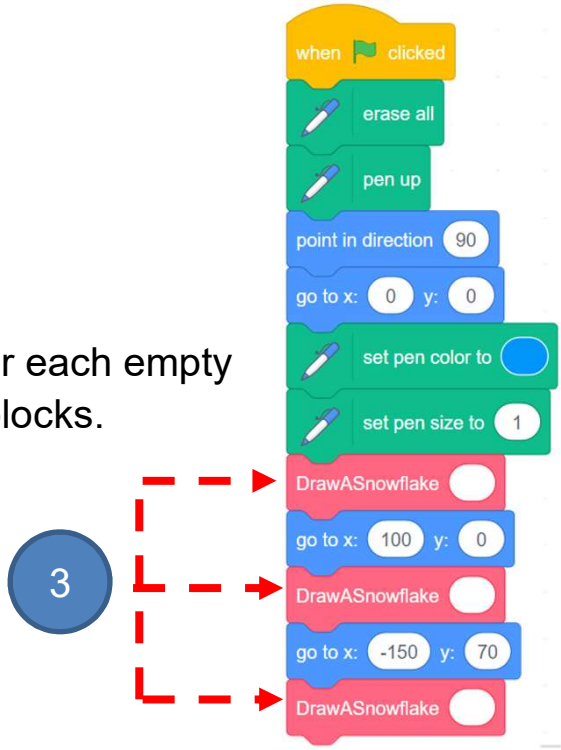
1. Drag **size** from **DrawASquare (size)** to replace **30** in the **move 30 steps** block in the **DrawASquare (size)** custom block.



2. Drag **size** from **DrawASnowflake (size)** into the empty slot ( ) of the **DrawASquare ( )** block in the **DrawASnowflake (size)** custom block.



3. Choose three different size values for each empty slot ( ) of the **DrawASnowflake ( )** blocks.



Scratch Final Project
Student Guide

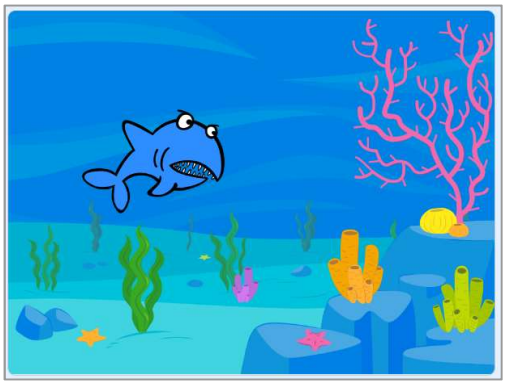
Content

Scratch Final Project

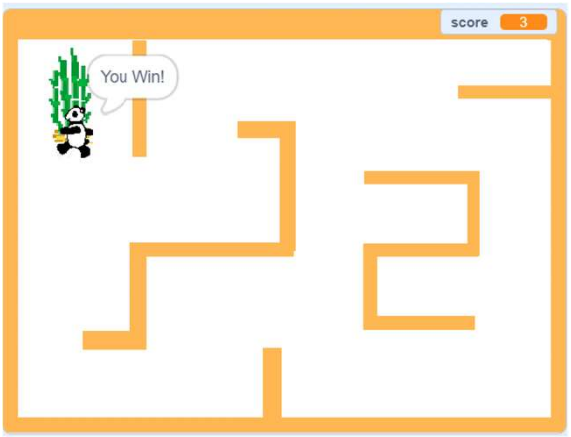
HERE IS THE CHANCE TO SHOW YOUR CODING SKILLS AND DIGITAL CREATIVITY!

- ❑ In this project, you will design and build a Scratch project.
- ❑ Before you start, let's play all Scratch projects again in the previous units and review what features you have learnt so far!

See Appendix P.20-21



Do you still remember the fish exploring under the sea, the panda walking in a maze, and Gobo having a picnic with his friend?



Scratch Final Project Design Worksheet

- ❑ Work in pairs to design and build a Scratch project. Set a theme of the project by identifying a problem or an issue in your daily life and solve it by developing a Scratch program! (e.g. helping to improve the living of the people in need, recycling for protecting our environment etc.)

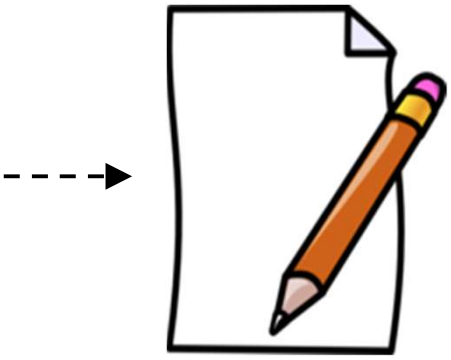
- ❑ It can be a game or a story. Think about what kind of project you would like to make. Remember that you will have 8 lessons for this project.

