



Drone Logistics Service Learning
Math Editions

Lesson 4: Batteries Now Included

Problem Solving | Distance | Fractions | Percentage | Coding with Functions



"Your skills are growing all the time. The most amazing thing has happened. PowerInc has invented very cool batteries that will let your Tello Drones fly 1000 yards on a single charge.

The loading docks all have the latest charging station installed. These stations will automatically charge the drones if the battery is too low to make the next flight.

Now all we need is your super cool coding skills to make this possible."

Instructions:

Learn about the logic `if-else-do` block on the next two pages and complete the mini-games.

Update your loading dock code to include an if statement to determine if the battery needs to be charged or not.

Have some fun with the Battery Percent Game at the end of the lesson.

If In Real Life:

Computer software is all about the **if statements**. The software usually waits for the user to do something and then the **if statements** kick in. For example:

"If user clicks on little X on the top of the window, close window"

"If user clicks on print icon, print document"

"If user double clicks on internet browser, open internet browser"

These **if statements** are all around us, even in people. Come up with 3 human examples and share them with the class. Here are 3 from me:

1. If hungry, find and eat food
2. If tired, find bed and sleep (this is my cat)
3. If cats meowing by bowl, feed cat

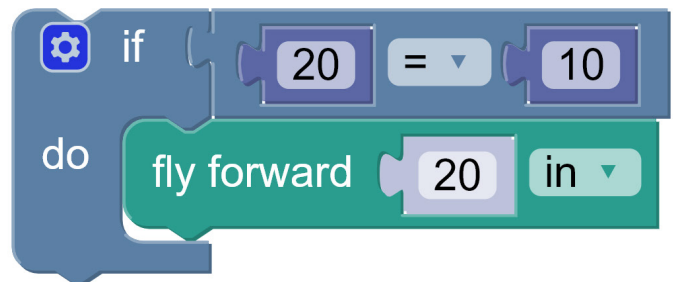
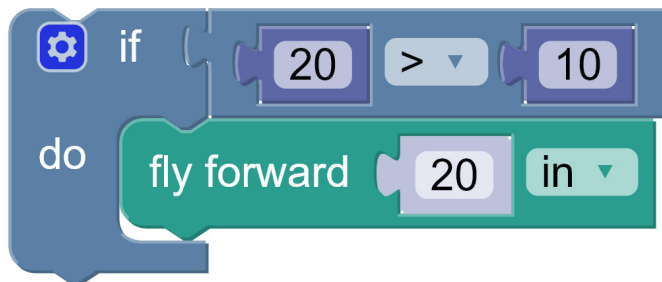
Let's take a look at the **if block** in DroneBlocks



If In DroneBlocks:





DroneBlocks has a really cool logic block. The blocks work with **conditional statements**. We set the condition "if $10 > 5$ " then that condition is **true** and the block runs some code. If the statement is not true, then it is false. "if $10 < 5$ " then that condition is false and nothing happens.

Let's check out two examples:



On the left we can see the condition is if 20 is larger than 10 then fly forward. Is 20 larger than 10? Yes. Yes it is. So the drone will fly forward 20 inches.

Is 20 the same as (or equal to) 10? No. So the drone will do nothing. :) Let's play true or false mini-game with the following statements: (circle the correct answer)

	True / False
	True / False
	True / False
	True / False

Do You Wanna Build An Else Statement?

If **statements** so far have been pretty much “yes or no”. Do this or do nothing. Let’s add a bit more choice into the matter. Sometimes we want to do one thing or another. It’s really cool having more than one action per condition. Look at this for example:

You’re getting a new pet. You don’t know if the pet is a dog, cat, or a turtle.
You’ve only got two names picked out. The dogs name will be *Sparky*. If the pet is not a dog, then it can be named *Taylor*.

This is what the pseudocode would look like:

```
if pet = dog
    then name = Sparky
else
    then name = Taylor
```

Using this above statement, complete the following mini-game by writing down the names of the following pets:



.....



.....



.....



