Australian **Computing** Academy

DT Mini Challenge

Satellite (Blockly)

The Australian Digital Technologies Challenges is an initiative of, and funded by the Australian Government Department of Education and Training (https://www.education.gov.au/).

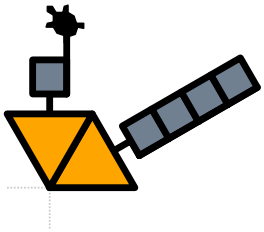
1

INTRODUCING THE TURTLE

1.1. Turtle

1.1.1. Introducing the turtle

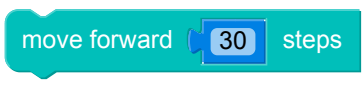
Meet the turtle (https://en.wikipedia.org/wiki/Turtle_graphics)!

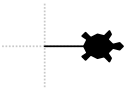


In this course, you'll be using Blockly - a visual programming language - to drive the turtle to make a satellite!

1.1.2. Move forward

To draw a satellite, first we have to be able to move! Click the ▶ button:





When you run the Blockly code, it makes the turtle move forward!

The number is the number of *turtle steps* to move. A bigger number will move the turtle further!

Try changing 30 to 100, and running it again.

1.1.3. Turning corners

The turtle always starts facing to the right of the screen.

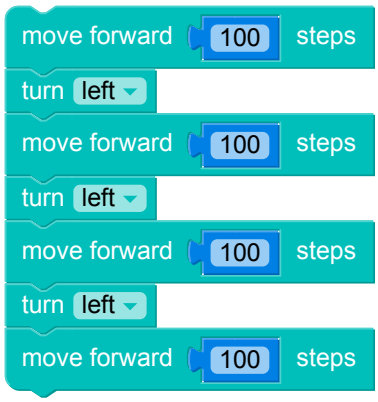
If you want to change which way the turtle is facing, you can use the turn left block. To make the block turn right, you need to click on left and select **right** from the dropdown list.

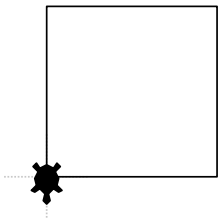
Click the ▶ button to see the turtle turning left:

Now the turtle is facing up! Change it to turn right by clicking the dropdown in the block and run it again.

1.1.4. Drawing a shape

You can combine the move forward and turn blocks to draw shapes:



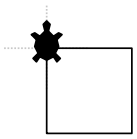


1.1.5. Problem: Starting from Square One

Write a program to draw a square below the turtle.

Each side of the squre should be 50 **turtle steps** long.

The easiest way to do this is to use turn right rather than turn left. The turtle starts in the top left corner of the square:

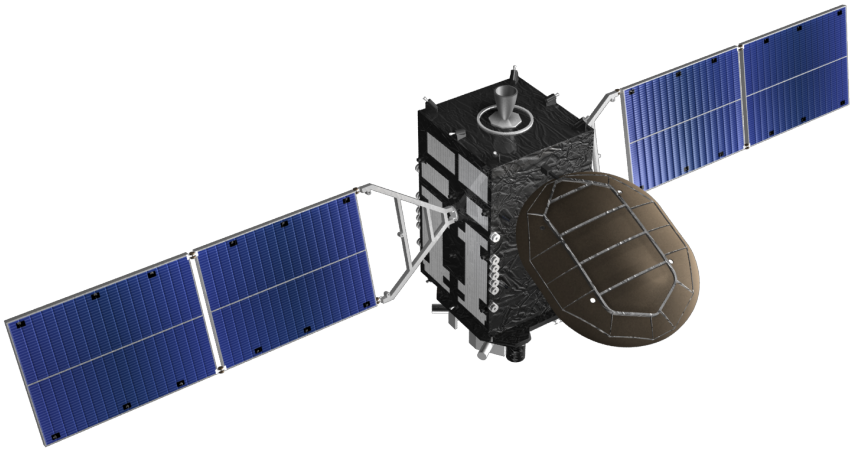


Testing

- [ ] Testing the top of the square.
- [ ] Testing the right side of the square.
- [ ] Testing the bottom of the square.
- [ ] Testing the left side of the square.
- [ ] Testing the whole square.
- [ ] Testing for no extra lines.
- [ ] **Fantastic, you've drawn the *right* (pun intended) square!**

1.1.6. Problem: Two solar panels

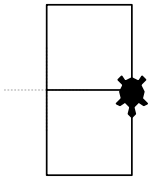
The QZS-3 satellite has a pair of solar panels on each side.



Provided by: National Space Policy Secretariat, Cabinet Office | Source: qzss.go.jp

Write a program to draw two square solar panels. Each side of each panel should be 50 **steps** long.