To Think…

- ❑ Think about the kind of project you and your groupmates would like to make.

- ❑ Which topic would you like to choose?

- ❑ Would you like to take one of your previous Scratch projects and build something based on that?



Problem Identification

Discuss with your groupmates, think about the Scratch project's theme and design. Use this worksheet to jot down your ideas.

1. Does anyone around you is facing problem? Who is she/he?

2. What problem is he/she facing?

3. How would you like to use Scratch to solve his/her problem? Describe your idea.

Problem Identification

Below are some examples for reference.

What would you do if you face the following situations?

Grandma would like to learn more about technology but she does not have chances to explore.

Nowadays, some students are not getting enough exercise and even facing obesity, can you create a Scratch project to provide some tips on how to keep healthy?

Your classmates feel stressed before examination. Can you tell her a joke in Scratch and make her feel relaxed?

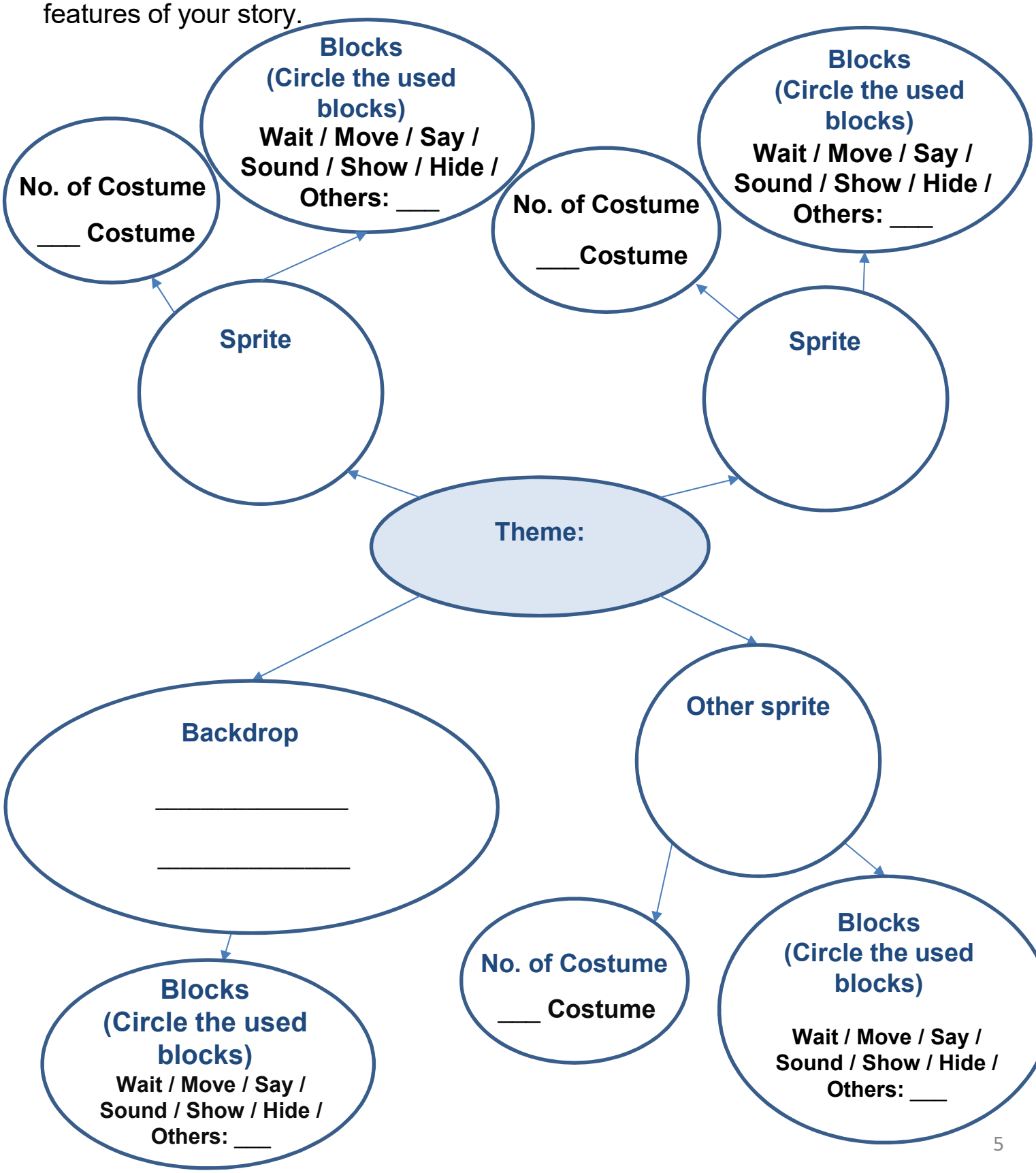
Your little brother is not familiar to the types of recycling. Try to create a recycling game for him?

