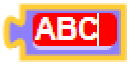
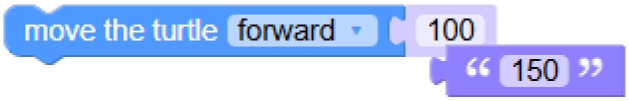
Words and Lists

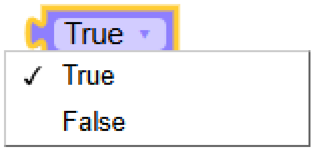
Inputs are pieces of information you use with blocks or procedures you write. The information can be numbers, letters, collections of words, colors, and much more, as you will see. You are already familiar with some of them.

These two blocks are designed for numbers and letters. You can't enter letters into the block with **123**. If you try, the background turns red, like this:

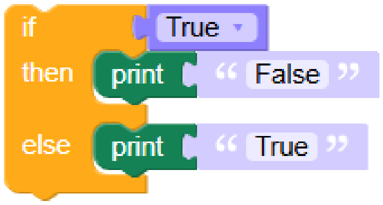


You can enter numbers into the text box, but you can't do math with them. They are no longer numbers. Logo Blocks won't let you connect a text box to a **forward** block, for example:





This block checks a condition and reports either True or False. Here's an example of using it.

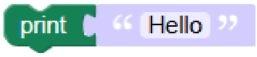


About Words and Lists

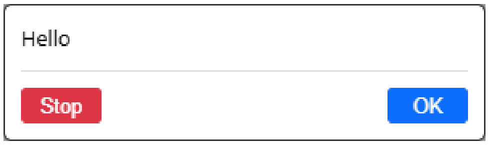
Many of these blocks use **words** and **lists**. A **list** is a collection of words or numbers, or even other lists. There are spaces between the items in a list. A list can even contain another list. A **word** is a numbers or letters with no spaces between them, like feb26.

About Alert Boxes

You may notice that the Editor code often includes this text when you use the ***print*** block,

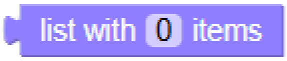


```
IGNORE ALERT `Hello`
```

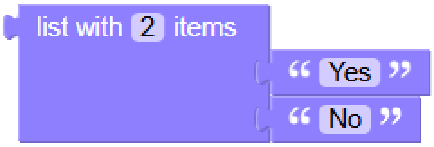


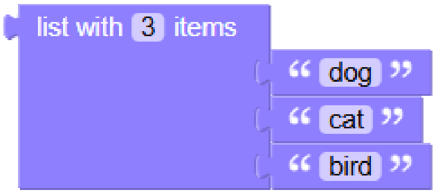
print displays its text in an alert box on the screen. (In the full Logo language, the PRINT command would put the text in the Listener window, which Logo Blocks doesn't have.) ALERT generally sends back information to Logo, so the IGNORE part of the Logo instruction means to ignore anything that the ALERT box might report back to it. You can ignore the *IGNORE ALERT* code.

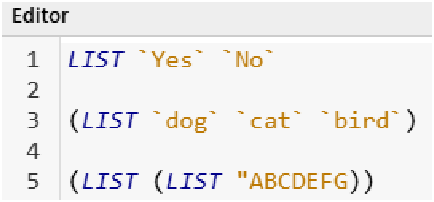
Let's learn more blocks that use words and lists.



Initially, the list block has no items. You can set the number of the items you want to include in the list. Here are some examples.

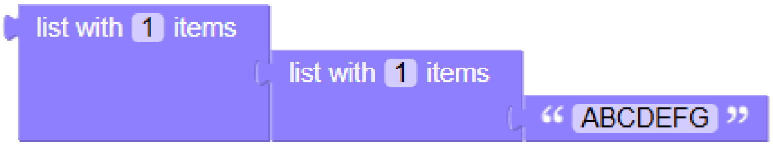






Editor

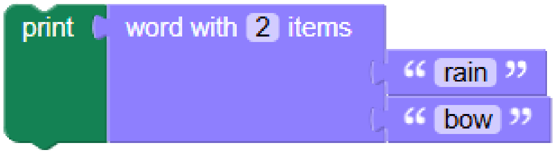
```
LIST `Yes` `No`

(LIST `dog` `cat` `bird`)

(LIST (LIST "ABCDEFG))
```



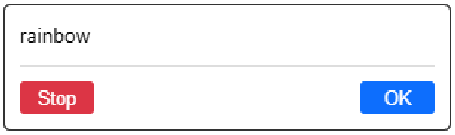
The code in the Logo Editor shows that it places parentheses around a list with more than or fewer than two items, which is the standard number of inputs for a list. If the text is in upper and lower case, it is called a string and starts and ends with a backquote, like this: `Yes`. If all the characters are uppercase or are letters, it starts with a quote mark.

```
word with 0 items
```

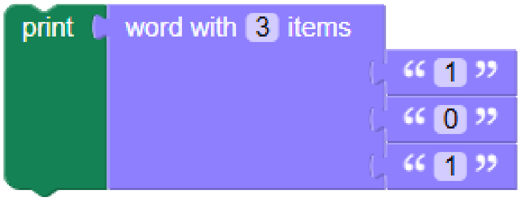
A word is a sequence of letters and/or numbers with no spaces between them. Here are some examples.