The turtle should start between the two panels:



Testing

- [ ] Testing the middle of the panels.
- [ ] Testing the bottom of the panels.
- [ ] Testing the top of the panels.
- [ ] Testing the whole panels.
- [ ] Testing for no extra lines.
- [ ] **Great work! You made some solar panels!**

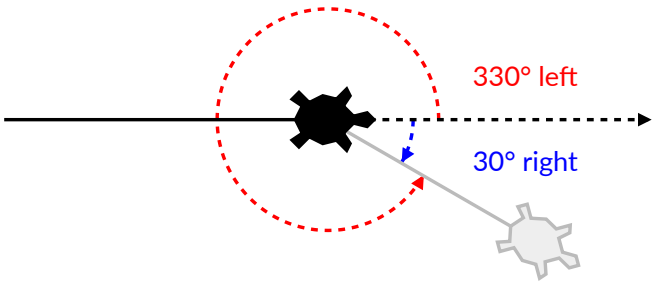
1.2. Angles

1.2.1. Turning different angles

So far our turn block has been *left* or *right* and this has been a 90 degree turn. We can turn the turtle different angles using a different turn block...

You can think of an *angle* as a *change of direction*. The blue block can have any number typed into it. Angles are measured in *degrees*. A 360° turn is a complete circle. Other turns are fractions of 360°.

Try our interactive diagram! You can drag the grey turtle around.

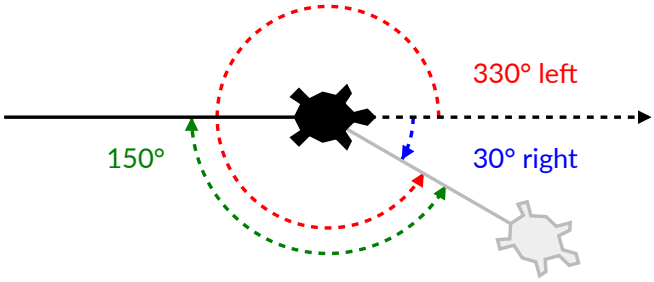


You can see that the angle is the difference between the direction the black turtle is facing and the direction the grey turtle is facing. Most shapes can be drawn as a combination of lines and directions.

1.2.2. Use our turn calculator!

We've added a green *internal* angle between the two lines in our diagram.

You can see the 180° – 30° = 150° turn you need:

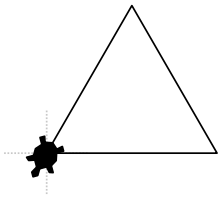


If you get stuck with angle calculations, use the diagram!

1.2.3. Problem: Equilateral triangle

Write a Turtle program to draw an equilateral triangle, with the sides being 100 **turtle steps** long. All angles *inside* an equilateral triangle are 60°.

The output of your program should look like this:



The *bottom left corner* of the triangle is where the turtle starts.

> **Hint**
>
> You need to do an angle calculation to draw this shape!

Testing

- [ ] Testing the bottom of the triangle.
- [ ] Testing the right side of the triangle.
- [ ] Testing the left side of the triangle.
- [ ] Testing the whole triangle.
- [ ] Testing for no extra lines.
- [ ] **Fantastic, you've drawn an equilateral triangle with `turtle`!**

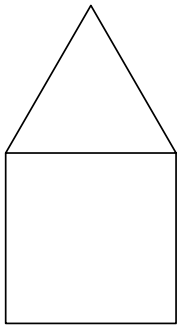
1.2.4. Problem: Dragon's body

The body of the Space-X Dragon looks like a bit like a triangle on top of a square.

Use the turtle to draw body of the Dragon!

The triangle at the top should have angles that are all 60° and all sides of the house should be 100 turtle steps long. The sides of the roof will also be 100 steps long.

The result should look like this:



The top left corner of the square is where the turtle starts.

Optional challenge!

Try drawing the picture in one line, without drawing over the same line twice. Think about the *order* you need to draw the lines in.

Testing

- [ ] Testing the bottom of the triangle (top of the square).
- [ ] Testing the right wall of the Dragon (the right side of the square).
- [ ] Testing the left wall of the Dragon (the left side of the square).
- [ ] Testing the floor of the Dragon (the bottom of the square).
- [ ] **Testing the body of the Dragon** (the square).
- [ ] Testing the left side of the triangle.
- [ ] Testing the right side of the triangle.
- [ ] **Testing the triangle of the Dragon** (the triangle).
- [ ] **Testing the whole Dragon.**
- [ ] Testing for no extra lines.
- [ ] **Well done, your Turtle is now an astronaut!**

2

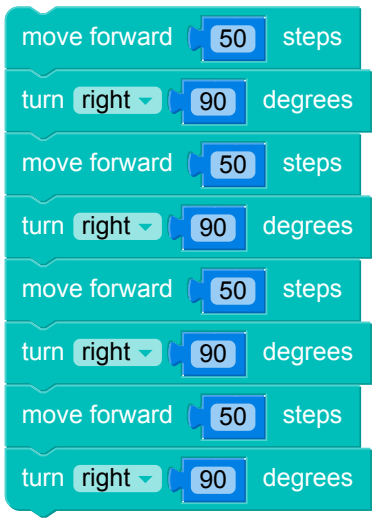
LOOPING WITH THE TURTLE

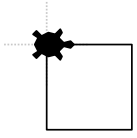
2.1. Loops with the turtle

2.1.1. Drawing shapes with loops

You probably noticed that you repeated yourself a lot in turtle programs. Using loops makes turtle much less repetitive!