Your little sister loves playing Mathematics games. How about creating a quiz by using Scratch?

Your friend loves music! Can you think of creating a music box that can play different music?

Design your Scratch Story

How would you like to use Scratch to make a story? Use the mind map to list the features of your story.

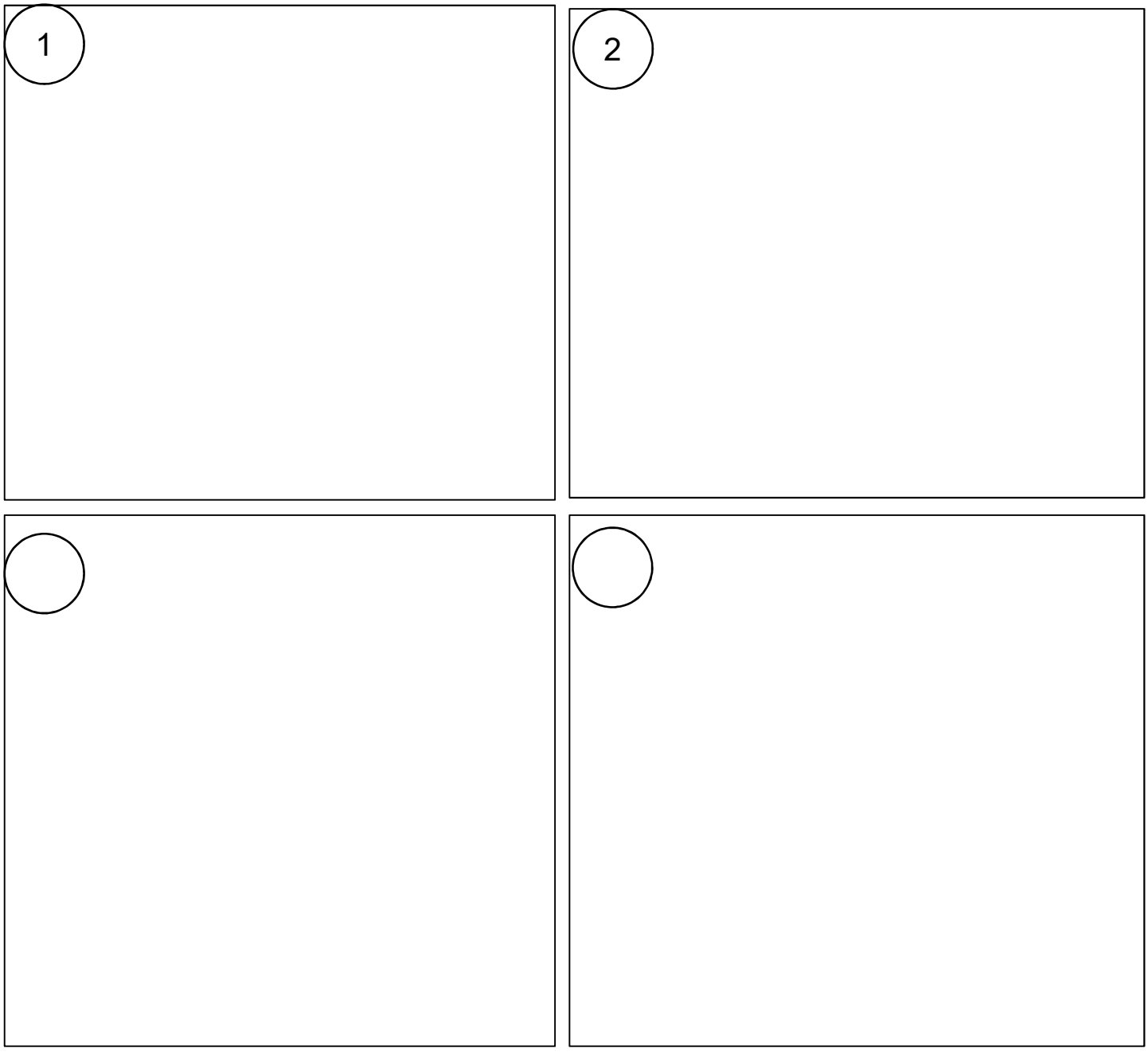


Design your Scratch Story: Storyboard

Try to draw pictures to represent the sequence of your story in the boxes below. Put number 1, 2, 3 and 4 in the circles.

