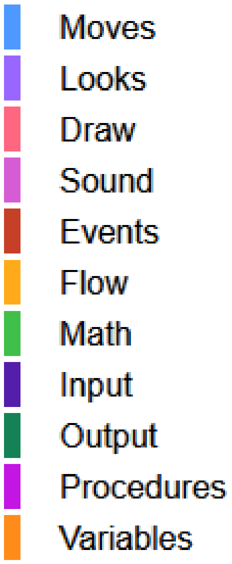
Logo Blocks Tutorial ~ Advanced Mode

If you haven't use Logo Blocks before, start with the Beginner Mode tutorial.
Next, go through the Intermediate Mode tutorial.
Finally, return here to learn even more block commands!

When Logo Blocks first starts up, you see the **Beginner** set of blocks. Click the word **Advanced** to get to more commands. You can click on any of the words in the list at the left to get to the blocks in that group.

In **Advanced** Mode, blocks are organized into eleven categories.



Moves: turn and move the turtle, set the speed, set the screen mode
Looks: set the turtle shape and size, hide or show the turtle
Draw: set colors and pen size, pen up or down, draw shapes, fill, erase
Sound: say words or sounds
Events: act on certain events that occur
Flow: use commands that control what happens (IF…THEN and more)
Math: use random numbers and math operations
Input: work all types of data you can give to blocks
Output: print text in a dialog
Procedures: write simple procedures with or without an input
Variables: create and use variables to create flexible procedures

Some of the **Advanced** categories have the same blocks as in **Intermediate** mode, so this tutorial won't cover them. Refer to the Intermediate mode tutorial for more information about using blocks in these categories:

- **Moves**
- **Looks**
- **Draw**
- **Sound**

Let's learn some more block commands!

Moves

These blocks are the same as in Intermediate mode. Refer to that tutorial for help using them.

Looks

These blocks are the same as in Intermediate mode. Refer to that tutorial for help using them.

Draw

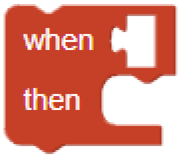
These blocks are the same as in Intermediate mode. Refer to that tutorial for help using them.

Sound

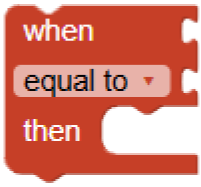
These blocks are the same as in Intermediate mode. Refer to that tutorial for help using them.

Events

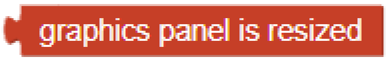
These commands allow you to look for things to happen and then react to those situations. Handling events can seem rather complicated at first, but once you try some examples and edit them to do slightly different things, you will see how they work.



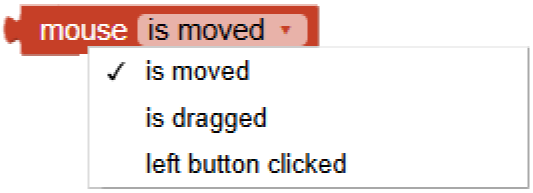
WHEN blocks take an input about a situation that might change.
You can tell Logo Blocks what to do what under certain circumstances



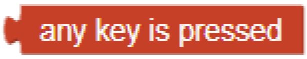
The events you can look for are shown below.



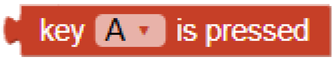
This event detects if the size of the graphics panel changes.



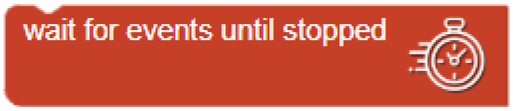
This event detects if the mouse is moved, dragged, or its left button is clicked. You can choose the condition.



This event is triggered if the user presses any key.



This event is triggered if the user presses a specific key. You can set the key to any letter (works with both upper and lower case), number, the space bar, or an arrow key.



