CLASS 9: FLOWCHARTS – FOR LOOPS

ENGR 102 – Introduction to Engineering

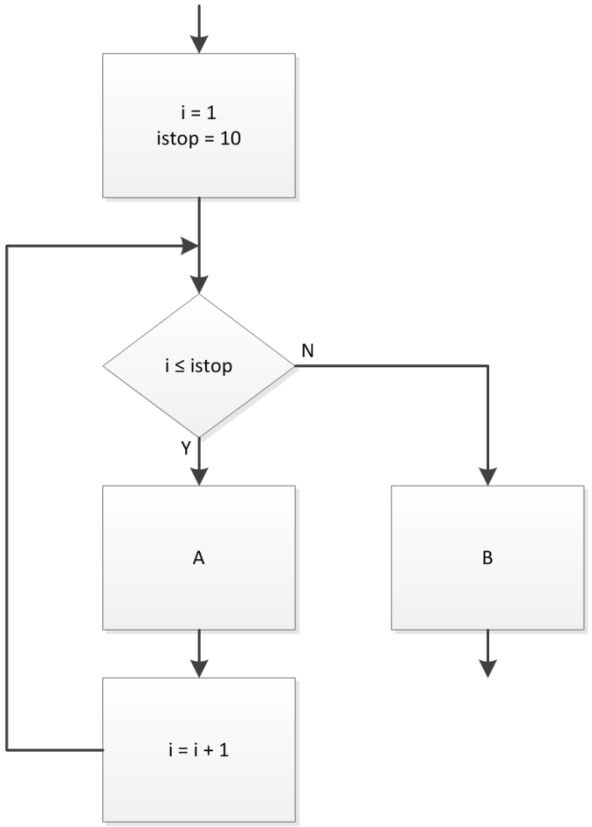
for Loop

for Loop

- We've seen that the number of while loop iterations is not known ahead of time
  - May depend on inputs, for example
- Sometimes we want a loop to execute an exact, specified number of times
- A ***for loop***
  - Utilize a ***loop counter***
  - Increment (or decrement) the counter on each iteration
  - Loop until the counter reaches a certain value
- Can be thought of as a while loop with the addition of a loop counter
  - But, a very distinct entity when implemented in code

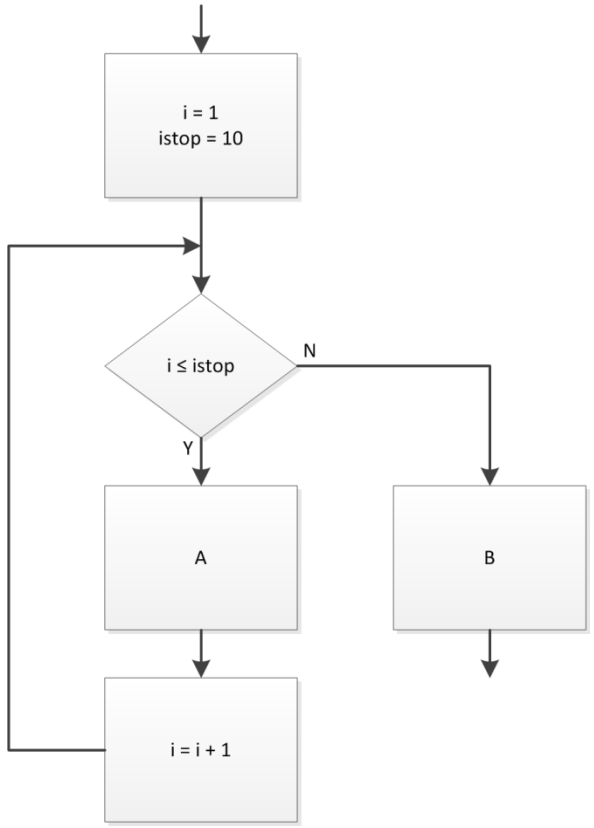
for Loop

- Initialize the loop counter
  - i, j, k are common, but name does not matter
- Set the range for i
  - Not necessary to define variable istop
- Execute loop instructions, A
- Increment loop counter, i
- Repeat until loop counter reaches its stopping value
- Continue on to B