Try to think about the theme and write down your ideas below:

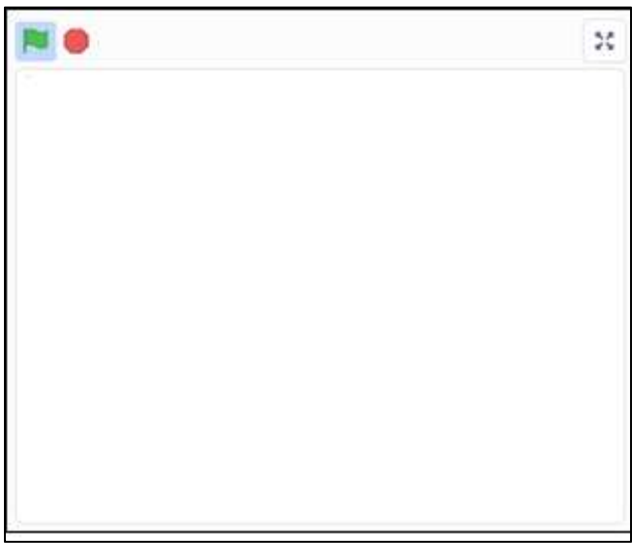
1. Describe the costumes/ motion of your sprites in the story.
2. Introduce the design of using different backdrops



Design your Scratch Game

How would you like to use Scratch to make a game? Use the flow diagram to design your game.

Game Design:



Sprite	
Motion	
Design / Rules of the Game	

Design your Scratch Game

Use the flow diagram to design your game with a starting and ending point. It usually involves conditionals to determine an action.

To-Do Checklist

- ❑ Try to make a To-Do Checklist before starting. It is a good practice to keep track of your progress.

  - ❖ Before each lesson, write down what you want to accomplish.
  - ❖ At the end of each lesson, write down what is completed, or what problems prevents you from completing that task.

Date/Lesson	Tasks to be completed	Status (completed / encountered problems, how you will fix it)

To-Do Checklist

Date/Lesson	Tasks to be completed	Status (completed / encountered problems, how you will fix it)

Pair up with a group

Peer Assessment Worksheet

Pair up with a group. Listen to the other group's design carefully. Try to provide them some constructive feedback for a better design!

For example,

*The **backdrop** **is** beautiful.* *The **sprites** are vivid and fun.*

*You can change the cat **sprite**'s costumes.* *Please add **sound** effect.*

Group ( )

Please write down two things that you like about this project.

_____

_____

Other Suggestions

_____

Group ( )

Please write down two things that you like about this project.

_____

_____

Other Suggestions

_____

Group Presentation

- ❑ Following your teacher's instructions, you may first enhance your project by using the feedback from "Peer Assessment Worksheet". Then present and share your project by using the Scratch Studio.

- ❑ When you present your project, you should introduce:

1. Yourself and your groupmates:

_____

2. The theme / goal of the project (the problem you identified and solved):

_____

_____

3. What you are most proud of your project:

_____

4. Difficulties you overcome.

_____

Group Presentation

Group Presentation Peer Assessment Worksheet

Pair up with a group. Listen to the other group's design carefully. Try to provide them some constructive feedback for a better design!

For example,

*The **backdrop** is beautiful.* *The **sprites** are vivid and fun.*

*You can change the cat **sprite**'s costumes.* *Please add **sound** effect.*

Group ( )

Please write down two things that you like about this project.

_____

_____

Other Suggestions

_____

Group ( )

Please write down two things that you like about this project.

_____

_____

Other Suggestions

_____

Reflection

Self-assessment

❑ Please tick ☑ and fill in the blanks as appropriate.

1. Did you meet your own goals for your project?

   ☐ Yes ☐ I can do better

2. If you could redo your project, how would you enhance the project? Would you change the design or codes of the project?

   ☐ I want to change the design, because _____

   ☐ I want to change the codes, because _____

   _____

3. What was your role as a partner in this project?

   ☐ Group Leader ☐ Programmer ☐ Graphic Designer ☐ Others ___

   Think about what you did?