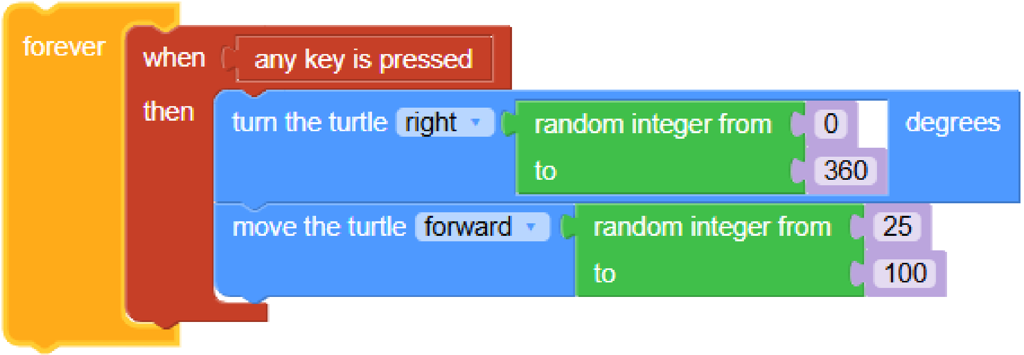
This block causes the program to halt and wait for the specific event to occur. Also put this block at the bottom of your code when using events.

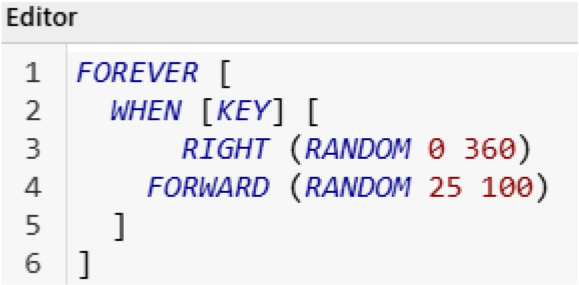
Let's see some examples of these blocks in use.

You can try these examples as is and then edit the blocks so act differently. It's up to you!

Example A:

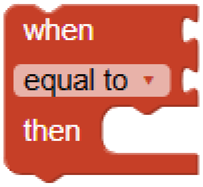
In this example, pressing a key moves and turns the turtle random amounts.
It doesn't pay attention to what keys are pressed.
Click the red Stop button to end the program.

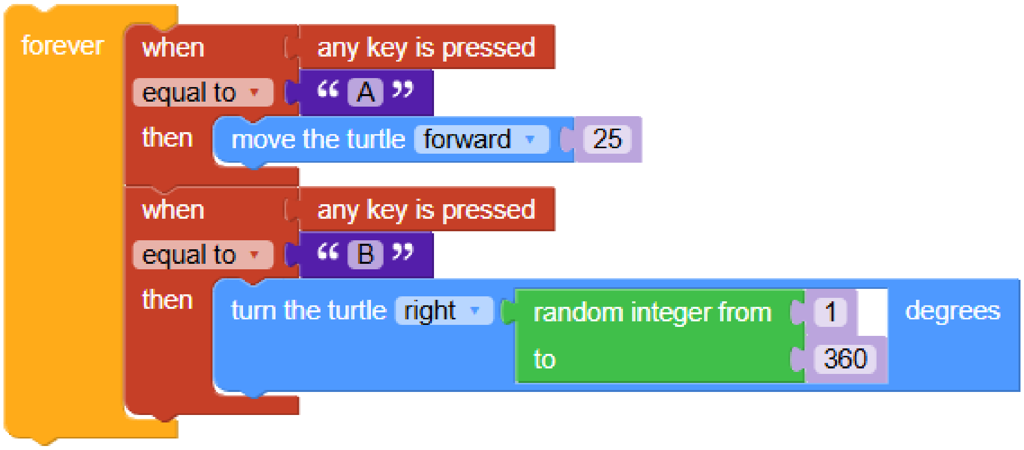



Editor

```
FOREVER [
  WHEN [KEY] [
      RIGHT (RANDOM 0 360)
    FORWARD (RANDOM 25 100)
   ]
]
```

If you want different keys to perform different actions, use this block and this code:



