Code Academy

Chapter 4: Loops (4.7)

PART3: NESTED LOOPS

Code Academy

Chapter 3-Part3:

- **Goals**
  - To understand nested loops

- **Contents:**
- Nested loops

Code Academy

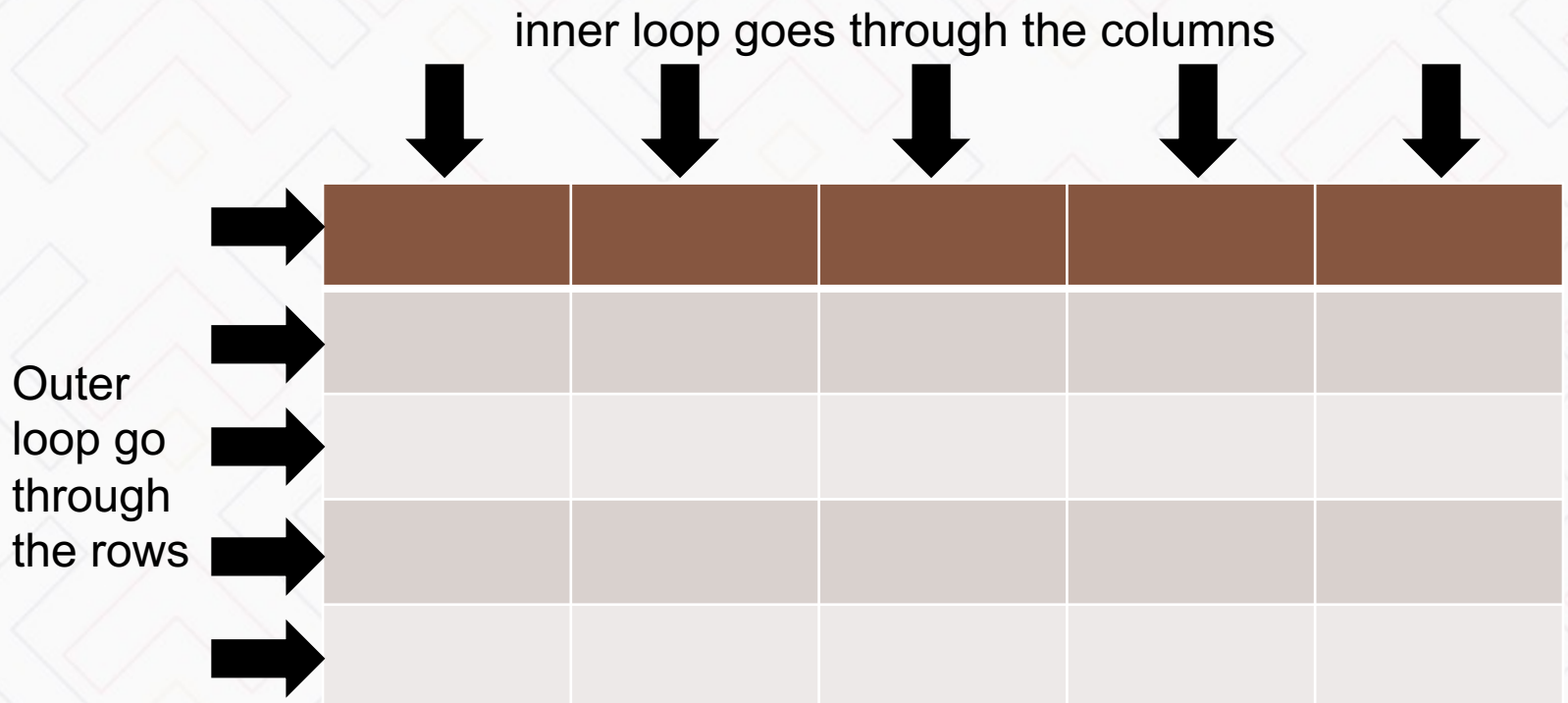
Nested Loops: Loops Inside of Loops

- In Chapter Three we learned how to nest **if** statements to allow us to make complex decisions:
  - Remember that to nest the **if** statements we need to indent the code block
- Complex problems sometimes require a nested loop, one loop nested inside another loop:
  - The nested loop will be indented inside the code block of the first loop

Code Academy

Nested Loops: Loops Inside of Loops

- A good example of using nested loops is when you are processing cells in a table:
  - The outer loop iterates over all of the rows in the table
  - The inner loop processes the columns in the current row