for Loop

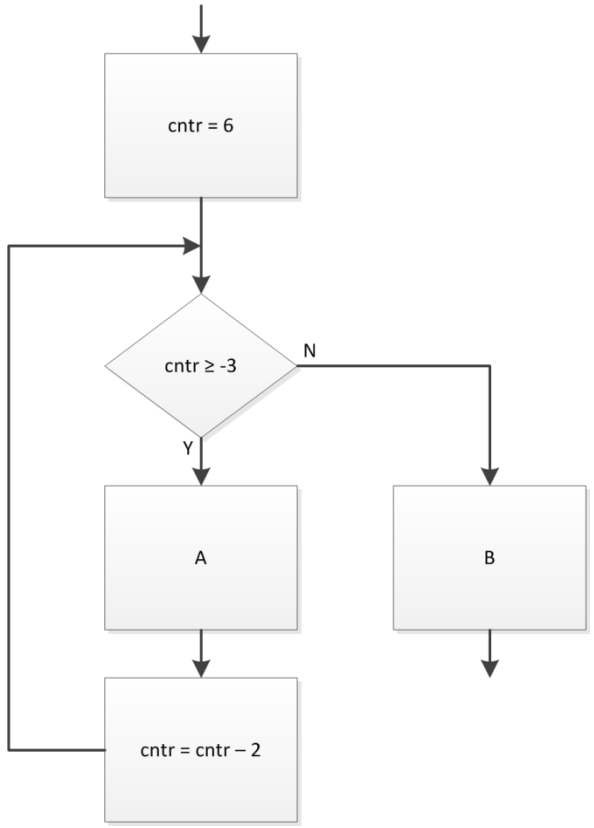
- for loops are ***counted loops***
- Number of loop iterations is known and is constant
  - Here loop executes 10 times
- Stopping value not necessarily hard-coded
  - Could depend on an input or vector size, etc.



for Loop

- Loop counter may start at value other than 1
- Increment size may be a value other than 1
- Loop counter may count backwards

