CODING WITH MINECRAFT 5: CONDITIONALS

Student workbook

education.minecraft.net

CONTENTS

OVERVIEW

Unit summary

In this unit, we'll explore the concept of conditionals. An important part of programming is telling the computer when to perform a certain task. Conditionals accomplish this by requiring a certain condition or rule to be met before an action is performed. You'll create a game in Minecraft that compares your age with other classmates and code your agent to chop wood and mine for resources to automate common Minecraft tasks. You'll also be building things like pyramids and make your code work a bit more intelligently so it responds to what the user inputs. Then you'll learn about collaborative design and pair programming to challenge yourself and a partner to create an independent project automating some other Minecraft task of your choice.

Learning goals

By the end of this unit, you'll be able to:

- Describe the importance of conditionals in coding.
- Create IF THEN and IF THEN ELSE conditional statements.
- Code with a variety of conditional blocks to automate their agent to find and collect important resources.
- Evaluate code to identify problems like infinite loops and debug the code with conditionals.
- Add a Say block inside if then blocks to help debug problem code.
- Work collaboratively to design an original creative project to apply their coding skills in new ways.

Coding concepts

Conditionals: The part of a program that tells a computer **when** to perform a certain action.

Types of conditional statements

IF THEN	IF a condition is met, THEN an action is performed. For example: IF you clean your room, THEN you can go out with your friends.
IF THEN ELSE	IF a condition is met, THEN an action is performed, ELSE a different action is performed. For example: IF you finish your chores this week, THEN you get your allowance, ELSE you're grounded.

LESSON A: INTRODUCTION TO CONDITIONALS

Overview: Conditionals

Computer programs are made up of instructions that tell the computer how to process input and deliver output. In Minecraft, the instructions you write are triggered by events or by commands, and they control what actions take place in the Minecraft world. An important part of programming is telling the computer WHEN to perform a certain task. For this, we use something called 'conditionals', because a certain condition or rule has to be met before an action is performed.

You're already familiar with the concept of conditionals in your daily lives. Have you ever heard your parent say any of the following?

- "If you clean your room, you can go out with your friends."
- "If your homework is done, you can play video games."
- "If you do your chores all week, you get your allowance, or else you are grounded."

These are all conditionals! Conditionals follow the format of **IF** this, **THEN** that. (Sometimes the THEN is implied.)

> **IF** (condition is met), **THEN** (action performed)

Another format for conditionals is **IF** this, **THEN** that, **ELSE** something else.

> **IF** (condition is met), **THEN** (action performed), **ELSE** (different action performed)

Types of conditions in MakeCode

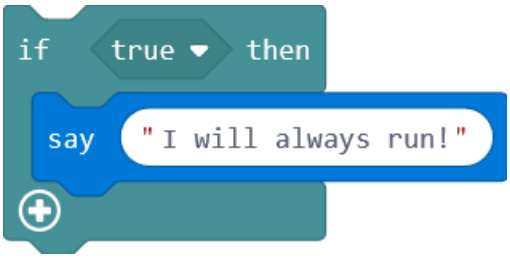
MakeCode for Minecraft features a conditional block called an IF… THEN… block.

There is also another related block called the IF… THEN… ELSE… block.

These blocks are in the LOGIC Toolbox drawer. These blocks will check to see if a certain condition is true, and if so, then they will perform their operations.

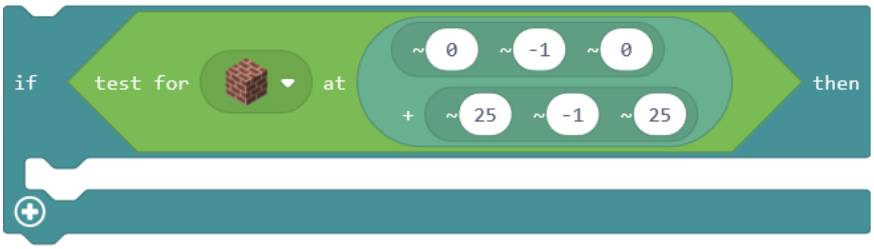
If you select the little plus sign at the bottom of these blocks, you can add more conditions.