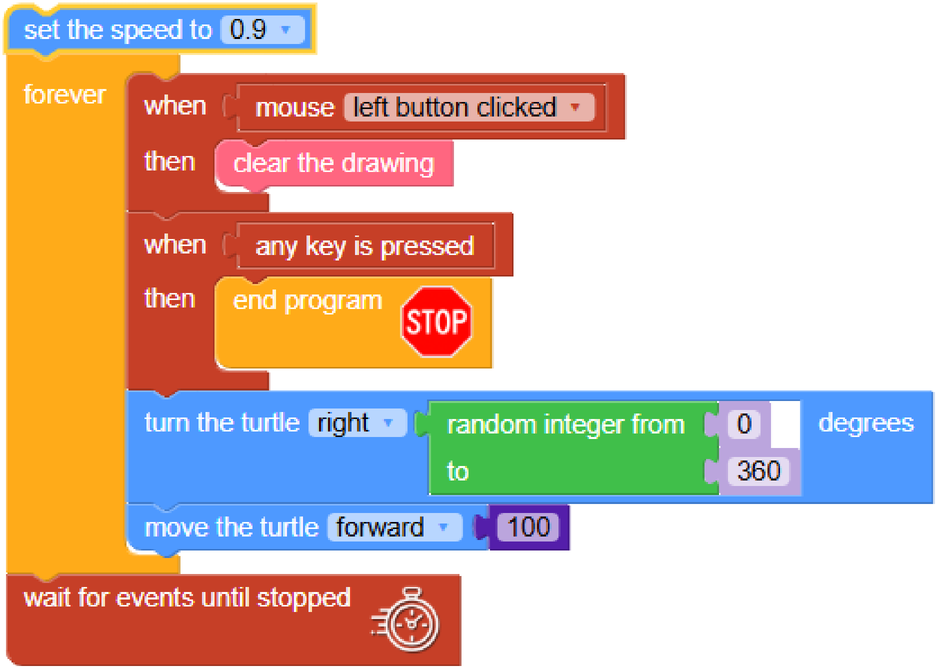


```
forever
  when any key is pressed equal to "A"
  then move the turtle forward 25
  when any key is pressed equal to "B"
  then turn the turtle right random integer from 1 to 360 degrees
```

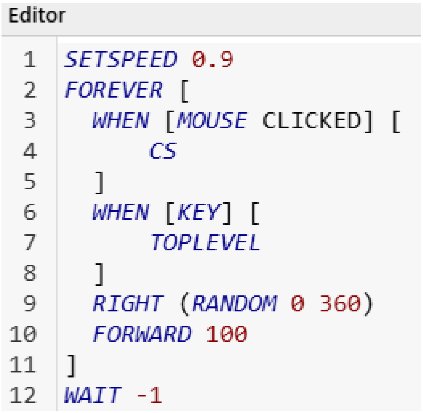
Pressing the A key moves the turtle forward 25 steps. Pressing B turns it a random amount.

Example B:

In this example, clicking the mouse button erases the drawing.
Pressing any key stops the program.
The rest of the time, the turtle repeatedly turns a random number and moves forward 100 steps. The speed is set to 0.9 so you watch the turtle draw.



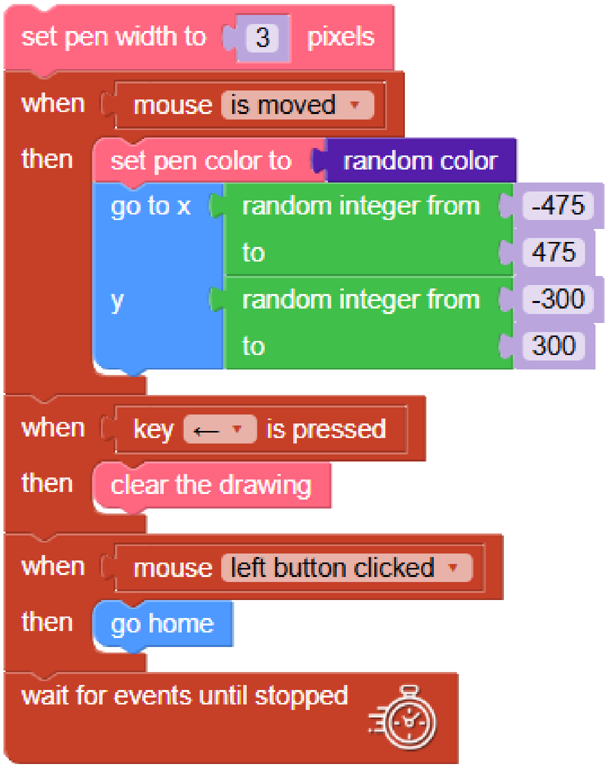
```
set the speed to 0.9
forever
  when mouse left button clicked
  then clear the drawing
  when any key is pressed
  then end program STOP
  turn the turtle right random integer from 0 to 360 degrees
  move the turtle forward 100
wait for events until stopped
```