Iteration	cntr	Process
1	6	A
2	4	A
3	2	A
4	0	A
5	-2	A
6	-4	B

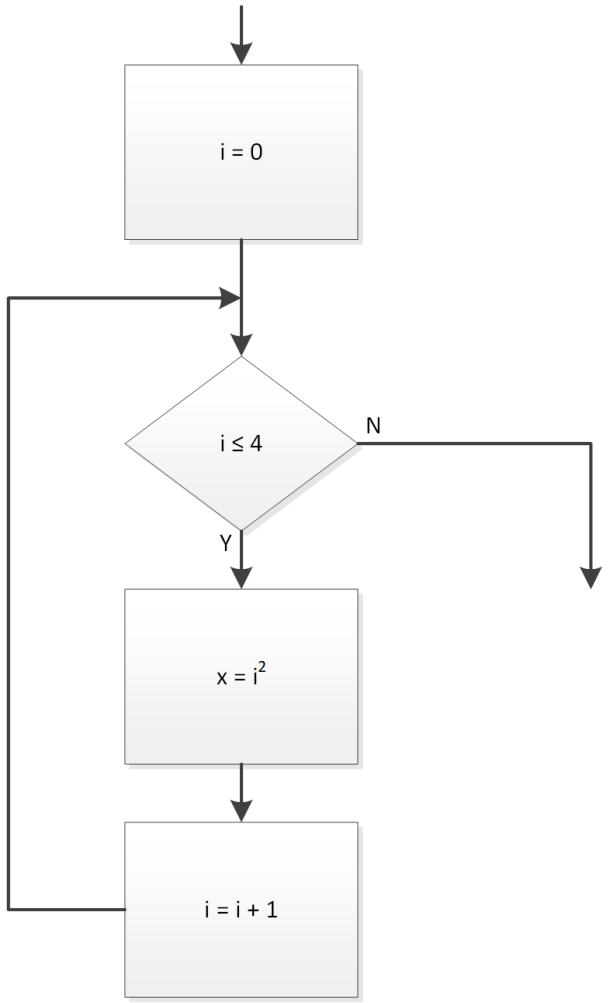


for Loop – Example 1

- Here, the loop counter, i, is used to update a variable, x, on each iteration

Iteration	i	x
1	0	0
2	1	1
3	2	4
4	3	9
5	4	16

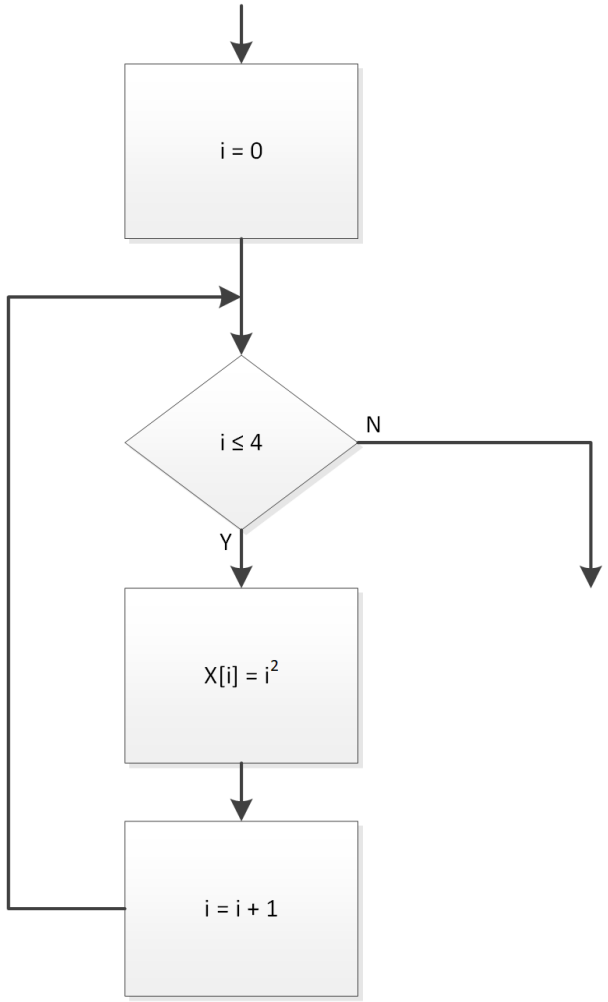
- When loop terminates, and flow proceeds to the next process step, x = 16
  - A scalar
  - No record of previous values of x



for Loop – Example 2

- Now, modify the loop process to store values of x as a ***vector***
  - Use loop counter to index the vector

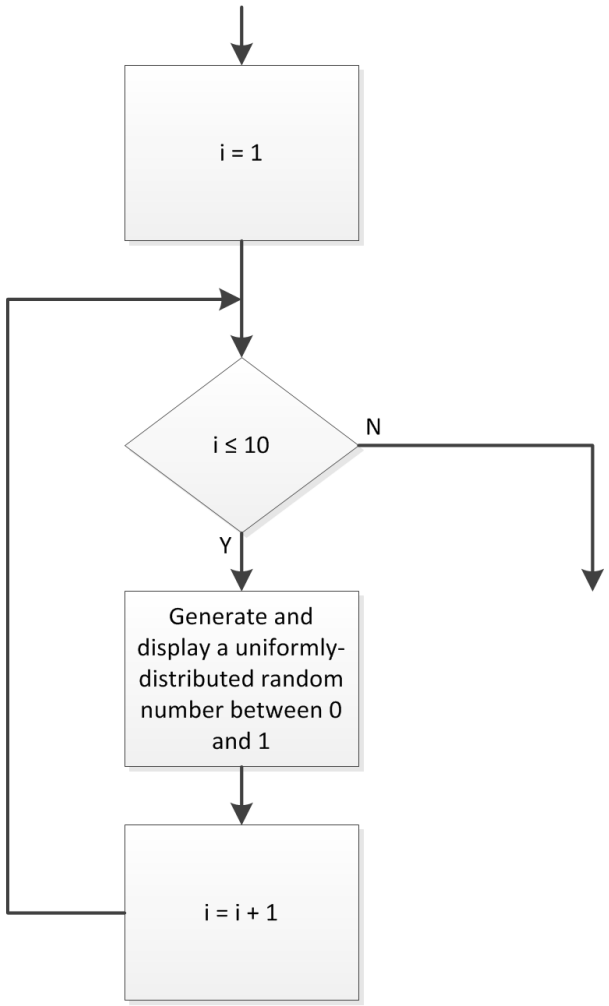
i	X[i]	x
0	0	[0]
1	1	[0, 1]
2	4	[0, 1, 4]
3	9	[0, 1, 4, 9]
4	16	[0, 1, 4, 9, 16]



- When loop terminates, x = [0, 1, 4, 9, 16]
  - A ***vector***
  - x grows with each iteration