By default, IF statements have a hexagon that says “true” in them when you first place them. If you don’t change this, then anything that the “If… Then…” block encloses will run every time because “True” is always true, like this example:



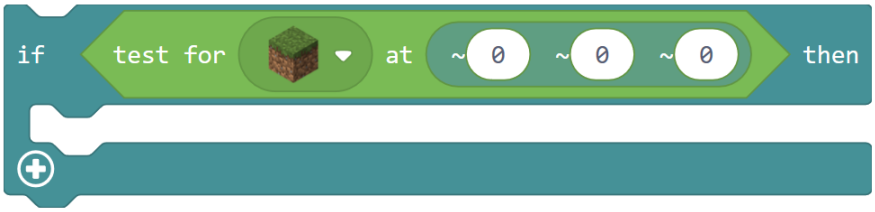
However, you can substitute another block, or a combination of blocks, next to the If. The blocks you use as conditions do not have to be hexagons but ultimately should resolve to true or false.

This example from the BLOCKS Drawer tests for a certain type of block at a specific set of coordinates:



In this example, they represent the player’s exact location. This is always going to be false because the player is standing there, so there cannot be a block where its head or body is.

However, you can specify `~0 -1 ~0` to check the block the player is currently walking on, or even use a Position range to detect the presence of a certain block within a given area.



Today’s activities

- Play an unplugged activity to get familiar with conditionals: Simon says IF THEN ELSE
- Code a game with conditionals: How old are you?

Coding activity: How old are you?

In this activity, you'll practice using the conditional if then else block in MakeCode. You'll code a program that uses a condition to compare your friend's age with your own, and print different messages if they are younger, older, or the same age.

Import the tutorial to MakeCode from here:
https://minecraft.makecode.com/?ipc=1#tutorial:github:mojang/educationcontent/CodingWithMinecraft/unit_5/lesson_A

Try it out!

Time to run the program:

- In Minecraft, enter T to open the chat window.
- Ask your classmate their age, and enter age x – where x is your neighbor's age.
- Look in the sky above you at your message!

Challenges

Let's change some things to make our own different and unique situations!

Challenge 1 - Compare Ages and Modify Messages

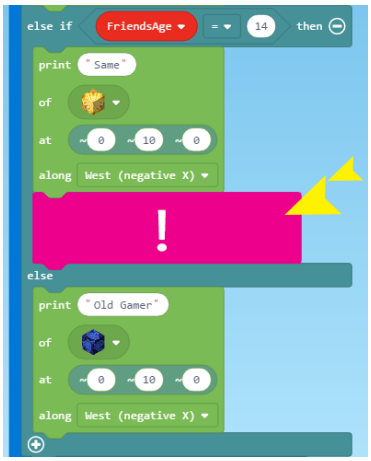
Rather than compare students at 12 years old, compare students to another age. If you're brave, ask your teacher how old they are and put their age in for comparison. Modify the messages for people who are younger than your new age, the same age, or older than your new age. Finally, modify the blocks that print.

Challenge 2 - Compare Birthdays

Place in another if then else to check whether your friend has the same birthday as you.

To do this you will need to take in another variable from on chat command "age". Using a number variable might be easiest, but you could do this in other ways too.

In the following picture, you can see where you would need to test for the proper month, so put your new if then else there.



Hint: Because you are printing two messages, one for saying you are the same age and one for responding to the month, the second printing will not start until the first is finished.

If you print from two blocks of code like this, remember that moving around will most likely affect the position of the second block.

Extensions

Here, there are no rules... Copy the code and change things around to see what kind of results you can create. Suggestions are given, but do as you like!

Extension 1 - Try Other Types of Conversions

Use this extension to change your weight from pounds (lbs) to kilograms (kg) and vice versa. If you enter **weight** in the chat window, the code shows you a menu that explains the program.

If statements allow you to detect what the user wants to convert to. Also, a If statement checks whether it is ok to display an answer message.

How could you change this code so that the output is not in the chat window? Maybe you could print blocks instead as an answer....

What other things could you ask people?

Extension 2 - Create Blocks and Items

This code has a menu. Enter **create** in a chat window. The code accepts the ID for blocks and items.