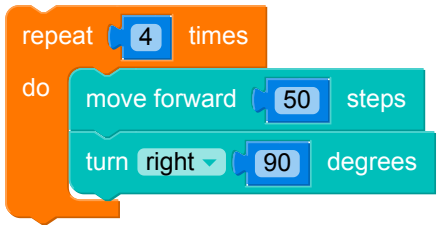
Drawing a square the long way:

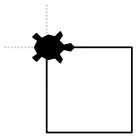




If we used a loop, then we wouldn't have to repeat same two blocks over and over again.

Here's a much shorter way of drawing a square, using loops:



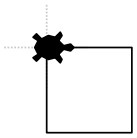


2.1.2. Problem: Back to Square One

Write a program to draw a square below the turtle.

Try using the repeat block to make life easier!

Each side should be **50 turtle steps** long.



Hint

Look for patterns that repeat themselves, and place those inside the repeat block!

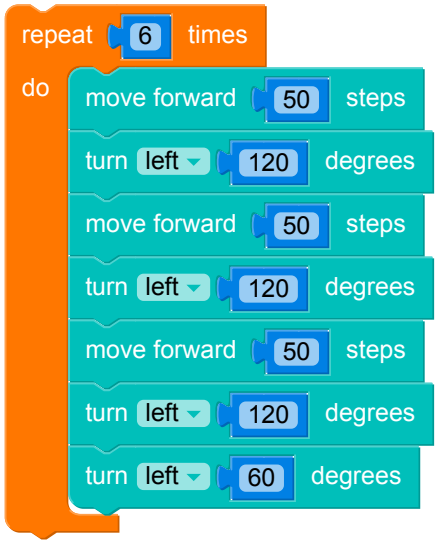
Testing

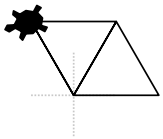
- [ ] Testing the top of the square.
- [ ] Testing the right side of the square.
- [ ] Testing the bottom of the square.
- [ ] Testing the left side of the square.
- [ ] Testing the whole square.
- [ ] Testing for no extra lines.
- [ ] **Fantastic, you've drawn the *right* (pun intended) square!**

2.2. Loops and Movement

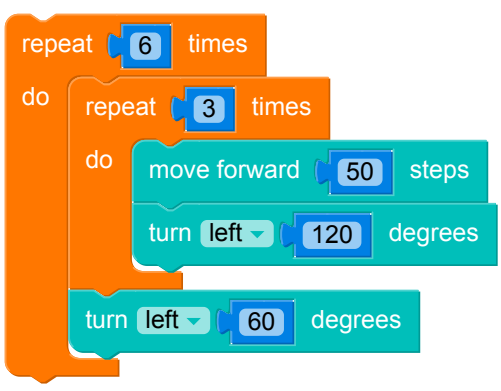
2.2.1. Repeat Block

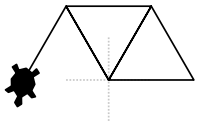
We can use loops like the repeat block to make some great patterns!





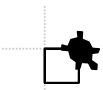
Here's the same code again as above, but simplified by adding *another loop*!



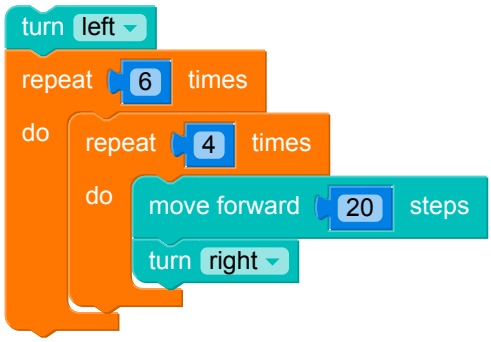


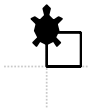
2.2.2. Squares on squares on squares

When we use a loop block, it repeats everything inside of it. Let's say we wanted to draw 6 squares. We could try something like this:



Any repeated pattern can be made simpler by using *two loops*!

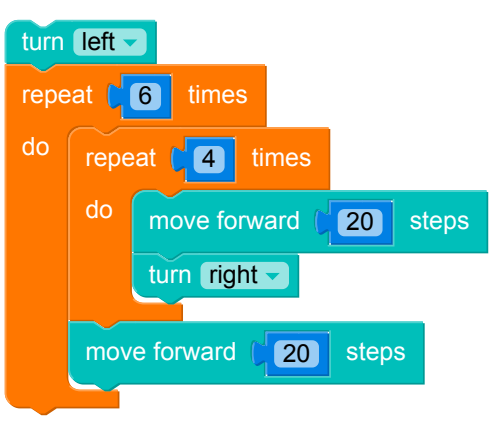


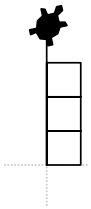


Six squares are drawn, but **on top of each other**! To draw more squares, we'll need to add in an extra **move forward** step.