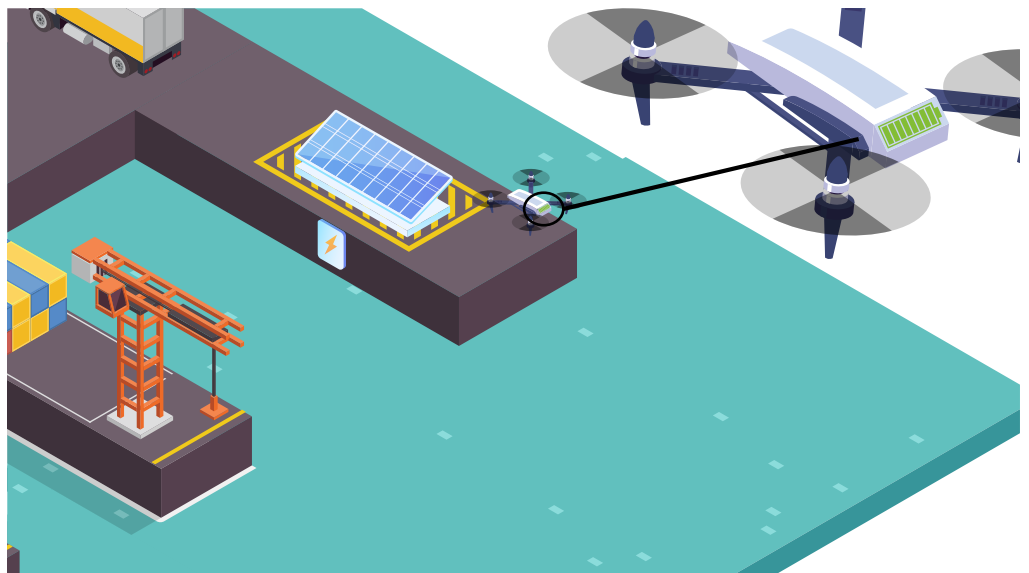
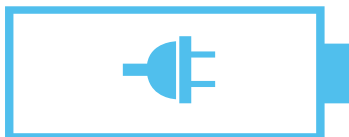
.....



.....

Well done. You only get to keep one pet, sorry. Five pets are a lot of work.

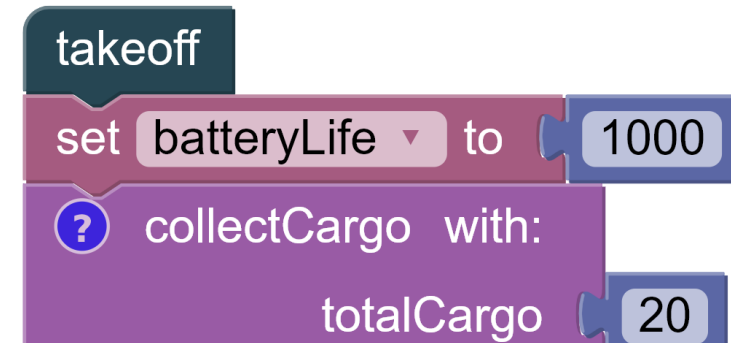
If this else this is a really good way to make decisions in your code. We are going to use this to update our code and introduce battery life. **Exciting times.**



Tool Time:

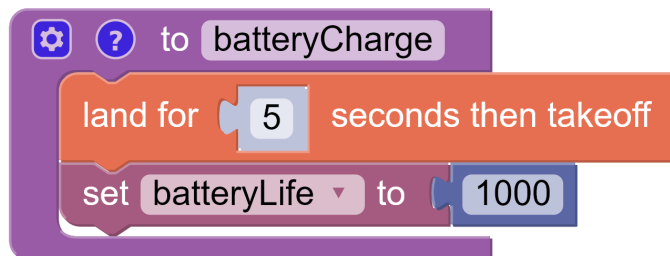
It's time to upgrade our code. Let's start by creating and adding a `batteryLife` variable to the mix. The starting `batteryLife` value is set to 1000. Cool.

