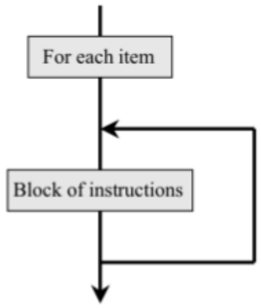
Model 1 `for` Each Value

A `for` loop executes the same block of code "for each item in a sequence". Create a new file named `loops.py`, and enter the following code:

```
print("hello")
for x in [2, 7, 1]:
    print("the number is", x)
print("goodbye")
```



Questions (15 min)

Start time:

1. Run the `loops.py` program. How many times does the indented line of code execute under the `for` loop?

2. How many times does the line of code NOT indented execute after the `for` loop?

3. Identify the value of `x` each time the indented line of code is executed.

a) 1st time:

b) 2nd time:

c) 3rd time:

4. Modify the list `[2, 7, 1]` in the following ways, and rerun the program each time. Indicate how many times the `for` loop executes.

a) non-consecutive numbers: `[5, -7, 0]`

b) numbers decreasing in value: `[3, 2, 1, 0]`

c) all have the same value: `[4, 4]`

d) single value in a list: `[8]`

5. In general, what determines the number of times that the loop repeats?

6. What determines the value of the variable `x`? Explain your answer in terms of what is assigned (`x = ...`) each time the loop runs.

7. Modify the program as follows:

a) Write a statement that assigns `[0, 1, 2, 3, 4]` to the variable `numbers`.

b) Rewrite the `for x ...` statement to use the variable `numbers` instead.

c) Does the assignment need to come before or after the `for` statement?

8. Add the following code at the end of your program:

```
for c in "Hi!":
    print(c)
```

a) What is the output of this `for` statement?

b) What determined how many times `print(c)` was called?

c) Explain what a `for` statement does with strings.

9. (Optional) What other types, besides lists and strings, can a `for` loop handle? Experiment by adding examples to your `loops.py` program. Summarize here what works and what doesn't.