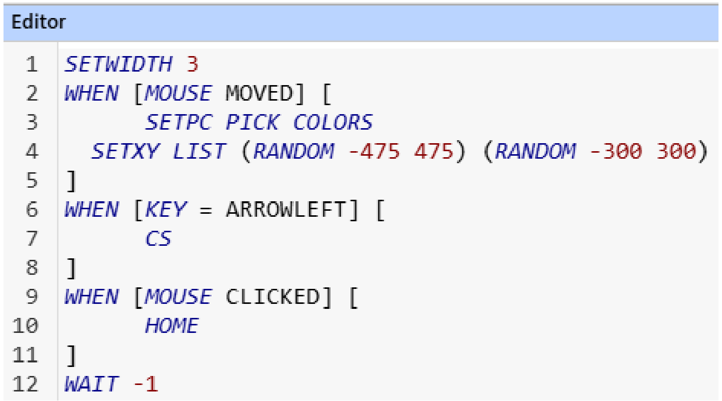
```
SETSPEED 0.9
FOREVER [
  WHEN [MOUSE CLICKED] [
      CS
  ]
  WHEN [KEY] [
      TOPLEVEL
  ]
  RIGHT (RANDOM 0 360)
  FORWARD 100
]
WAIT -1
```

Example C:

In this example, moving the mouse draws a line to a random x-y coordinate location.
Pressing the left arrow key erases the drawing.
Clicking the mouse moves the turtle to its home in the middle of the screen.
You may wish to explore different turtle speeds.



```
set pen width to 3 pixels
when mouse is moved
then set pen color to random color
     go to x random integer from -475
             to 475
           y random integer from -300
             to 300
when key ← is pressed
then clear the drawing
when mouse left button clicked
then go home
wait for events until stopped
```

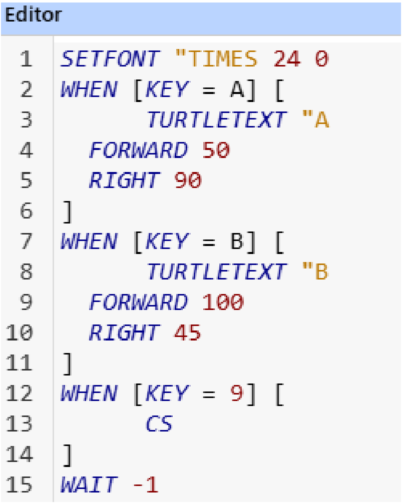


Editor

```
SETWIDTH 3
WHEN [MOUSE MOVED] [
       SETPC PICK COLORS
  SETXY LIST (RANDOM -475 475) (RANDOM -300 300)
]
WHEN [KEY = ARROWLEFT] [
       CS
]
WHEN [MOUSE CLICKED] [
       HOME
]
WAIT -1
```

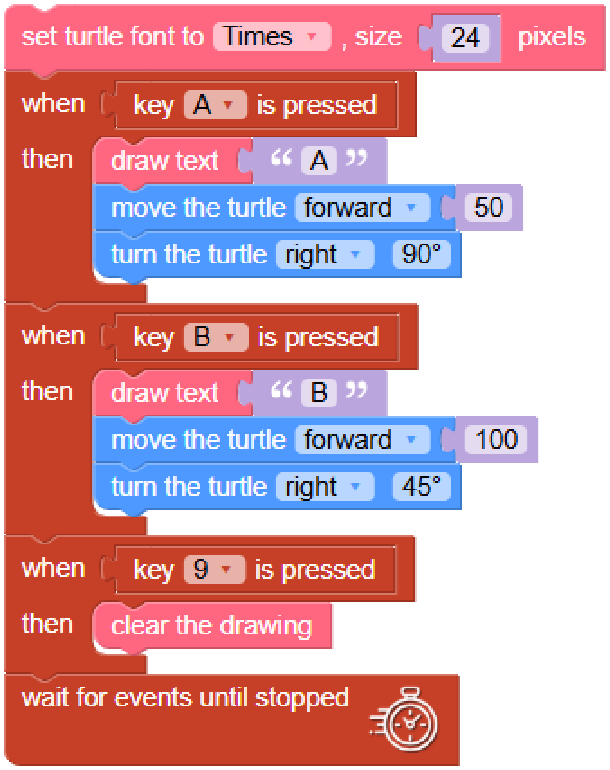
Example D:

In this example, pressing A moves and turns the turtle and prints the letter A on the screen.
Pressing B moves the turtle a different distance and prints the letter B on the screen.
Pressing the number 9 erases the screen.