Now make a let's quickly make a function called `batteryCharge`. This is just a simple function that lands the drone for 5 seconds and re-sets `batteryLife` to 1000. As our Tello Drone in real-life doesn't have this cool auto-charging feature, we are using the land for 5 seconds as a way of pretending it is charging so we can still write the code. :)



1. Create a new `batteryLife` variable.

2. Set `batteryLife` to 1000. Do this right after takeoff.

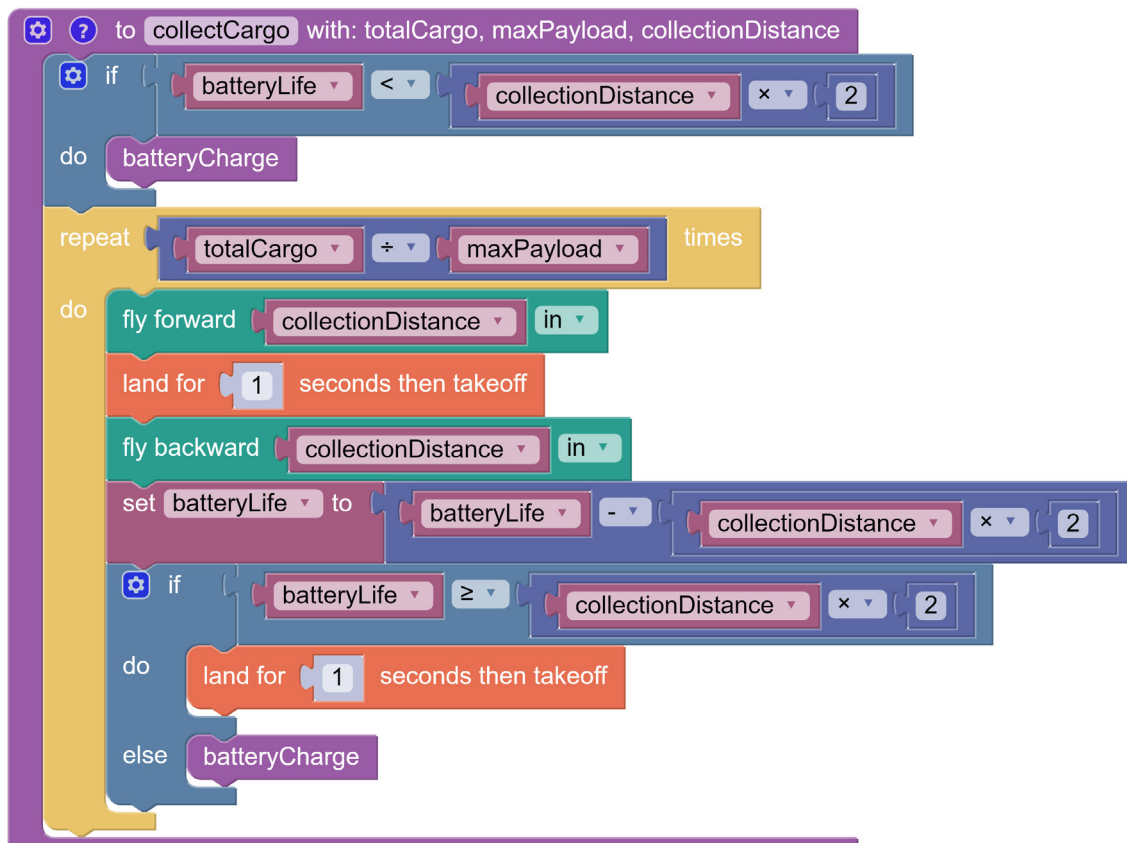


3. Create a `batteryCharge` function

4. Add block `land for 5 seconds...`

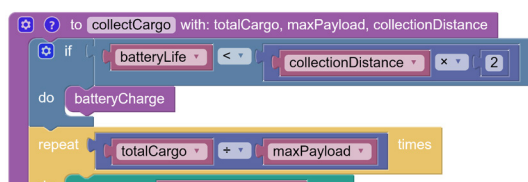
5. Set `batteryLife` to 1000

Ok, so let's think about this for a moment. The `batteryLife` is 1000 and we have a function block called `batteryCharge`. Now we need to add the part of the code the reduces the battery life by the number of yards flown. We will break it down on the next page.