2.2.3. Moving between loops

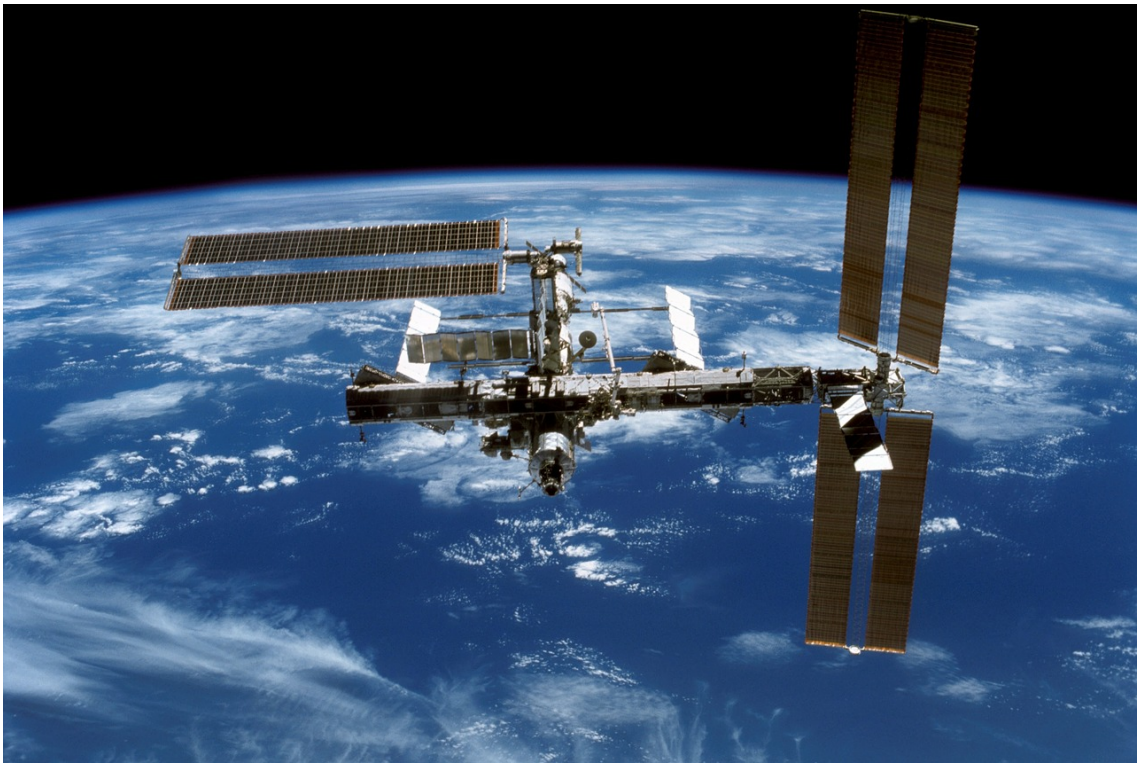
Here, we've added an extra **move forward** step at the end of the instructions, so that each time the square is drawn in a new position, pointing the right way.





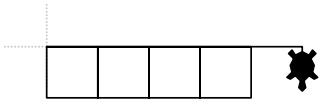
2.2.4. Problem: So many solar panels

The International Space Station has many solar panels in a row!



Let's draw a row of solar panels!

The row should be 6 solar panels long, and each panel should have sides **30 turtle steps** long.



Testing

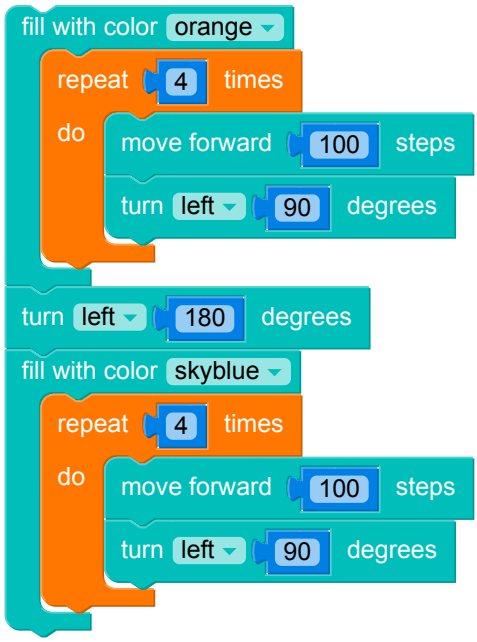
- [ ] Testing the right side of the first square.
- [ ] Testing the left side of the first square.
- [ ] Testing the whole square.
- [ ] Testing for no extra lines.
- [ ] **Great work!**

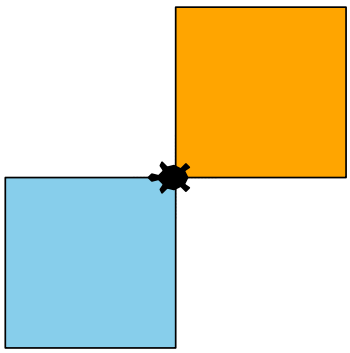
2.3. Satellite colour

2.3.1. Fills and loops

The fill with color block can be used to fill in a shape with colour.