If the ID is for a block (1 - 255), the block is placed.

If the ID is for an item (256 - 452), the item is equipped to your character.

Hint: Some items or blocks may not appear because they need certain other conditions to be placed, for example, a wall.

Write your ideas here:

LESSON B: CODING WITH CONDITIONALS

Today's activities

Code your agent to do tasks for you:

- Chop down trees and collect the wood
- Mine for resources

Coding activity: Agent Tree Chopper

Chopping trees for wood is hard work but necessary in Minecraft if you want to craft objects and tools in Survival mode. But you can automate this chore through code with the help of our agent! Let's teach the agent how to chop down a tree of any height and return back down to the ground.

When you start a new MakeCode project, it's often a good idea to plan out what steps you want to take before you start coding.

Here's what you'll want to do:

1. Figure out how tall the tree is:
    - As long as there's a block in front of the agent, keep moving up
    - Keep track of how tall the tree is
2. Repeat the following the same number of times as the tree is tall:
    - Destroy the block in front of you
    - Move down
3. Collect everything

A Different Conditional

The last activity used `if then else` statements to check conditions and perform actions the way you wanted them done. In this activity, you'll use conditionals in a `while` loop. The agent will continue moving up as long as there is a block of tree to climb.

Import the project to MakeCode from here:
https://minecraft.makecode.com/?ipc=1#tutorial:github:mojang/educationcontent/CodingWithMinecraft/unit_5/lesson_B/tree_chopper

Challenges

It would be nice to have the agent teleport as part of `on chat command "chop"`. It is hard to predict which direction the agent will be facing after you teleport it. The directional commands ("tp" and "lt") work, but one command would be more efficient.

Can we automate this? After completing these two challenges, you should be able to accomplish this. First, think about the problem.

You want to:

1. Teleport the agent.
1. Turn the agent until it finds the base of the tree.
2. Begin climbing and run the code we already made.

The only potential drawback to this plan is that you would still need to stand directly at the base of the tree in order for the agent to start in the right place. Low hanging branches might need to be cleared. Perhaps as a bonus you might figure out how to fix this issue as well, but first complete Challenges 1 and 2.

Challenge 1 - Teleport Agent and Get Some Things Set Up for Challenge 2

1. Can you have the agent teleport as part of on chat command "chop"?

After that you will need some things to accomplish Challenge 2. Please create the following:

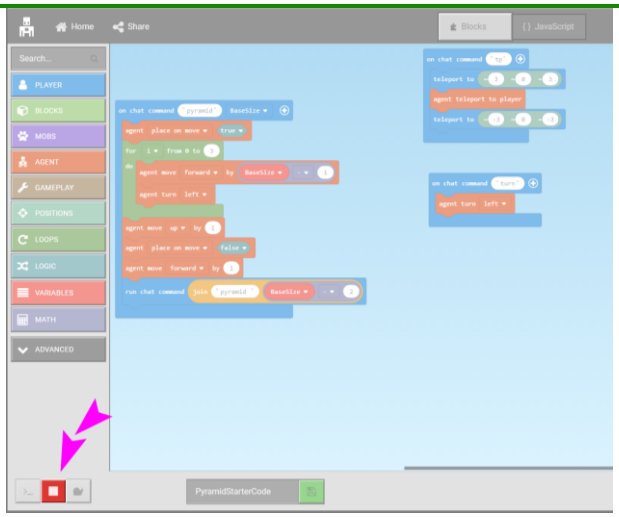
1. Create a variable called `TurnToTree` and set this to `true`.
2. Create a while loop that makes the agent turn left or right.

Test your code!

What happened? Oh no, you're stuck in another infinite loop. `TurnToTree` is `true`, so your new while loop just keeps going and going. Therefore, your agent just keeps spinning and spinning.

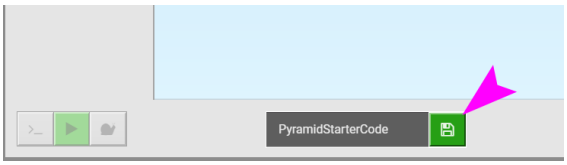
Read below how to stop this...