Editor

```
SETFONT "TIMES 24 0
WHEN [KEY = A] [
       TURTLETEXT "A
  FORWARD 50
  RIGHT 90
]
WHEN [KEY = B] [
       TURTLETEXT "B
  FORWARD 100
  RIGHT 45
]
WHEN [KEY = 9] [
       CS
]
WAIT -1
```

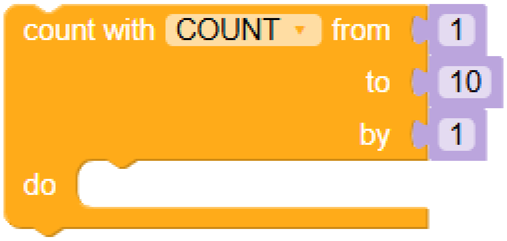


```
set turtle font to Times , size 24 pixels
when key A is pressed
then draw text " A "
     move the turtle forward 50
     turn the turtle right 90°
when key B is pressed
then draw text " B "
     move the turtle forward 100
     turn the turtle right 45°
when key 9 is pressed
then clear the drawing
wait for events until stopped
```

Flow

Use these blocks to control the flow of the program. **If/Then** blocks are covered in the Intermediate Mode tutorial, as are the **compare**, **repeat**, **forever**, **wait**, and **end program** blocks.

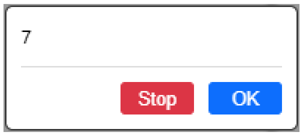
Here are the new blocks introduced in the Advanced Mode.



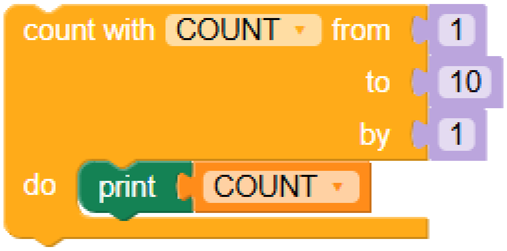
```
count with COUNT from 1
                   to 10
                   by 1
do
```

The **count** block uses a variable named COUNT to keep track of numbers. You can tell it to count by 1s, 2s, 3s, 10s, 100s, or any number. Tell it what number to start at and when to end.

Here is a simple example to start with. It simply counts from 1 to 10. The numbers appear one at a time in an alert box. If you change to a very large "to" number and don't want to finish the sequence, you can click the red **Stop** button in the alert box. Otherwise click **OK** to continue counting.



```
7
```

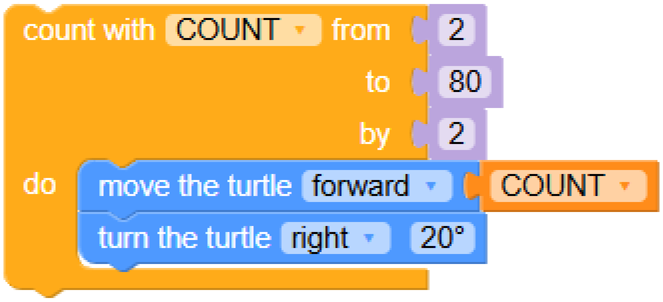


```
count with COUNT from 1
                   to 10
                   by 1
do  print COUNT
```

You'll find the **print** block in the Output section.

The **COUNT** variable is in the Variables section.

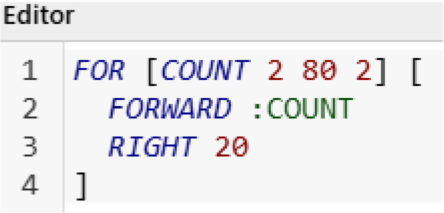
This more complicated use of this block creates a spiral design.



```
count with COUNT from 2
                   to 80
                   by 2
do  move the turtle forward COUNT
    turn the turtle right 20°
```