   _____

4. Are you a good partner? Did you respect and inspire your partner in the project?

   ☐ Yes ☐ I can do better

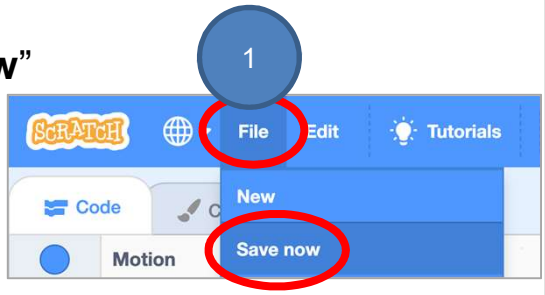
   Because I… _____

Group Presentation

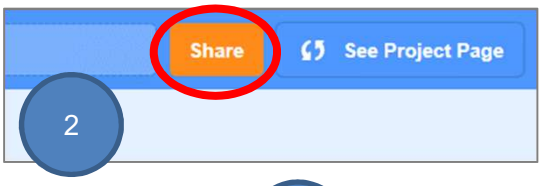
Sharing to Studio for Group Presentation

When you finish, you will add your project to your teacher's Studio.

1. Save your project by clicking "**Save now**" under the File menu.

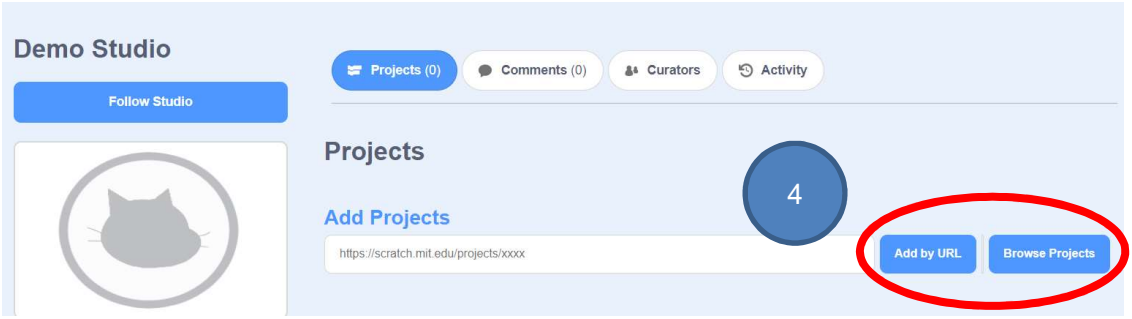


2. Click the orange "**Share**" button.

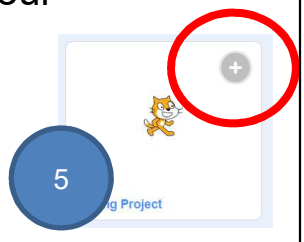


3. Go to your teacher's Studio (they will give you a URL).

4. In the "**Add projects**" column, you can **Add by URL** or **Browse Projects**.



5. If you choose **Browse Project**, then you will see all your shared projects.
   Find the right one and click the "+" to add it to studio.

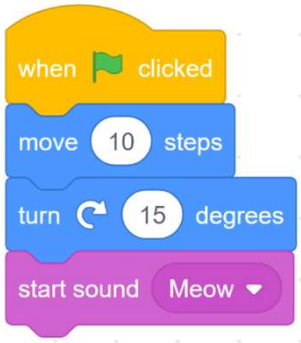


Review Basic Programming Constructs

As you plan your project, think about the basic programming constructs, the key concepts and practices that you have learnt, as well as the blocks you have used before. Try to use various Scratch blocks you've learned so far.

SEQUENCES

Multiple instructions put together form a sequence that happens in order.

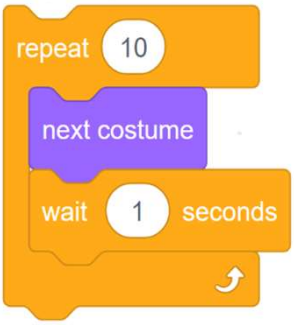


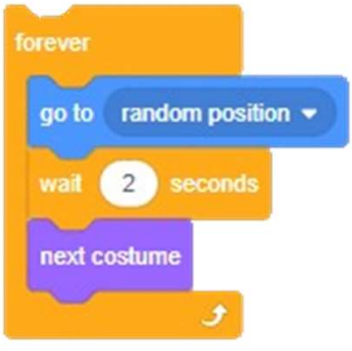
BRANCHING / SELECTION

Sometimes you need to make decisions. Trigger sprite action based on if-then blocks.

