



# List Constructors

These are three very common list constructors in the **Variables** menu. Each returns a new list and doesn't modify the input arguments. This is meant to supplement the "List Constructors" CS Illustrated handout.

## ADJOIN

- Available via **ToolSprite** sprite or **tools.ypr** project, and takes exactly two arguments
- In BYOB it is shown as "adjoin **ITEM** to **LIST**", i.e., 
- Reports a new list, the result of attaching the **ITEM** to the front of **LIST**, which is shown on the left if we're writing BYOB code on paper (as below), and in BYOB visually as the new first element in the top (1) slot.
- There is a similar block "adjoin to **LIST** this item **NEW** on the right", i.e.,

 that attaches **NEW** to the end (right) of **LIST** if we're writing BYOB code on paper, and in BYOB is shown visually in the last slot on the bottom.

	ITEM	LIST	
<code>adjoin(</code>	<code>x,</code>	<code>z</code>	<code>) ==&gt; ERROR, 2nd argument to adjoin has to be a list</code>
<code>adjoin(</code>	<code>()</code>	<code>z</code>	<code>) ==&gt; ERROR, 2nd argument to adjoin has to be a list</code>
<code>adjoin(</code>	<code>(r g b)</code>	<code>z</code>	<code>) ==&gt; ERROR, 2nd argument to adjoin has to be a list</code>
<code>adjoin(</code>	<code>x</code>	<code>()</code>	<code>) ==&gt; (x)</code>
<code>adjoin(</code>	<code>()</code>	<code>()</code>	<code>) ==&gt; (() ;; a list w/1 element in it, an empty list</code>
<code>adjoin(</code>	<code>(r g b)</code>	<code>()</code>	<code>) ==&gt; ((r g b)) ;; a list of the 3-element list (r g b)</code>
<code>adjoin(</code>	<code>x</code>	<code>(c m y k)</code>	<code>) ==&gt; (x c m y k)</code>
<code>adjoin(</code>	<code>()</code>	<code>(c m y k)</code>	<code>) ==&gt; (() c m y k)</code>
<code>adjoin(</code>	<code>(r g b)</code>	<code>(c m y k)</code>	<code>) ==&gt; ((r g b) c m y k) ;; as in CS Illustrated handout</code>

## APPEND




\* Available via **ToolSprite** sprite or **tools.ypr** project, and takes any number of arguments (even 0)

\* In BYOB it looks like  or  or  or ...

\* Reports the result of merging all input lists into a single list. If the inner lists have lists, it doesn't merge those.

	LIST1	LIST2	
<code>append(</code>	<code>x</code>	<code>z</code>	<code>) ==&gt; ERROR, all arguments to append must be lists</code>
<code>append(</code>	<code>()</code>	<code>z</code>	<code>) ==&gt; ERROR, all arguments to append must be lists</code>
<code>append(</code>	<code>(r g b)</code>	<code>z</code>	<code>) ==&gt; ERROR, all arguments to append must be lists</code>
<code>append(</code>	<code>x</code>	<code>()</code>	<code>) ==&gt; ERROR, all arguments to append must be lists</code>
<code>append(</code>	<code>()</code>	<code>()</code>	<code>) ==&gt; ()</code>
<code>append(</code>	<code>(r g b)</code>	<code>()</code>	<code>) ==&gt; (r g b)</code>
<code>append(</code>	<code>x</code>	<code>(c m y k)</code>	<code>) ==&gt; ERROR, all arguments to append must be lists</code>
<code>append(</code>	<code>()</code>	<code>(c m y k)</code>	<code>) ==&gt; (c m y k)</code>
<code>append(</code>	<code>(r g b)</code>	<code>(c m y k)</code>	<code>) ==&gt; (r g b c m y k) ;; as in CS Illustrated handout</code>

## LIST

- Built-in to BYOB, and takes any number of arguments (even 0)
- In BYOB it looks like  or  or  or ...
- Reports the result of wrapping all the input elements in a list, in the same order they came in.

	ELEMENT1	ELEMENT2	
<code>list(</code>	<code>x</code>	<code>z</code>	<code>) ==&gt; (x z)</code>
<code>list(</code>	<code>()</code>	<code>z</code>	<code>) ==&gt; (() z)</code>
<code>list(</code>	<code>(r g b)</code>	<code>z</code>	<code>) ==&gt; ((r g b) z)</code>
<code>list(</code>	<code>x</code>	<code>()</code>	<code>) ==&gt; (x ())</code>
<code>list(</code>	<code>()</code>	<code>()</code>	<code>) ==&gt; (() ())</code>
<code>list(</code>	<code>(r g b)</code>	<code>()</code>	<code>) ==&gt; ((r g b) ())</code>
<code>list(</code>	<code>x</code>	<code>(c m y k)</code>	<code>) ==&gt; (x (c m y k))</code>
<code>list(</code>	<code>()</code>	<code>(c m y k)</code>	<code>) ==&gt; (() (c m y k))</code>
<code>list(</code>	<code>(r g b)</code>	<code>(c m y k)</code>	<code>) ==&gt; ((r g b) (c m y k)) ;; as in CS Illustrated handout</code>