Any shapes which you draw inside of fill block will be filled with the colour you choose (at the end of drawing the shape).

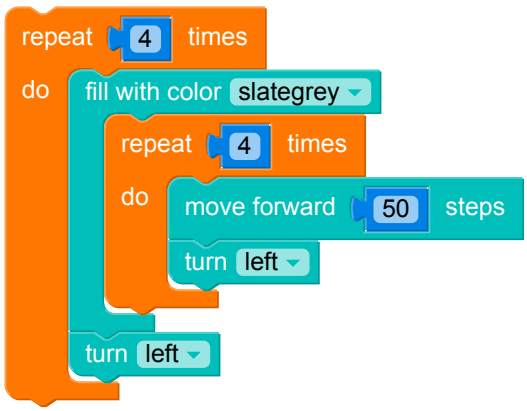


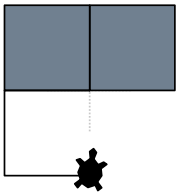


If we tried to put the fill *inside* the loop, then it would try to separately fill each side of the shape, which wouldn't work!

2.3.2. Choosing the fill colour

We can select the colour we want by choosing the colour name in the fill with color block dropdown.





Try filling it with a different colour!

2.3.3. Problem: Satellite wings

Satellites are usually powered by solar-panel wings! They need to be dark coloured to absorb the sun's light!

The Solar Terrestrial Relations Observatory (STEREO) Satellite

Let's draw a wing with **4 panels**, each of them a square with a length of `30`. The colour of each panel is `slategrey`.

See an example below!

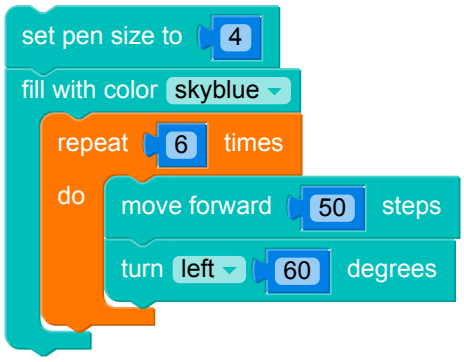
Testing

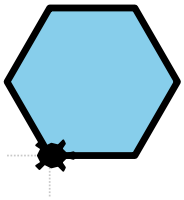
- [ ] Testing the outlines of all the panels.
- [ ] Testing the fill colour.
- [ ] Testing for no extra lines.
- [ ] **Nice work! You now have the power!**

2.3.4. Pen size

We can make the lines thicker by using the pensize block *before* you draw the line.

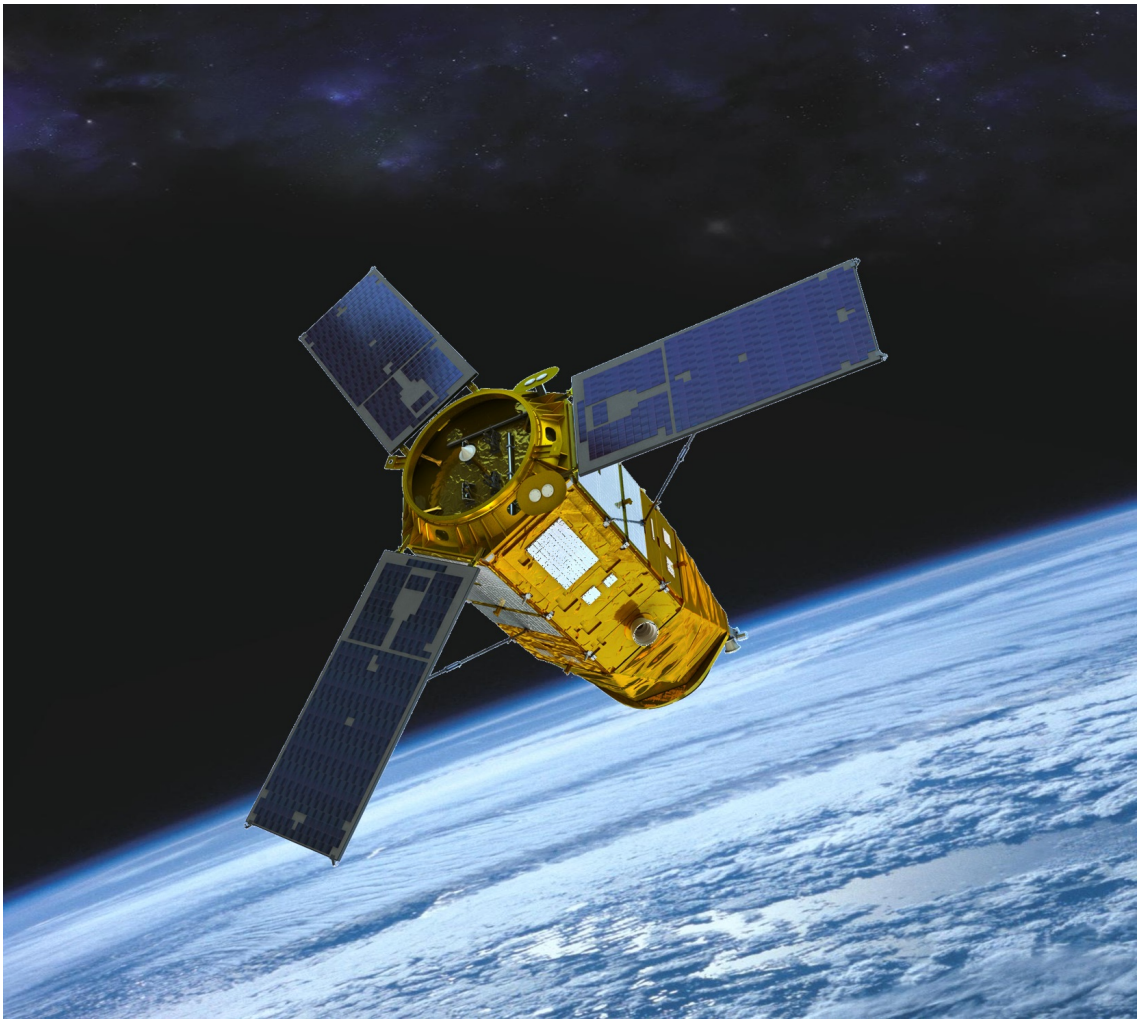
Here's an example:





2.3.5. Problem: Satellite body

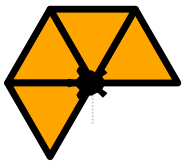
Some satellites are shaped like hexagons.



An artist rendering of the KOMPSAT-3A | Source: Korean Aerospace Research Institute KARI

It's time to give your satellite a body! The body is going to be made up of **6** eqilateral triangles, with a side length of 50 steps, and a pen size of 4.

See an example below!



Testing

- [ ] Testing all the lines.
- [ ] Testing the fill colour.
- [ ] Testing for no extra lines.
- [ ] **Great work, your satellite has a body!**

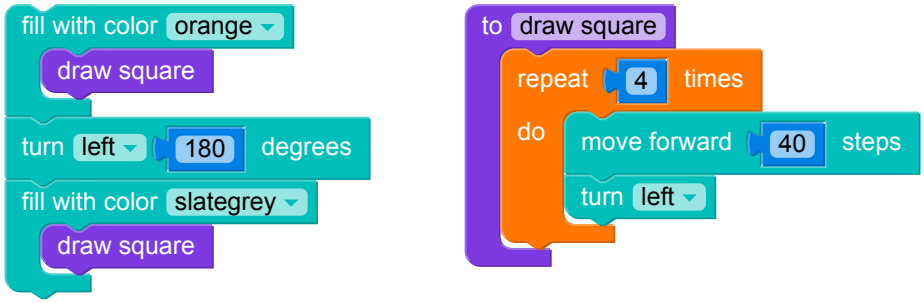
3

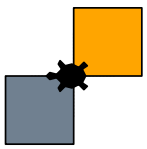
FUNCTIONS

3.1. Functions

3.1.1. Using functions

Functions are repeatable blocks of code. You can call a function by using the label, which can be anything you like. Here, the function is called draw square, have a look at it run!



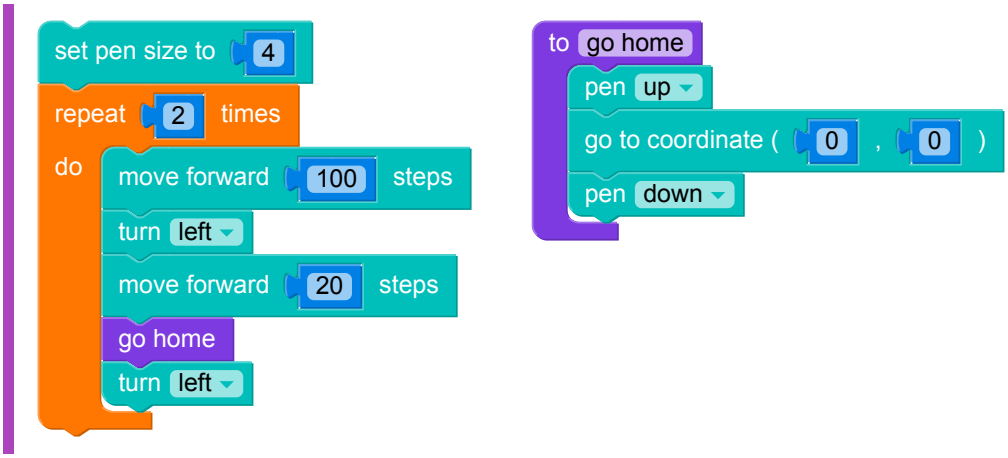


See how we called the draw square function *twice*, but we only had to write it *once*. You can run functions as many times as you like.