ITERATION / REPETITION

Repeat and forever blocks let your sprite easily do things over and over.

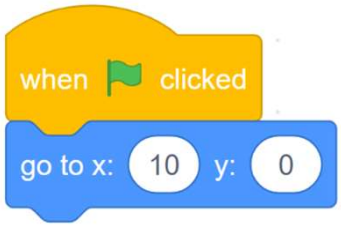




Review Key Concepts and Practices

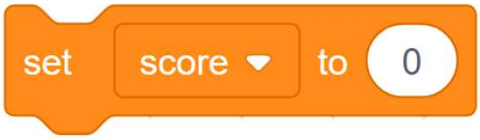
EVENTS

Control sprites based on an event like a key press.



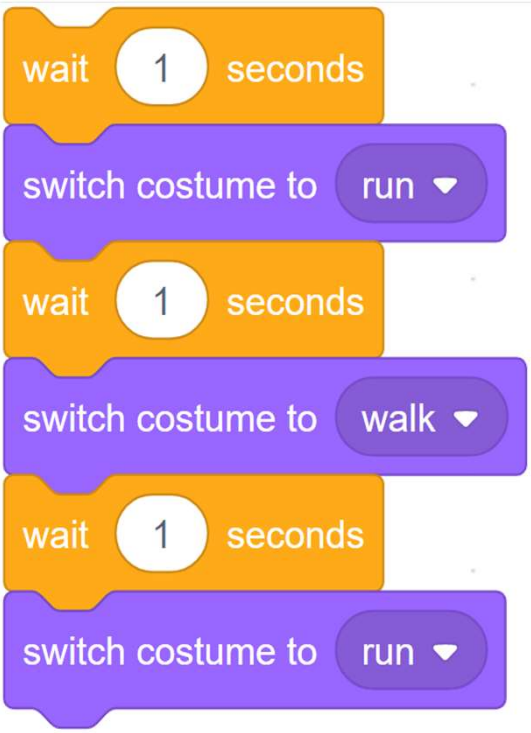
VARIABLES

Variables can help you save information and update it in your Scratch project.



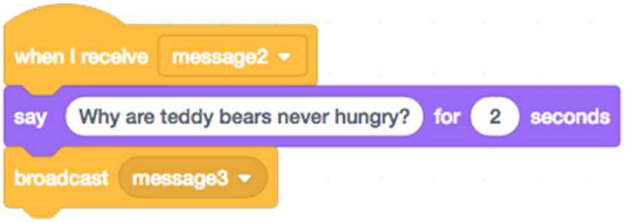
COSTUME CHANGES

Show movement by changing a sprite's costume!



BROADCAST

Broadcast and **when I receive** blocks let sprites talk to each other to trigger actions.



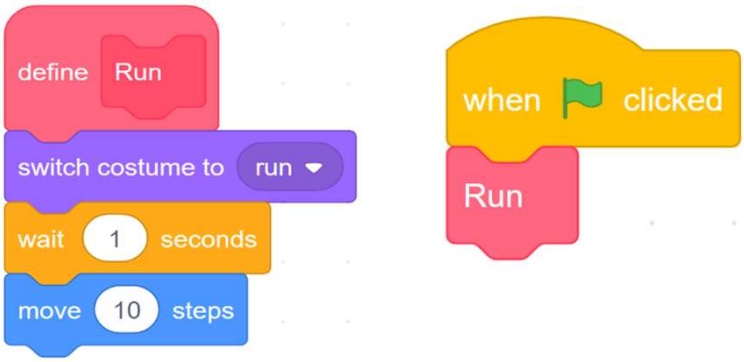
NAMING

Give an good name to the sprites is also important.

Review Key Concepts and Practices

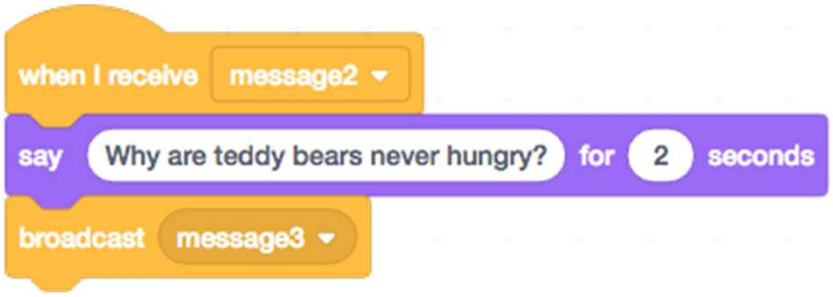
MY BLOCKS

Use custom blocks (My Blocks) to break complex tasks into smaller blocks of code that can be reused.