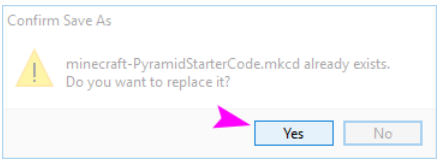
> Tip: If you are caught in an infinite loop, you will need to stop your code from the code connection window.
>
> 1. Click the stop button in the lower-left corner of your code connection window.



2. Restart the coding environment. Click the play button. If the play button is not on, then your code will not run in Minecraft.

3. Saving your code again can restart your code as well, so do that for good measure. Save one more time and say **yes** to the warning message.





This is ok because now you are prepared for Challenge 2, but you need to figure out a way to stop the agent when he sees the tree.

Challenge 2 - Have the Agent Face the Tree Automatically

You are now ready to solve the second task: Turn the agent until it finds the base of the tree.

To do this, you will need to set `TurnToTree` to `false` inside the while loop, but you need a condition for this. You want the agent to stop when it finds the tree.

> There is no find block for the agent, but there is a detect block. Maybe this could be useful?

This block returns true or false if the agent detects or finds a block in the direction you tell it.

How could you test this condition and then stop the agent from turning? You need to set `TurnToTree` to `false`, but where and how?

Write your ideas here:

Extensions

Here, there are no rules... Copy the code and change things around to see what kind of results you can create. Suggestions are given, but do as you like!

Extension 1 - Increase the Size of the Hole

The agent works hard for you! You can get it to dig for you too. Enter the on chat command **dig 4** to try out this code.

Can you get the agent to dig and place something at the same time? Can you increase the size of the hole? Right now, it is 2 x 2. You might get the agent to dig a 4 x 4 hole and line it with gold as it goes down.

Extension 2 - Create a Maze

This extension code is aimed at getting you to think about how you might build some code to create a tunneling robot. The agent will accept two numbers this time. First, it digs down and then will dig horizontally.

Enter the on chat command **dig 5 4** to try out this code.

How could you get your agent to build an underground maze? What else might you do with this underground hole it is digging? You could fill it with lava or put some mobs down there or even build an underground home. What other pieces of code might help you do this?

Write your ideas here:

Coding activity: All Mine!

In addition to chopping trees for wood, you'll probably want to mine the earth for precious minerals. One good strategy for finding the rarest of minerals, diamond, is to tunnel deep beneath the earth to levels Y12 or Y13, and then create a series of branching tunnels off a central passageway. If you completed the Unit 5: Iteration – Independent Staircase Project, you may have some code that gets the agent to dig it for you. If not, get yourself down to Y12 or Y13, and we'll meet you there!

Import the tutorial to MakeCode from here:
https://minecraft.makecode.com/?ipc=1#tutorial:github:mojang/educationcontent/CodingWithMinecraft/unit_5/lesson_B/all_mine

Getting down there!

Find your world position in the game (press the `F1` key to see your world position on the screen), and then enter the command: `/tp X 12 Z`, where X and Z are your current world coordinates. This will teleport you directly down to level 12 altitude in your Minecraft world. Make sure you equip yourself with a pickaxe and torches to make some room and to see down there!

In this activity, you'll create a basic mining agent that automatically destroys blocks in front of it and collects everything. You'll use conditional statements to test for precious minerals. The agent is one block tall, so you'll send it forward 64 blocks, and then have it turn around, move up one block, and come back. If the agent finds anything of value, it will let you know and collect the ore.

Challenges

Let's change some things to make our own different and unique situations! Let's set up a on chat command that will check for two conditions. If the user enters **1**, the character will be teleported down to level 12 automatically. If the user enters **2**, the user will be teleported out of the ground.