3.1.2. Goto function

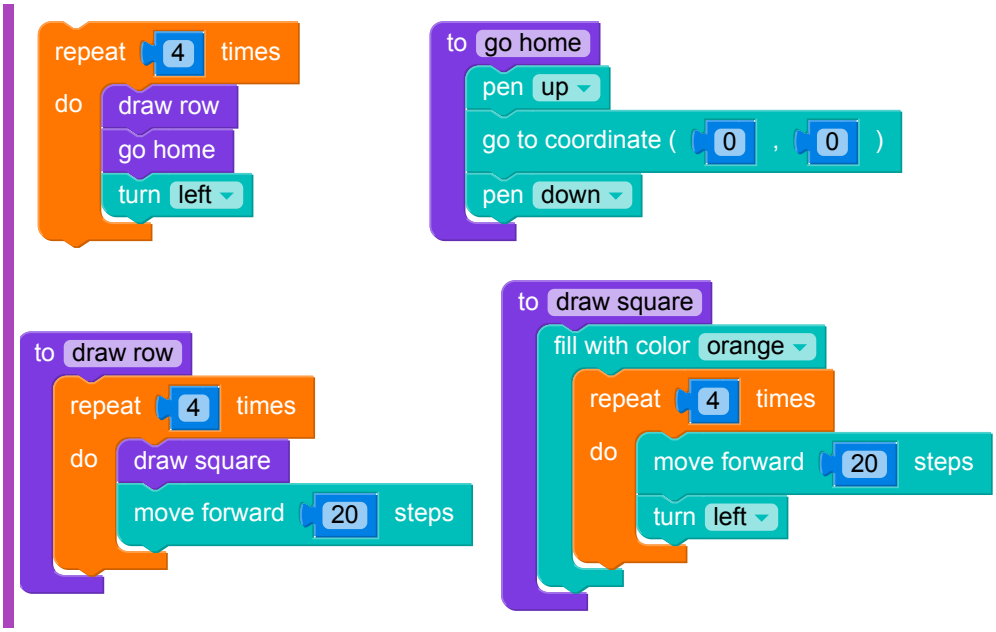
The go to block can be really useful if you want to repeat a pattern that doesn't end in the center. Combined with pen up and pen down, this can be really useful!

Let's make a go home function that uses these blocks!



3.1.3. Functions within functions

Functions make things easier! You can make functions to perform a specific task, and then *call* that function as many times as you like.



3.1.4. Problem: Part of the satellite

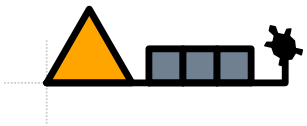
It's time to draw the first part of the satellite! We're going to draw a whole section!

The pen size is 4.

The center is a equilateral triangle, with a side length of **50** and is filled with orange.

The wings start **60** steps from the middle, and each square has a width and height of **20** steps. The fill colour is slategrey.

Here's what your satellite should look like:



Think about how you can split this problem up into smaller problems, and treat them separately using functions!

Testing

- [ ] Testing all the lines.
- [ ] Testing the fill colour.
- [ ] Testing for no extra lines.
- [ ] **Cool! We've nearly built the satellite!**

3.1.5. Problem: Build a satellite!

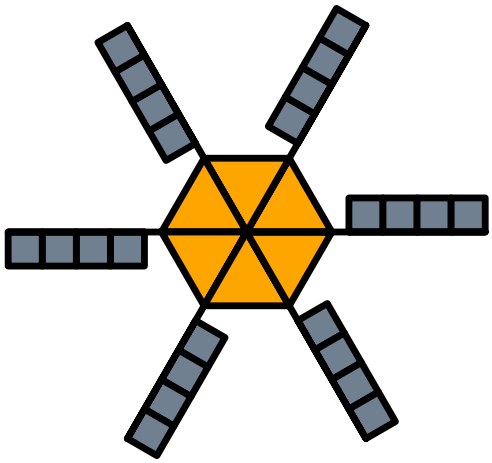
Now it's time for the last activity! Let's draw your completed satellite!

The pen size is 4.

Each center piece is a equilateral triangle, with a side length of **50** and the fill colour orange.

The wings start **60** steps from the middle, and each square has a width and height of **20** steps. The fill colour is slategrey.