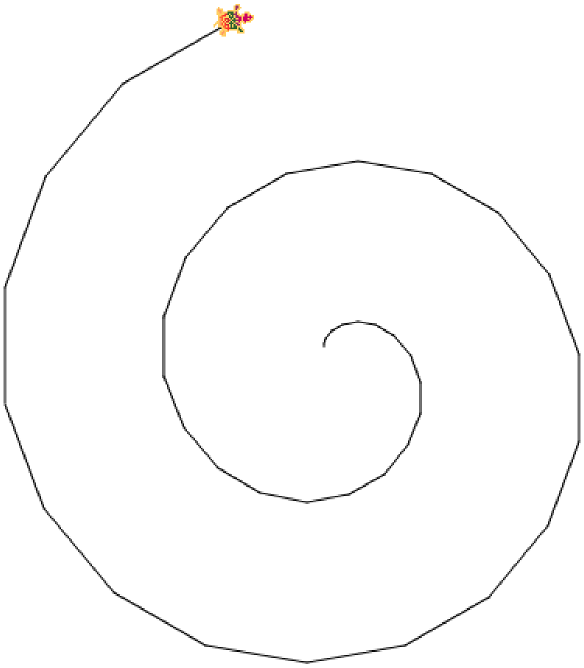
Can you imagine what the code will draw? Each step of the way, the turtle moves forward the value of COUNT, which increases by 2 every time. This is a type of loop. It uses the FOR command in Logo.

See the picture it draws on the next page:

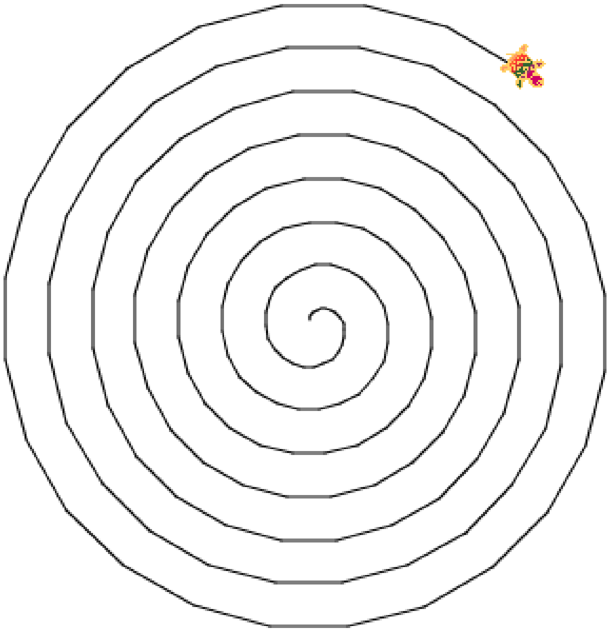


```
FOR [COUNT 2 80 2] [
  FORWARD :COUNT
  RIGHT 20
]
```

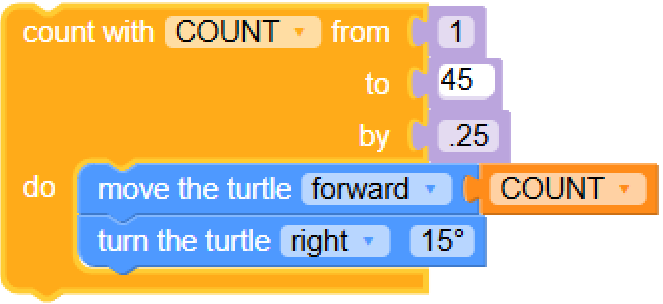
Experiment with the numbers and create your own version!



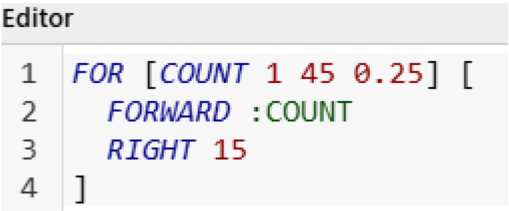
Can you figure out how to create this variation?



Notice that you can use a number less than 1 for the **by** value. This code uses .25. That means that the forward distance increases by a quarter of a step each time: 1, 1.25, 1.5, 1.75, 2, 2.25, 2.5, etc., all the way to 45.



```
count with COUNT from 1
                   to 45
                   by .25
do  move the turtle forward COUNT
    turn the turtle right 15°
```



```
FOR [COUNT 1 45 0.25] [
  FORWARD :COUNT
  RIGHT 15
]
```

Math

With these **Math** blocks, you can work with numbers in many different ways. Even though a couple of these blocks were covered in the **Intermediate** mode tutorial, they are also included here.