Challenge 1 - Set Up Conditions

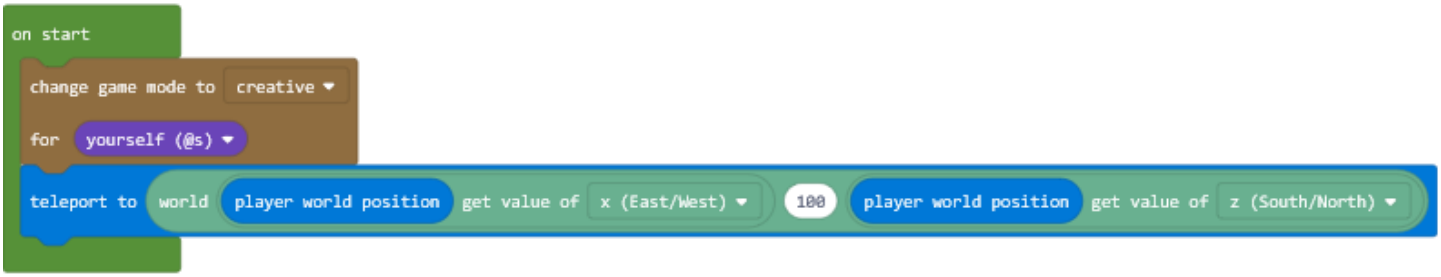
Set up an on chat command "go". Make this so on chat command "go" accepts a variable called **WhereTo**. Put an if then else inside of go.

Test for these conditions:

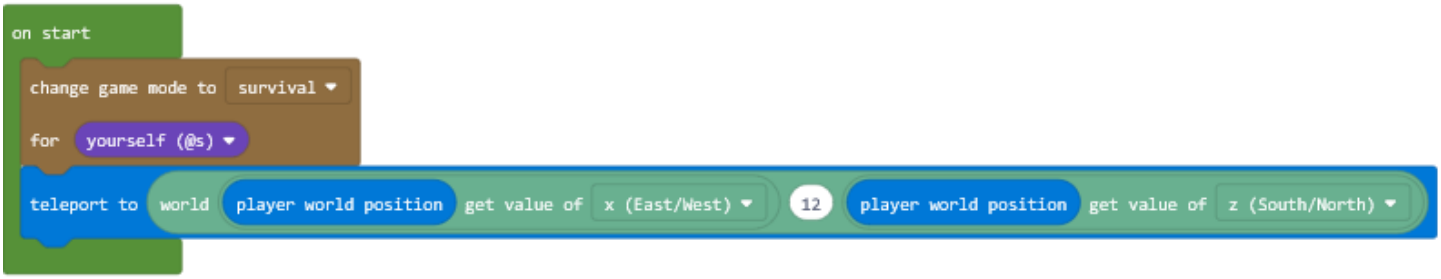
1. User enters "1" (down)
2. User enters "2" (up)
3. User enters nothing

You will need to compare what the user enters and then add the following to the if then else.

Put the following inside the up branch or 1:



Put the following inside the down branch or 2:



Give the user a message if they enter nothing.

Challenge 2 - Add More Conditions

What else could you do to make on chat command "go" from Challenge 1 better? Options:

- What about equipping your character with torches automatically before going down?

- You might clear out a small room so that after the character teleports they have some space.
- You could automatically teleport the agent to your character after you go down to level 12.

Choose at least one of the options above or your own modification and implement your option(s) for this challenge.

Write your ideas here:

Extensions

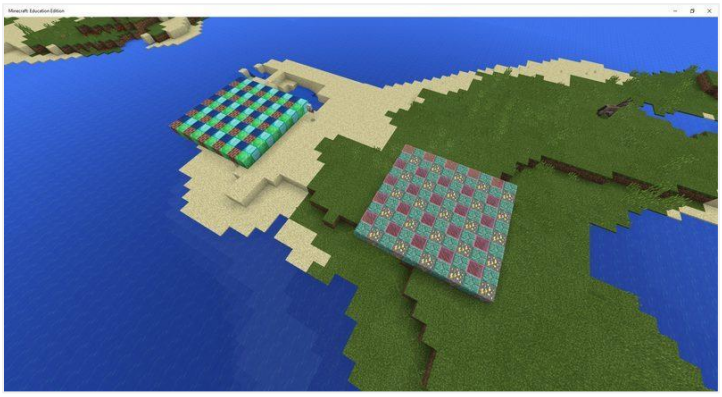
Here, there are no rules... Copy the code and change things around to see what kind of results you can create. Suggestions are given, but do as you like!

Extension 1 - Standard Checkerboard

Have your agent build a checkerboard. Make sure slot 1 and 2 have blocks in the agent's inventory!