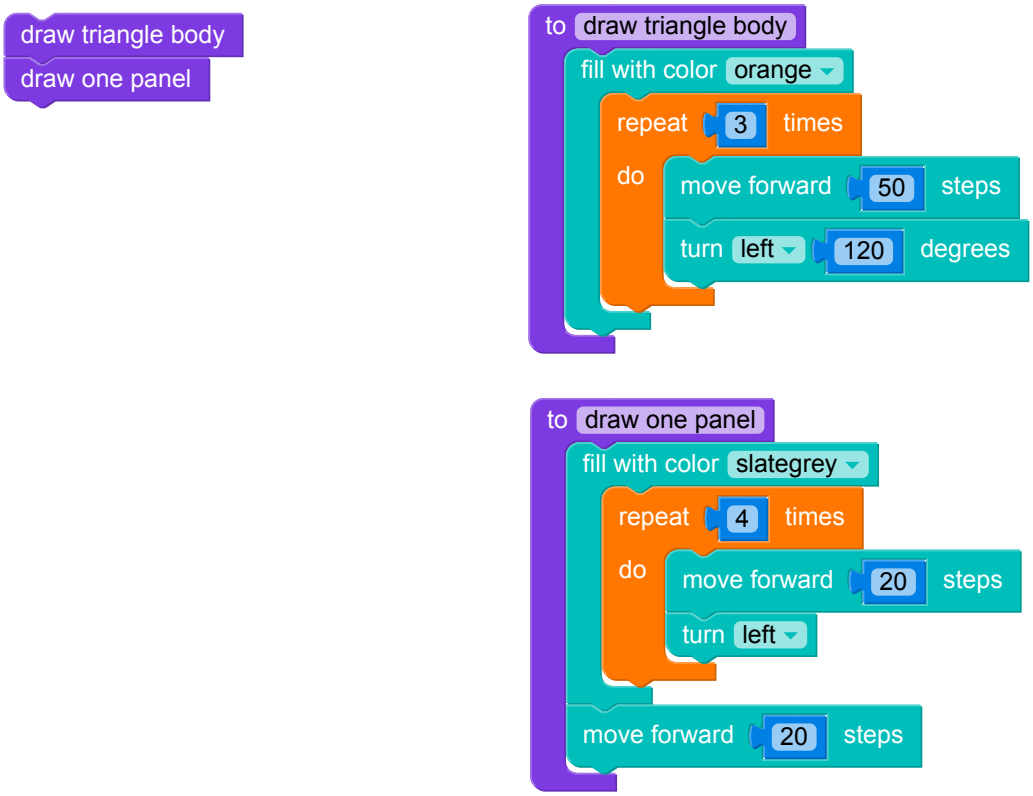
See what it should look like below!



There are many possible solutions to this activity! We've given you a start with functions to draw the triangle body and a single solar panel. Think about how you can split the entire satellite into smaller components, and then draw those compontents multiple times.

You'll need

program.blockly



Testing

- [ ] Testing all the lines.
- [ ] Testing the fill colour.
- [ ] Testing for no extra lines.
- [ ] **Amazing! You've built your own satellite! High five!**

3.2. Playground

3.2.1. Congratulations!

Congratulations on drawing your own satellite! You rock!

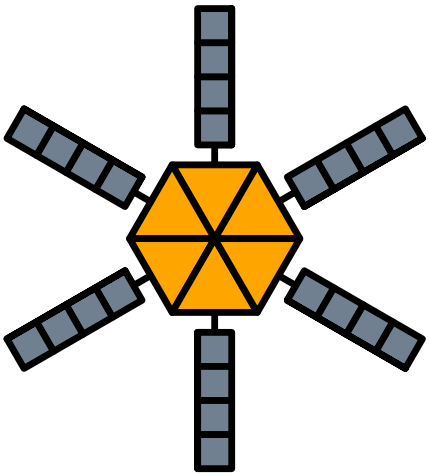
Don't forget to register with Moonhack central (https://www.moonhack.com/) to be part of the World Record attempt!

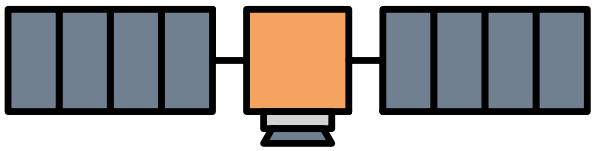
If you want to explore the turtle a bit more, try out our Turtle Geometry (https://groklearning.com/course/aca-dt-56-bk-turtle/) Challenge!

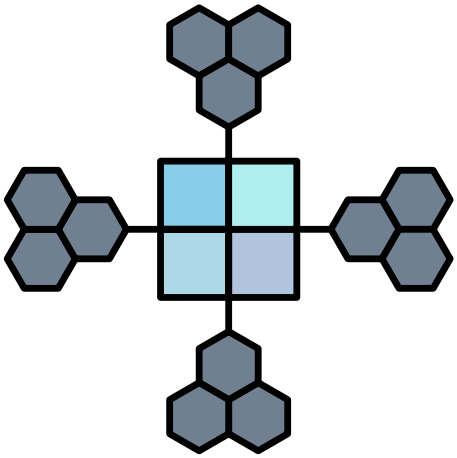
3.2.2. Problem: Satellite playground

What's this, thought you were finished?

Now you can make your own satellite with the turtle! Try and make a cooler one than the previous activity! Here are some examples!







Testing

- [ ] **This is a playground question, there is no right or wrong!**