Code Academy

Our Example Problem Statement

- Print a Table Header that contains x^1, x^2, x^3, and x^4
- Print a Table with four columns and ten rows that contain the powers of x^1, x^2, x^3, and x^4 for x = 1 to 10

x^1	x^2	x^3	x^4
1	1	1	1
2	4	8	16
3	9	27	81
…	…	…	…
10	100	1000	10000

Applying Nested Loops

- How would you print a table with rows and columns?
  - Print top line (i.e. header)
    - Use a for loop
  - Print table body…
    - How many rows are in the table?
    - How many columns in the table?
  - Loop per row
    - Loop per column
- In our example there are:
  - Ten rows in the table
  - Four columns in the table

x^1	x^2	x^3	x^4
1	1	1	1
2	4	8	16
3	9	27	81
…	…	…	…
10	100	1000	10000

Pseudocode to Print the Table

Print the table header

```
for x from 1 to 10
  print a new table row
  print a new line
```

- How do we print a table row?

```
For n from 1 to 4
  print x^n
```

- We have to place this loop inside the preceding loop
  - The inner loop is *"nested"* inside the outer loop

Pseudocode to Print the Table

Print the table header:

```
for x from 1 to 10
  for n from 1 to 4
    print Xⁿ
  print a new line
```

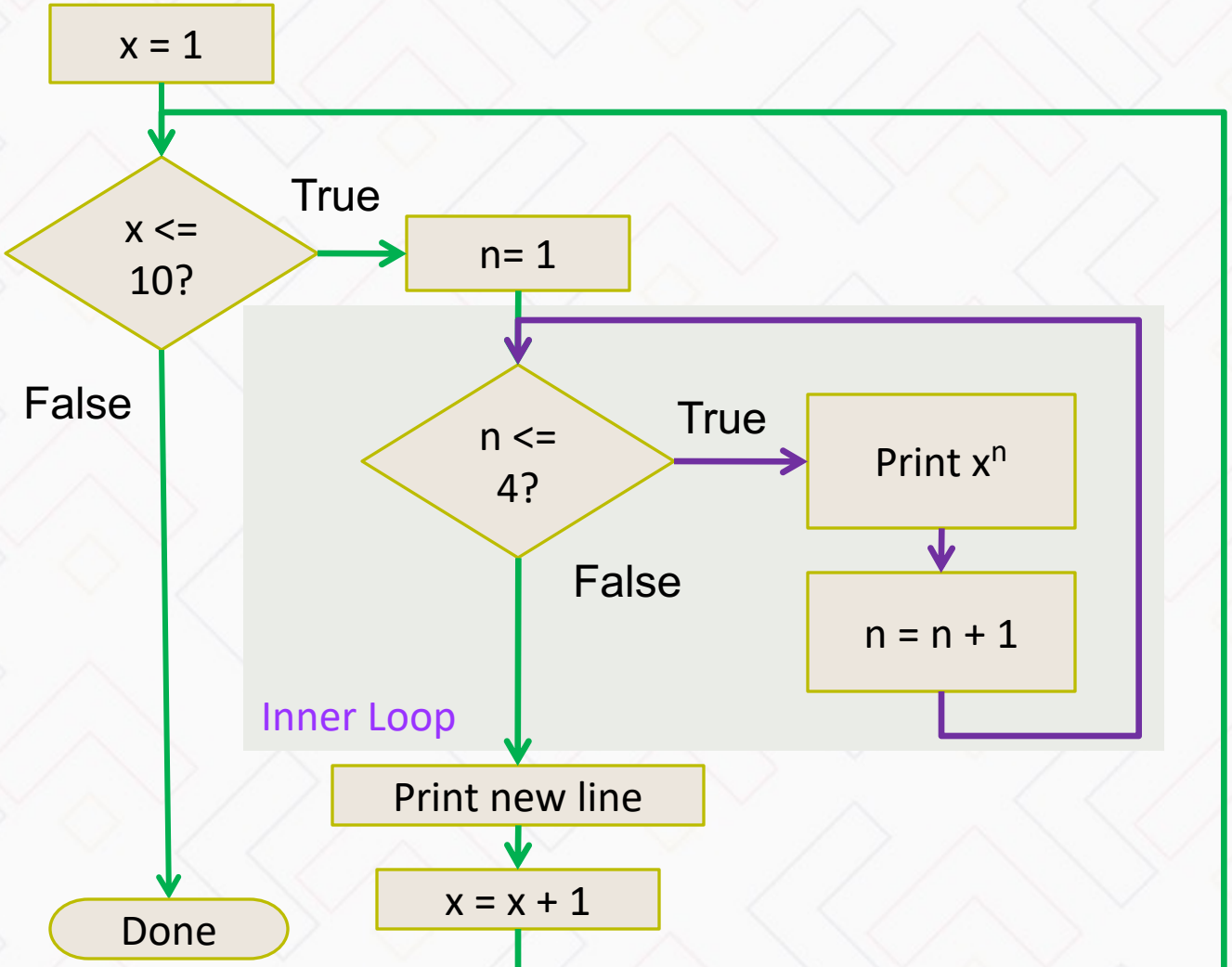
x ↓

n →

x^1	x^2	x^3	x^4
1	1	1	1
2	4	8	16
3	9	27	81
…	…	…	…
10	100	1000	10000

Code Academy

Flowchart of a Nested Loop



Powertable.py

Code Academy

```
1   #
2   #  This program prints a table of powers of x.
3   #
4
5   # Initialize constant variables for the max ranges.
6   NMAX = 4
7   XMAX = 10
8
9   # Print table header.
10  #
11
12  for n in range(1, NMAX + 1) :
13     print("%10d" % n, end="")
14
15  print()
16  for n in range(1, NMAX + 1) :
17     print("%10s" % "x ", end="")
18
19  print("\n", "    ", "-" * 35)
20
21  # Print table body.
22  #
23
24  for x in range(1, XMAX + 1) :
25     # Print the x row in the table.
26     for n in range(1, NMAX + 1) :
27        print("%10.0f" % x ** n, end="")
28
29     print()
30
```

The **end=“”** suppresses the new line, so the numbers are all printed on the same line

Body of outer loop, x = 1 ➔ 10

Body of inner loop, n = 1 ➔ 4

The Results

Code Academy

```
[evaluate Powertable header.py]
          1           2           3           4
          x           x           x           x
  ------------------------------------------------
          1           1           1           1
          2           4           8          16
          3           9          27          81
          4          16          64         256
          5          25         125         625
          6          36         216        1296