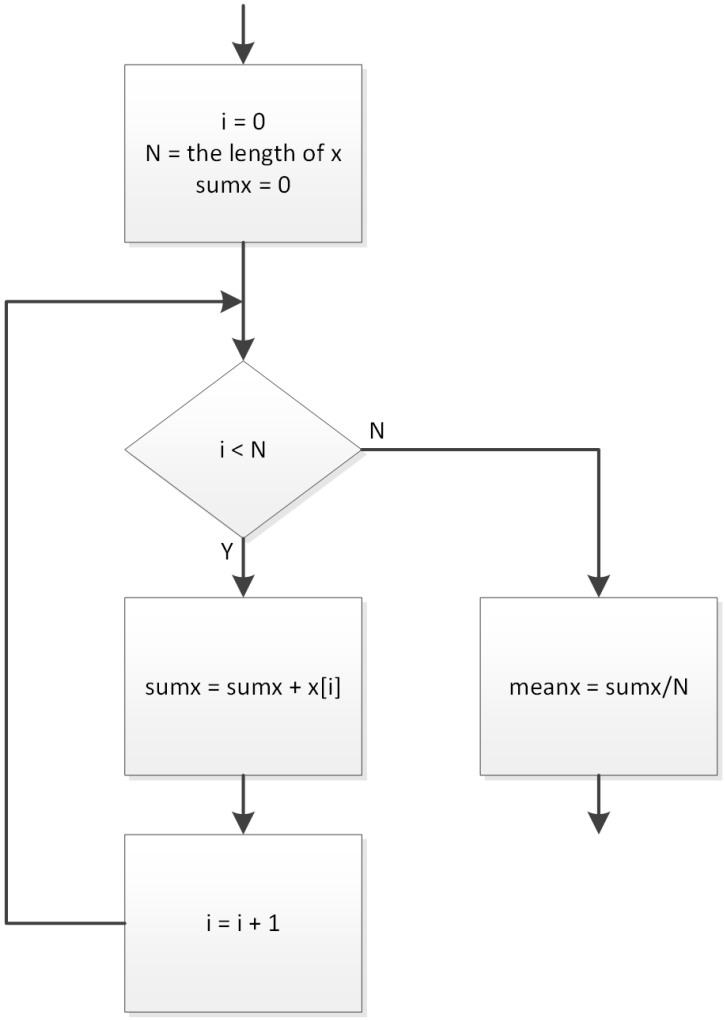
for Loop – Example 3

- The loop counter does not need to be used within the loop
  - Used as a counter *only*
- Here, a random number is generated and displayed each of the 10 times through the loop
  - Counter, i, has nothing to do with the values of the random numbers displayed



for Loop – Example 4

- Have a vector of values, x
- Find the ***mean*** of those values
  - Sum all values in x
    - A for loop
    - # of iterations equal to the length of x
    - Loop counter indexes x
  - Divide the sum by the number of elements in x
    - After exiting the loop



Nested Loops

Nested Loops

- A loop repeats some process some number of times
  - The repeated process can, itself, be a loop
  - A ***nested loop***
- Can have nested *for loops* or *while loops*
  - Can nest for loops within while loops and vice versa
- One application of a ***nested for loop*** is to step through every element in a matrix
  - Loop counter variables used as matrix indices
  - Outer loop steps through rows (or columns)
  - Inner loop steps through columns (or rows)

Nested for Loop – Example

- Recall how we index the elements within a matrix:
  - A_{ij} is the element on the i^{th} row and j^{th} column of the matrix A
  - Using Python syntax: A[i,j]
- Consider a 3×2 matrix

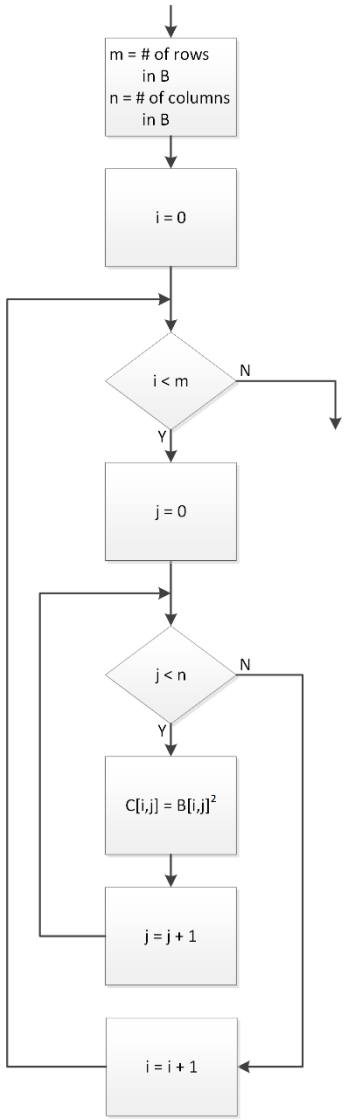
$$B = \begin{bmatrix} -2 & 1 \\ 0 & 8 \\ 7 & -3 \end{bmatrix}$$