This uses a bit of math and a if then else to alternate blocks and give that checkerboard look.

How could you change this? Could you make other patterns by playing with the if then else?



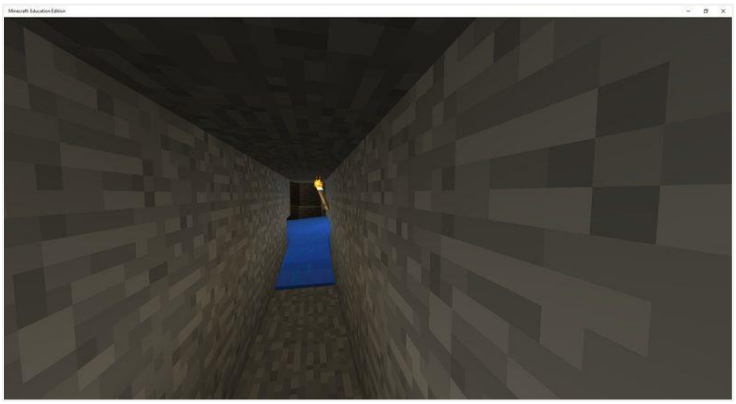
Could you add ideas from other code you have seen to make this different and more interesting? What about one block up and the next down?

Extension 2 - Maze Digger

This is similar to an earlier activity. Here the agent will dig for you BUT it will generate a random maze. There are several concepts we have discussed as well as a new one, functions. Functions will be covered in the next lesson, so you should take a look at them.

The agent turns randomly with the help of a if then else that checks what random number is generated.

How could you make this even more random? A cool maze definitely needs to be random! The natural features of the world will help, but what else could make this maze super fun? Maybe you could add animals or monsters as well!



Write your ideas here:

LESSON C: DEBUG PROBLEM CODE WITH CONDITIONALS

Today's activities

- Debug a loopy program
- Take a quiz

Coding activity: Pyramid

In this activity, you'll examine some starter code that creates pyramids of various sizes. Some problems with the code cause the actual pyramids to build themselves. Can you help sort out some of these "bugs" and improve the code with conditional IF statements?

Import the tutorial to MakeCode from here:
https://minecraft.makecode.com/?ipc=1#tutorial:github:mojang/educationcontent/CodingWithMinecraft/unit_5/lesson_C

Challenges

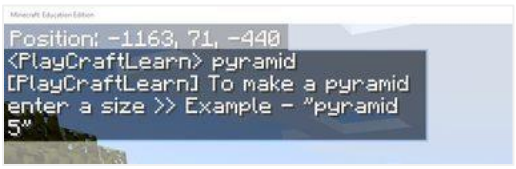
Now you can change some things to make your own different and unique situations!

Challenge 1 - Tell the User When the Pyramid Is Built

Using a `if` statement, tell the user when the pyramid is done. Using the variable `MadePyramid` might help. In your message, you might say something simple like "Pyramid Done!" or you can get more creative with it.

Challenge 2 - Say Something If the User Enters 0 or No Size Is Entered

Try to get the code to inform the user if they entered 0. There is a trick to this that you might have overlooked in Challenge 1. It has to do again with the variable `MakePyramid`. Again, you can be creative with your message here, but usually giving an example of what to enter is helpful. Something like this: `To make a pyramid enter a size >> Example - "pyramid 5"`



Extensions

Here, there are no rules... Copy the code and change things around to see what kind of results you can create. Suggestions are given, but do as you like!

Extension 1 - Wander

The agent randomly walks around leaving a trail of blocks. You can run this script a few times to create abstract art in the world. How might you also make the agent move up and down randomly?

Extension 2 - Build a Colorful Pyramid into the Side of a Hill

This script gets your agent to build pyramids while destroying blocks that are in the way. You will need to equip your agent with at least nine types of blocks. In case you forgot... to equip blocks, right-click or tap and hold your agent and move blocks over to its inventory. Fill up that top row!

