

# Writing *Snap!* code on paper

You will be asked to write *Snap!* code on this exam, so we've developed a technique for writing it on paper. There are a few key things to notice:

- We often write variables in **UPPERCASE**.
- We change spaces between words in block names to dashes (this makes it much easier to read).
- We use indentation just as *Snap!* does, to help us understand what is "inside" the **if**, **else**, and other Control structures. E.g., here's how you could write the **DrawSquare** and **n!** blocks:

The image shows two Snap! code blocks side-by-side. The left block is titled "Draw-Square (LENGTH)" and contains a "repeat (4)" block with three sub-blocks: "DrawSquare length", "move length steps", and "turn 90 degrees". The right block is titled "(N)!" and contains an "n!" block, an "if n = 0" block with a "report 1" block inside, and a "report n \* (n - 1)!" block.

- When you want to write a list of things, write them with an open parenthesis, then the first item, second item, etc (separated by spaces) and when you're done, put a closed parenthesis. If any of your items are a sentence, you have to put quotes around the sentence. So, for example, the following list of three things would be written as the equivalent 3-element-list:

A Snap! list block titled "things" containing three items: "1 life", "2 liberty", and "3 pursuit of happiness". The list has a length of 3.

- `(life liberty "pursuit of happiness")`.

- Similarly, a nested list just shows up as a nested set of parenthesis. So the following would be written as

- `((Love 5) (Hate 4) (The 10))`.

- If you want to pass in a function as argument, you know the function must be surrounded by a grey-border. Here are three new conventions:

- The grey border is written as *square brackets*: `[ ]`
- Blanks are written as parenthesis with underscore `_` in the middle, but common blocks that are passed in to HOFs can be simplified by just their name (and not the parens and underscores)
- Return values are written as `→ value`

- So the following would be written as:

- `Map[ (_)*(_) ]Reduce[ (_)+(_) ]over( (1 20 3 10) ) → 510`

- or, in the more simplified (and preferred) format:

- `Map[ * ]Reduce[ + ]over( (1 20 3 10) ) → 510`

A Snap! list block titled "nested" containing three nested lists. The first nested list has items "1 Love" and "1 5" with a length of 2. The second nested list has items "1 Hate" and "2 4" with a length of 2. The third nested list has items "1 The" and "2 10" with a length of 2. The outer list has a length of 3.

A Snap! code block representing the simplified format: `Map [ * ] Reduce [ + ] over list [ 1 20 3 10 ]`. The block is orange with green and grey sub-blocks.

510