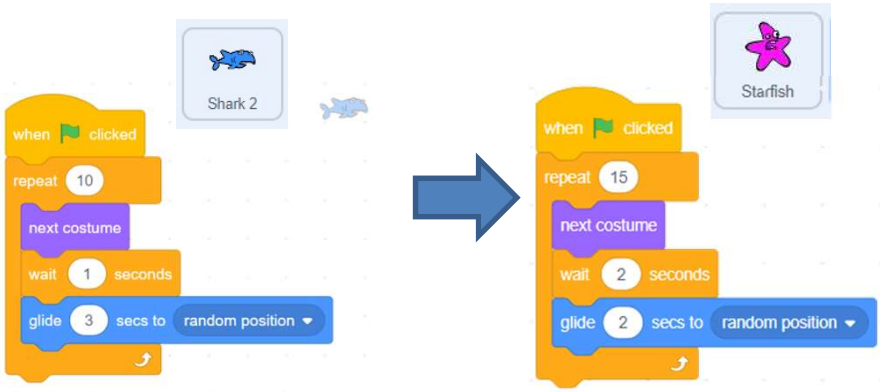
BROADCAST

Broadcast and **when I receive** blocks let sprites talk to each other to trigger actions.



REUSE AND REMIX PROGRAMS/ CODES

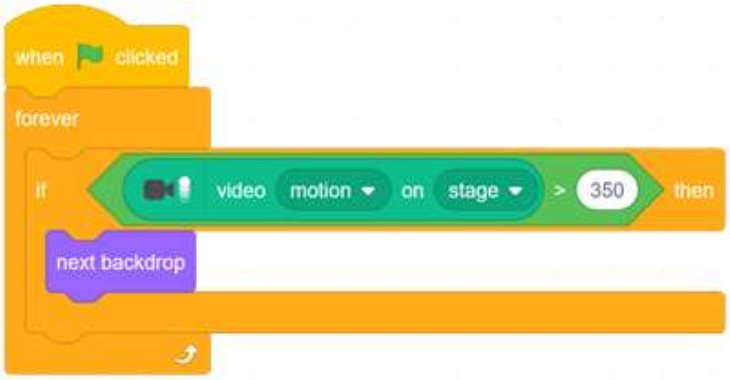
We can reuse and remix the codes of one sprite and use them for the second and third sprites.



Review Key Concepts and Practices

CONDITIONAL OPERATORS

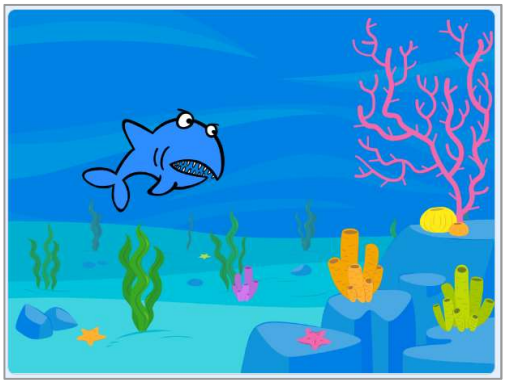
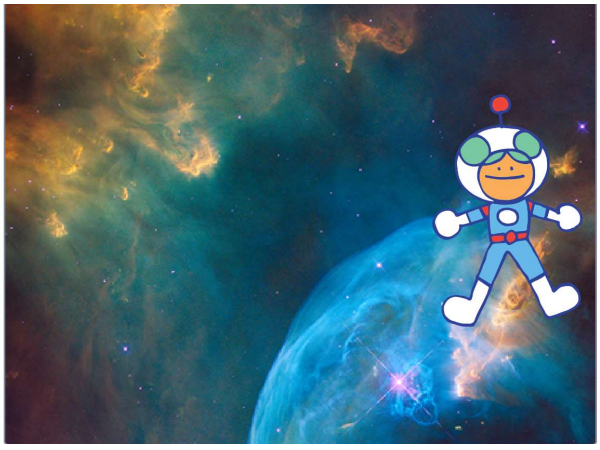
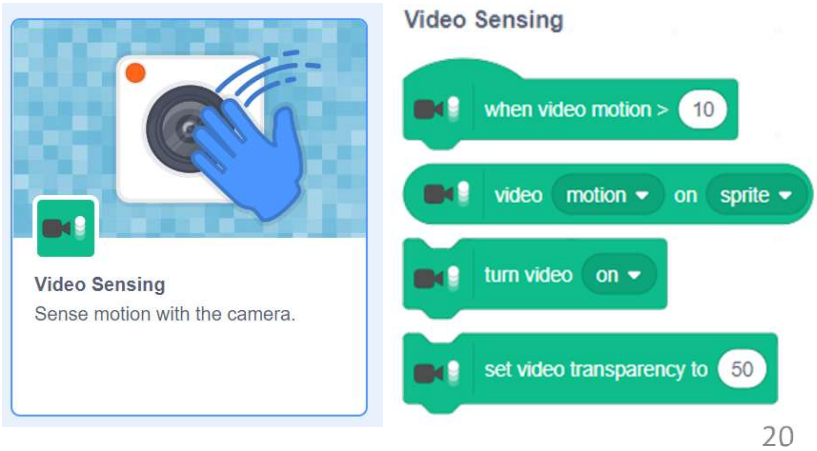
We use **operators** to evaluate whether a condition is true or false. Conditional expressions always use operators such as greater than (>), less than (<) or equal (=).