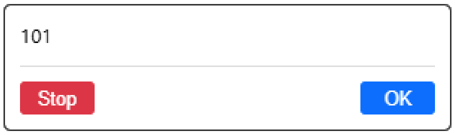
```
print word with 2 items
  “ rain ”
  “ bow ”
```



```
rainbow
```



```
print word with 3 items
  “ 1 ”
  “ 0 ”
  “ 1 ”
```

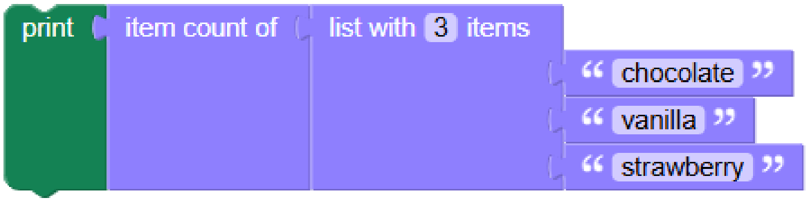


```
101
```

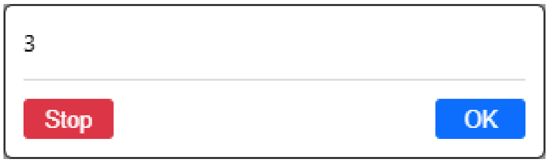
```
IGNORE ALERT WORD `rain` `bow`

IGNORE ALERT (WORD "1 "0 "1)
```

```
item count of
```

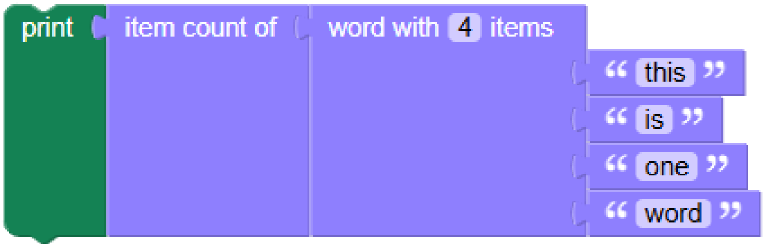
Item count of tells you how many items are in a word or list.



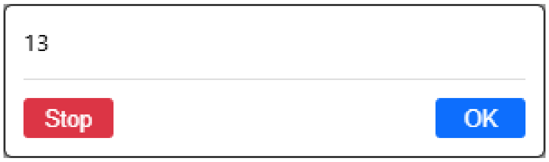
```
print item count of list with 3 items
  “ chocolate ”
  “ vanilla ”
  “ strawberry ”
```



```
3
```

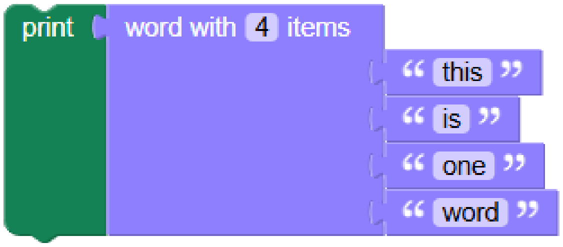


```
print item count of word with 4 items
  “ this ”
  “ is ”
  “ one ”
  “ word ”
```

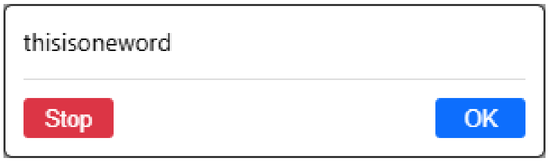


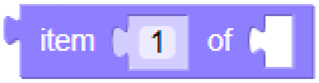
```
13
```

Are you surprised by that result? The result is 13 because it is counting all the letters in the word. If you print the word, like this:

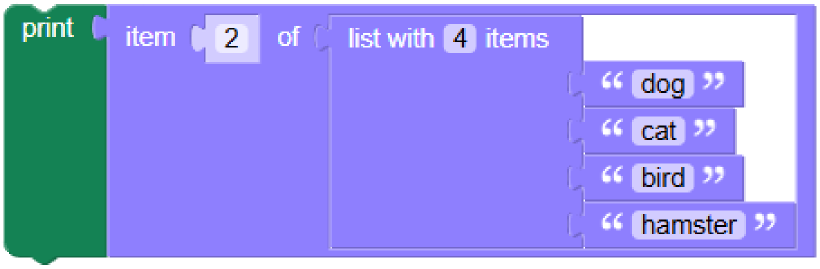


You’ll see why the result is 13.