Write your ideas here:

LESSON D: GET CREATIVE WITH CONDITIONALS

Today's activities

- Design your coding project with a partner
- Complete and turn in your Minecraft diary entry

Coding activity: Paired independent project

You've looked at how some of the different conditional blocks work:

- A while loop is like a repeat block combined with a conditional statement (repeat while this condition is true).
- A series of If statements separated by Else statements to create blocks of code that run in all sorts of different cases.
- At Boolean operators, which define a relationship between conditions such that both conditions must be true, or only one of the conditions needs to be true, in order for the whole expression to be considered true.

Project expectations

- Work with a partner
- Solve a specific problem in Minecraft with efficient and effective coding and use of conditionals
- Use one or more blocks from the Logic Toolbox drawer

Paired project work techniques

COLLABORATIVE DESIGN

Start by understanding the problem - Interview people around you who might have encountered the problem you are trying to solve. For example:

- If you are designing a survival tool, what do most players do to survive the first night?
- Ask your friends how they find food, build a shelter, and so on.
- What resources do they tend to look for first?
- Do they have a system?
- What would be more effective?
- What do they wish they had the ability to do?

Find ways to find out more information about the problem - Observe people playing Minecraft, with your focus being strictly on studying their behavior in-game. A good way to focus your observations is to make a list of key questions ahead of time:

- What do new players do first?
- How quickly do they decide to go underground?
- How long do they stay there?
- Do they tend to stay in one place or migrate?
- Which characteristics make a good place to build a base?

Then start brainstorming with your partner - Talk about a variety of different ideas. Remember far-out ideas are good. Some of the best programs come out of seemingly crazy ideas!

PAIR PROGRAMMING

It's valuable to collaborate with someone to create programs together. Sharing one computer, one person is at the keyboard acting as the driver and the other person provides directions as the navigator. There are two important rules:

1. When you're are at the keyboard, you're doing what the other person is telling you to do.
2. Practice good communication with each other throughout the entire coding process.

DEBUGGING

Be sure to test out a number of different ideas in Minecraft and keep track of what you learn.

Tips!

- Add a Say block inside an If Then block to test to see if certain conditions are actually being detected. If you do something in Minecraft that should trigger the condition, but you don't see the message printed to the screen, then you know that something is wrong with your condition.
- If you're updating your variables based on certain conditions, use a Say block to print out the value of the variable so you can make sure it is changing properly. If you are using a number variable, remember to use a Join block to join it with a word (or empty quotes) so it will print.

Project examples

IRON FINDER

Problem: One of the most common tasks early in the game is finding enough iron to make iron tools and perhaps a full set of iron armor for protection. Iron is found in streaks of iron in the stone hillsides and caverns, but it is often hidden from view.

Solution: Create a MakeCode Project that checks to see whether iron is near you, and then alerts you when you are very close to iron ore. After you find one block of iron ore, more is likely close by, so all you need is to hit on one block to strike an iron vein!

IMPROVED MINER

Problem: Mining is for resources is hard work and needs to be done as quickly as possible.

Solution: Work with the agent program you defined and see if you can make it better. Can you make the agent check multiple levels at a time? Can you check for other rare minerals as well, like emerald?

Minecraft diary

Compose a diary entry addressing the following:

- What Minecraft problem did you decide to solve? What does your program do?
- How did you use conditional statements in your project?
- Discuss one (or more) ways that working with a partner was different from just doing the project by yourself.
- Describe one point where you got stuck. Then discuss how you figured it out.
- Include at least one screenshot of your agent in action.
- Share your project to the web and include the URL.

Note: If you decided to improve one of this unit's coding activities, please talk about the new code you wrote in addition to what was already provided.

Grading criteria

Assessment	1	2	3	4
Project	Project lacks all three of the required elements.	Project lacks two out of the three required elements.	Project lacks one out of the three required elements.	Project solves a specific problem, efficiently and effectively.
Diary	Minecraft diary entry is missing four or more of the required prompts.	Minecraft diary entry is missing two or three of the required prompts.	Minecraft diary entry is missing one of the required prompts.	Minecraft diary addresses all prompts.

Write your ideas here:

GLOSSARY OF KEY TERMS

Conditionals: The part of a program that tells a computer **when** to perform a certain action. Also known as an IF THEN or IF THEN ELSE statement.