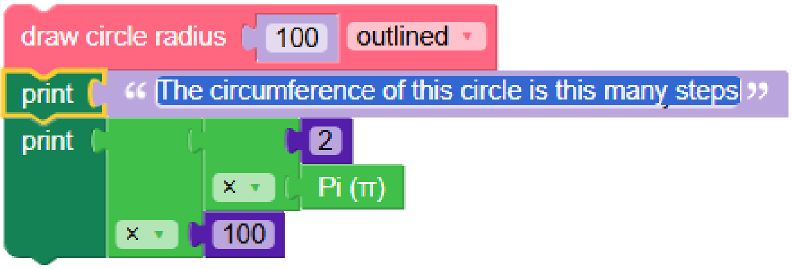
Use this block when you want to insert a number into a block. You can enter the number you want. You have used this block before.

The block gives you the value of pi, which can be written as π. Pi is a value used to compute the circumference and area of circles.

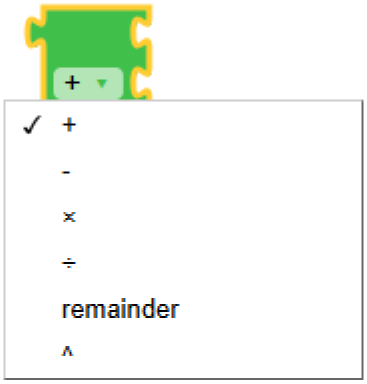
The circumference of a circle is $2\pi R$, or 2 times pi times the length of the radius.

The area of a circle is πR^2, or pi times the radius squared.

Here is block code that computes the circumference of a circle.

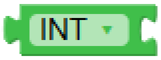


Can you write the code to figure out its area using pi?

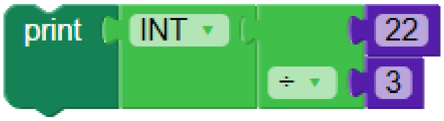


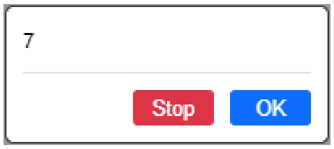
Use the **operations** block when you want to do math. Connect a number block (see above) in both spots. You can then add, subtract, multiply, or divide them, find the remainder, or use an exponent. Three to the second power, or three squared, is an example of an exponent, written like this: 3^2. It means 3 x 3 or 9.

For more examples, see the Intermediate tutorial.



INT is short for INTEGER. It takes a number as input and returns a whole number, like this:

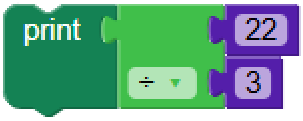


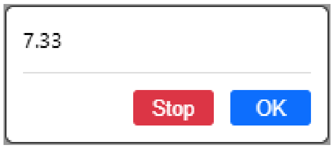


```
7
```

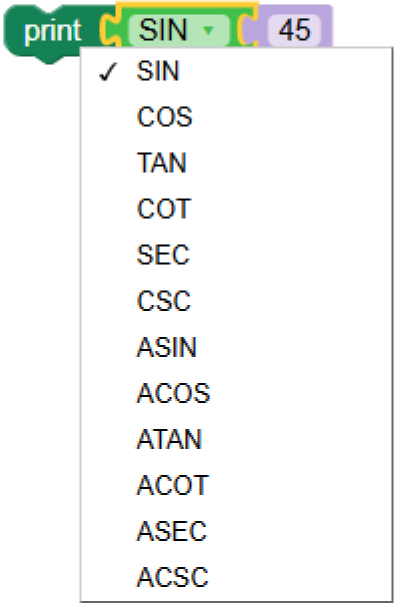
The input is 22/3. The integer result is 7.

If you remove the INT block, the result is no longer a whole number. It is a decimal. See the difference?





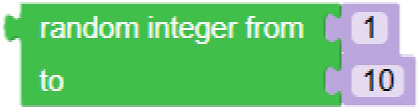
```
7.33
```



This block lets you explore all the trig functions: sine, cosine, tangent, cotangent, secant, cosecant, arcsine, arccosine, arctangent, arccotangent, arcsecant, and arccosecant.

If you don't understand these functions yet, don't worry! Trig is a high school math class.

This tutorial is not designed to teach you trig, but provides you with useful tools if you understand the math. If you are in high school, let your trig teacher know that you can use trig functions in Logo Blocks!



You've already seen examples of using the **random** block. Anytime you need a number, just replace the current value with this block. Enter numbers for the range you want and see what happens!

More sections of the Advanced Tutorial coming soon.