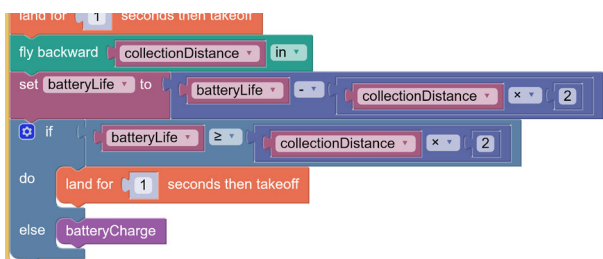


So That's How It's Done:

We are going to look at that giant block of code bit by bit. It will all make sense in a minute.



Before we fly, we check if we have enough batteryLife to make the trip. The battery will fly 1000 yards. So if the collectionDistance x 2 (there and back) is more than the current batteryLife, then we batteryCharge.

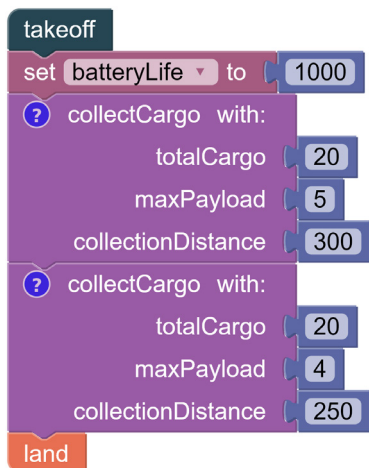


After the drone has flown there and back, we set the batteryLife to batteryLife minus the collectionDistance x 2 (yup, there and back again)

Once the batteryLife has been lowered, we then run an If Else Statement. This will determine if the drone lands to deliver the cargo (land for 1 second) or if it will run the batteryCharge function (land for 5 seconds).

Simmy Simmy Simulation:

Let's run this code in the Simulator and see what happens. Once you have updated your code, run the following parameters in the functions and see what happens.



What you should see is that for the first collection, the drone will be charging after each trip. Why? Well, because the collectionDistance x 2 will be 600 yards. So after each full trip, the drone will be left with 400 yards of charge. That's not enough for a 600 yard flight.

In the second collection we don't need to charge the drone every collection, but only every two collections.

How awesome is that! We can now charge our drones!

Going Going Gone:

On the next page we are going to dive into the exciting subject of fractions and percentages. Before we get started let's look at the 5 different icons we are going to be using



Green: Fully charged and ready to go



Purple: Still got the spark carrying on



Orange: Half way there. Almost time to charge



Red: Where's the charger? Power please



Black: zzzz. Good night

Fundamental Fractions:

Let's take a look at our battery icon. It's a whole battery, but it is made up of 8 equally sized smaller parts. Count them out. How many smaller parts are there? 8. We are going to write that in this format:



$8/8 = 1$ full battery

numerator (current amount)

denominator (total amount)



$6/8$ of a full battery

numerator (current amount)

denominator (total amount)



Half of a full battery

numerator (current amount)

denominator (total amount)



2 eights of a full battery

numerator (current amount)

denominator (total amount)



0 eights of a full battery (0 anything is 0)

numerator (current amount)

denominator (total amount)

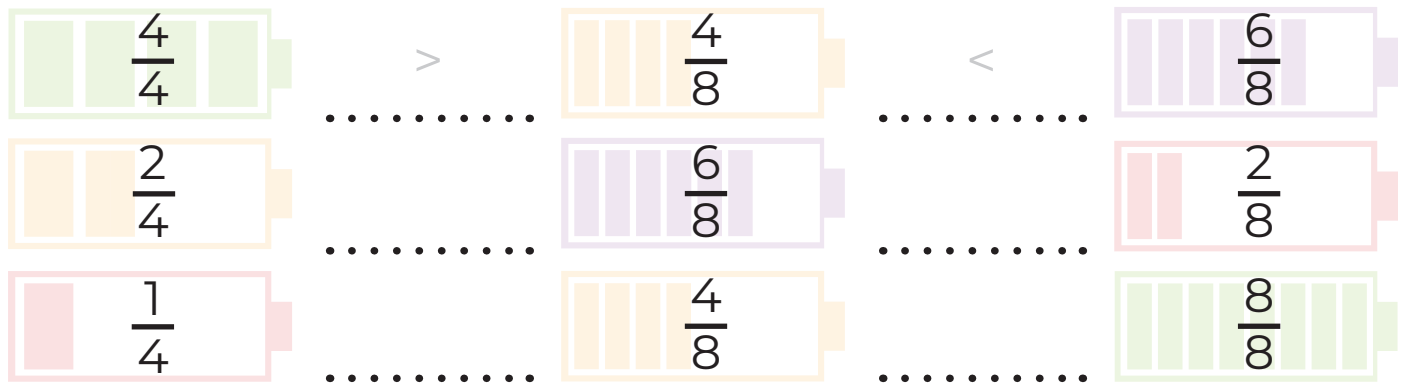
Comparing Fractions:

When you have different denominators, you can still compare equations. Which battery has more charge? Fill in greater than, less than or equal to below:

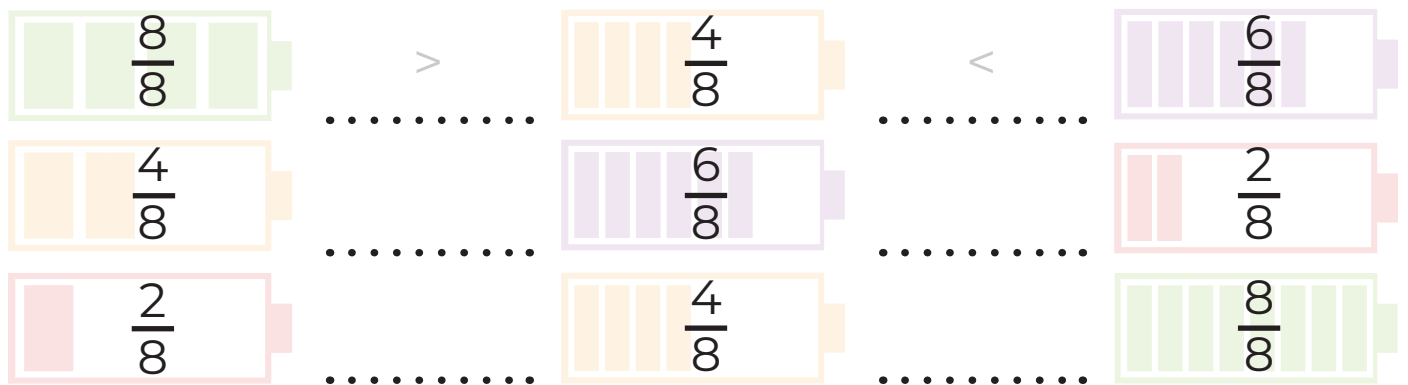


Comparing Fractions Again:

When you have different denominators, you can still compare equations. Let's look at the battery example again but this time we will look at the fractions instead of the batteries.



The secret to working out the answer is to compare apples with apples. Well, denominators with denominator. All we do is find out if we can get $x/4$ to become $x/8$. How does 4 become 8? We multiply it by 2. Just like that. So let's double the numerators and denominators of all the $/4$ and see what happens.



Awesome work! That's a lot of $>$ and $<$ signs you've been using. That equals great work!

Fractions to 100% Percent Done.

When you look at your phones and computers, most of the time the battery has a percent remaining sign. How do we go from $1/4$ or $2/8$ and come up with a %?

Well "percent" means one part in every hundred. We want to turn one forth into parts of 100. Let's look at how this is done:

We need to divide the numerator by the denominator and then multiple by 100.

So 4 divided by 8 would be 0.5. That is because if we have 4 candy bars and eight students, each student will only get half a candy bar.

Then we take $0.5 \times 100 = 50\%$

So if we only have 4 out of 8 that is the same as 50% or half. Awesome. Give it a try:

$\frac{6}{8} \quad \%$	$\frac{2}{8} \quad \%$	$\frac{1}{4} \quad \%$	$\frac{7}{8} \quad \%$
.....
$\frac{3}{4} \quad \%$	$\frac{2}{2} \quad \%$	$\frac{1}{8} \quad \%$	$\frac{2}{4} \quad \%$
.....