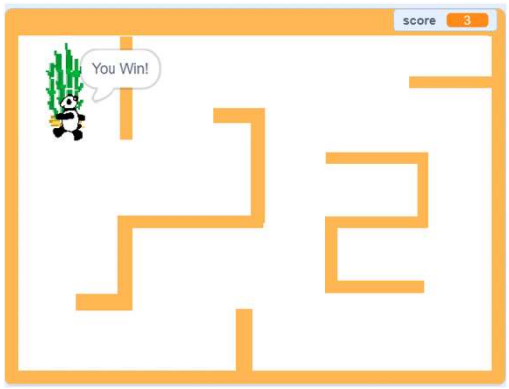
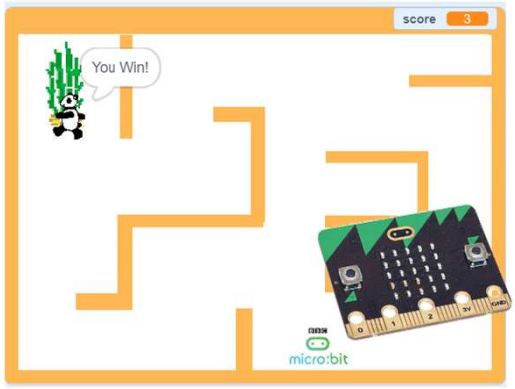
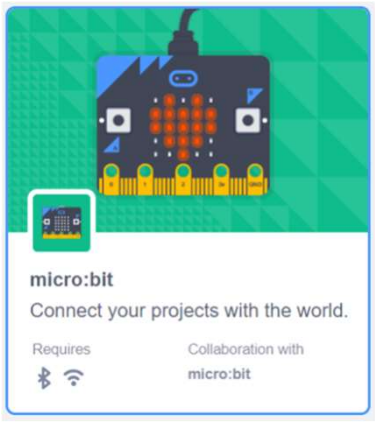
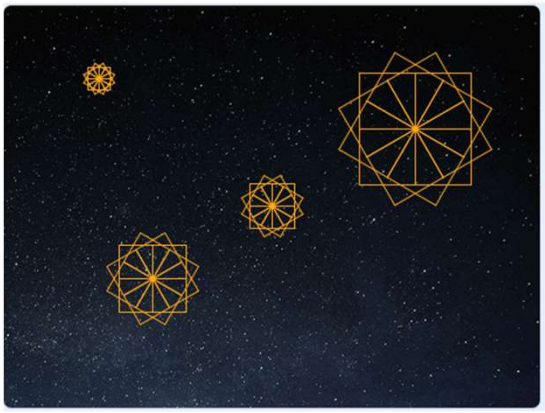
Appendix:

Review Scratch Features

- Let's review what features you have learnt so far in the previous Scratch units! You can play it again anytime!

Unit	Key Features Learnt
Unit 2  https://scratch.mit.edu/projects/722781437	
Unit 3  https://scratch.mit.edu/projects/731740461/	
Unit 4  https://scratch.mit.edu/projects/727401089	

Review Scratch Features

Unit	Key Features Learnt
Unit 5  https://scratch.mit.edu/projects/722154863	
Unit 6  https://scratch.mit.edu/projects/734787236/	
Unit 7 and 8  https://scratch.mit.edu/projects/737985288