- To access every element in B:
  - start on the first row and increment through all columns
  - Increment to the second row and increment through all columns
  - Continue through all rows
  - Two nested for loops

Nested for Loop – Example

$$B = \begin{bmatrix} -2 & 1 \\ 0 & 8 \\ 7 & -3 \end{bmatrix}$$

- Generate a matrix C whose entries are the squares of all of the elements in B
  - ***Nested for loop***
  - Outer loop steps through rows
    - Counter is row index
  - Inner loop steps through columns
    - Counter is column index



Pseudocode & Top-Down Design

Pseudocode

- Flowcharts provide a useful tool for designing algorithms
  - Allow for describing algorithmic structure
  - Ultimately used for generation of code
  - Details neglected in favor of concise structural and functional description
- ***Pseudocode*** provides a similar tool
  - One step closer to actual code
  - ***Textual*** description of an algorithm
  - ***Natural language*** mixed with language-specific syntax

Pseudocode – Example

- Consider an algorithm for determining the maximum of a vector of values
- Pseudocode might look like:

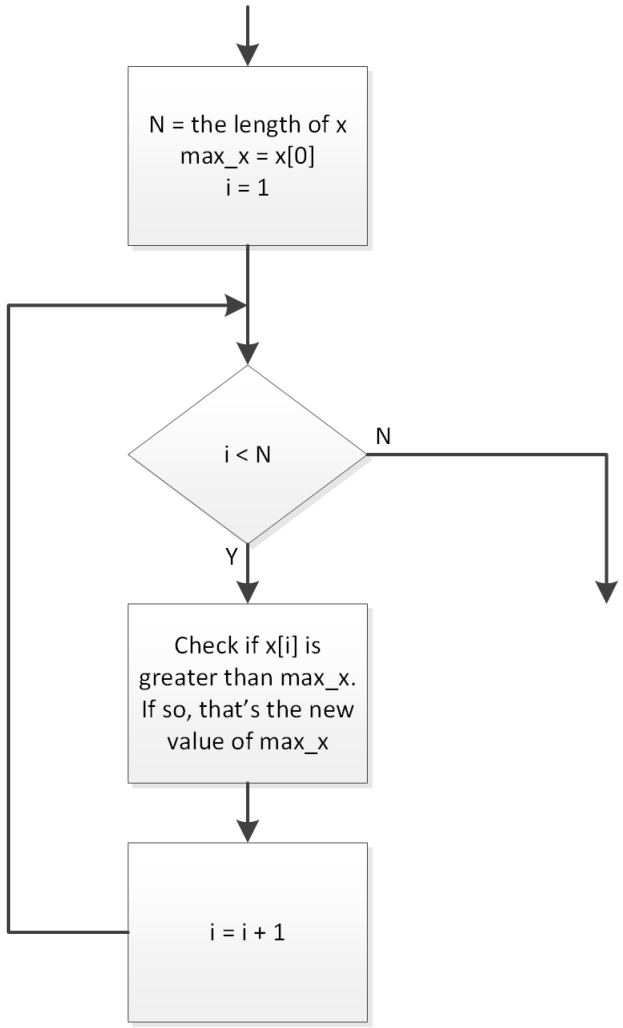
```
N = length of x

max_x = x[0]

for i = 1 through N-1

  if x[i] is greater than current
  max_x, then set max_x = x[i]
```

- We'll learn the Python-specific *for*-loop syntax in the following section of notes



Top-Down Design

- Flowcharts and pseudocode are useful tools for ***top-down design***
  - A good approach to any complex engineering design (and writing, as well)
  - First, define the overall system or algorithm at the top level (perhaps as a flowchart)
  - Then, fill in the details of individual functional blocks
- Top-level flowchart identifies individual functional blocks and shows how each fits into the algorithm
  - Each functional block may comprise its own flow chart or even multiple levels of flow charts
  - ***Hierarchical design***

Top-Down Design - Example

- Let's say you have deflection data from FEM analysis of a truss design
  - Data stored in text files
    - Deflection vs. location along truss
  - Parametric study
    - Three different component thicknesses
    - Two different materials
    - Six data sets
- Read in the data, calculate the max deflection and plot the deflection vs. position

Top-Down Design - Example