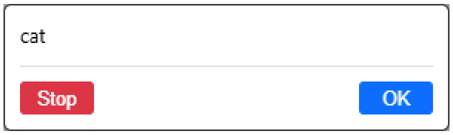

thisisoneword



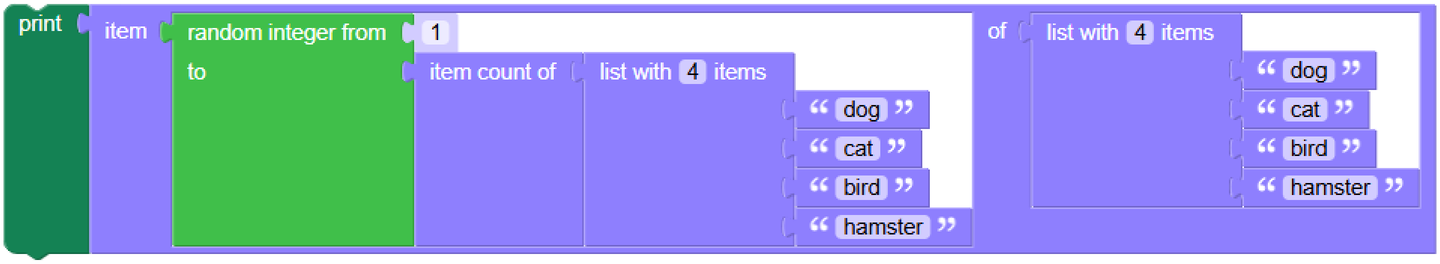
The ***Item*** block gives you the part of the list you specify.



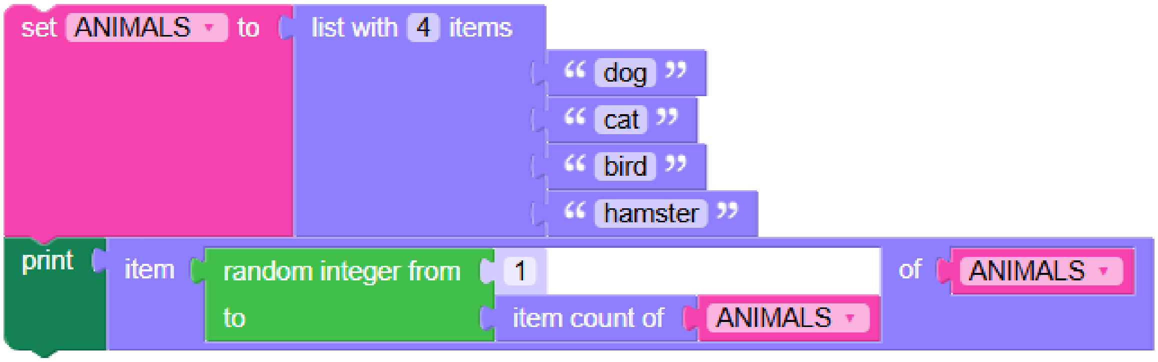
```
IGNORE ALERT ITEM 2 (LIST `dog` `cat` `bird` `hamster`)
```


cat

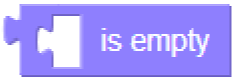
To choose a random item from a list, you could use this strategy:



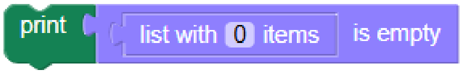
But that is a bit more complicated than it needs to be! Let’s try using a variable (much more about them later).



```
set ANIMALS to list with 4 items
    "dog"
    "cat"
    "bird"
    "hamster"
print item random integer from 1 to item count of ANIMALS of ANIMALS
```

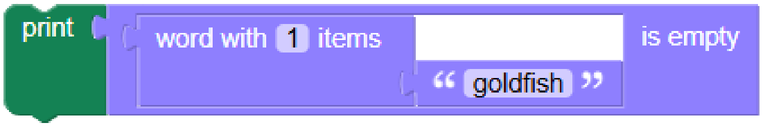


The block tells you if a word or list is empty.



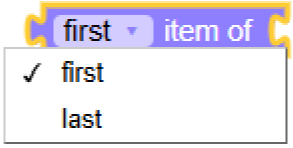
```
print list with 0 items is empty
```

this reports True

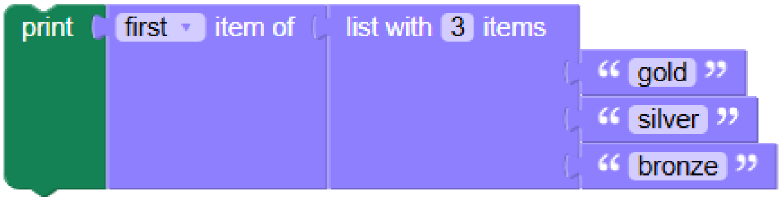


```
print word with 1 items
    "goldfish"
  is empty
```

this reports False



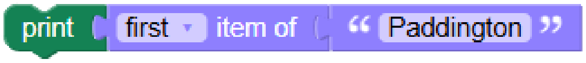
The ***first***/***last*** block tells you the first or last item in the word or list you give it. You can use it with variables to save an item before removing it from a list (as in the next block).



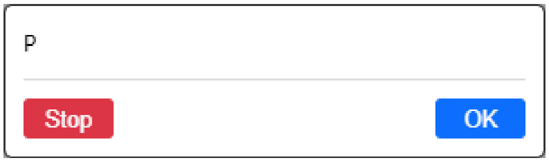
```
print first item of list with 3 items
    "gold"
    "silver"
    "bronze"
```

The first item in this list is **gold**, which is the result you get. Change ***first*** to ***last*** and you'll get **bronze**.

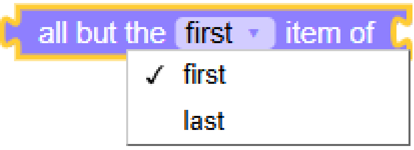
It works with words as well as with lists. Here, you get back the first letter of the word, P.



```
print first item of "Paddington"
```

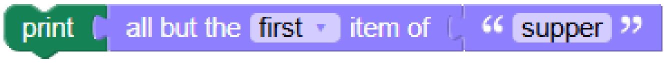


```
P
```

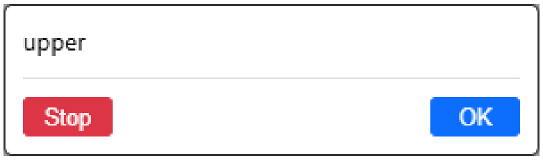


```
all but the [first ▾] item of
```

This block, which uses BUTFIRST and BUTLAST Logo commands is fun to use.

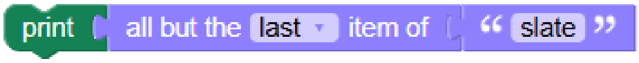


```
print all but the [first ▾] item of “ supper ”
```



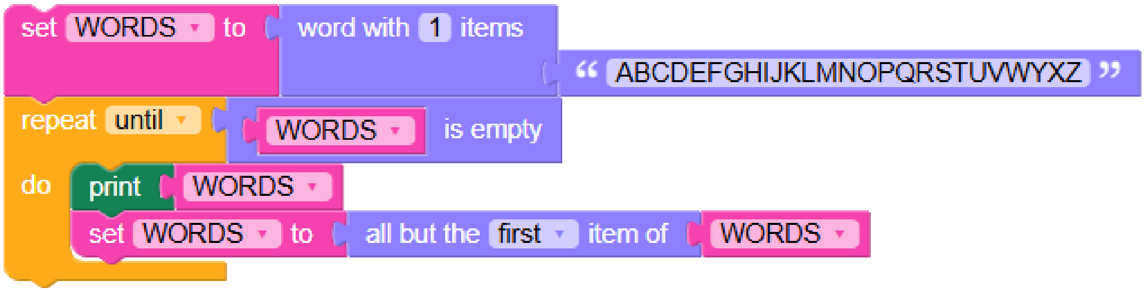
```
upper
```

Can you think of an input that produces a different word when you use all but the ***last*** item of another word? Here's one. Think of a different one and try it.



```
print all but the [last ▾] item of “ slate ”
```

Here is a more complicated use of this block. It removes a letter from the beginning of the alphabet one at a time, repeatedly, until there are no more letters left. It uses variables, which you will meet later. See if you can try it now!



```
set [WORDS ▾] to word with [1] items
                    “ ABCDEFGHIJKLMNOPQRSTUVWYXZ ”
repeat [until ▾] [WORDS ▾] is empty
do  print [WORDS ▾]
    set [WORDS ▾] to all but the [first ▾] item of [WORDS ▾]
```