          7          49         343        2401
          8          64         512        4096
          9          81         729        6561
         10         100        1000       10000
```

Code Academy

First Exercise

- Revise the program **powertable.py**, and make the following changes:
  - Change the value of NMAX to 6 and run the program
  - What changes in the table?
  - Change the value of NMAX back to 4
  - Change the value of XMAX to 4
  - What changes in the table?

Nested Loop Examples

Table 3 Nested Loop Examples

Nested Loops	Output	Explanation

`for i in range(3) :`
`  for j in range(4) :`
`  print("*", end="")`
`  print()`
	****
	****
	****
		Prints 3 rows of 4 asterisks each.

`for i in range(4) :`
`  for j in range(3) :`
`  print("*", end="")`
`  print()`
	***
	***
	***
	***
		Prints 4 rows of 3 asterisks each.

`for i in range(4) :`
`  for j in range(i + 1) :`
`  print("*", end="")`
`  print()`
	*
	**
	***
	****
		Prints 4 rows of lengths 1, 2, 3, and 4.

Hand Tracing the Loop

```
1  for i in range(4) :
2      for j in range(i + 1) :
3          print("*", end="")
4      print()
```

- i will have the values:
  - 0, 1, 2, 3 – So we will have four lines of stars
- j will have the values
  - 0 - So we will have one star
  - 0, 1 - So we will have two stars
  - 0, 1, 2 - So we will have three stars
  - 0, 1, 2, 3 - So we will have four stars

```
[evaluate nested loop example three.py]
*
**
***
****
```

Nested Loop Examples (2)

Table 3 Nested Loop Examples

`for i in range(3) :`
`  for j in range(5) :`
`  if j % 2 == 1 :`
`  print("*", end="")`
`  else :`
`  print("-", end="")`
`  print()`
	-*-*-
	-*-*-
	-*-*-
		Prints alternating dashes and asterisks.

`for i in range(3) :`
`  for j in range(5) :`
`  if i % 2 == j % 2 :`
`  print("*", end="")`
`  else :`
`  print(" ", end="")`
`  print()`
	* * *
	 * *
	* * *
		Prints a checkerboard pattern.

Code Academy

Second Problem Statement

- Print the following pattern of brackets:
  []][]][]
  []][]][]
  []][]][]
- The pattern consists of:
  - Three rows
  - Each row has four pairs of brackets
- What do we know?
  - We need two nested loops
    - The first loop (the outer loop) will print each of the three rows
    - The second loop (the inner loop) will print the four pairs of brackets

Pseudocode Code, Results

```
For i = 1 to 3
  For j = 1 to 4
    Print “[]”
  Print a new line
```

```
1  for i in range(3) :
2      for j in range(4) :
3          print("[]", end="")
4      print()
```

```
[evaluate nested loop example three.py]
[][][][]
[][][][]
[][][][]
```

Exam Averages Problem Statement

- It is common to repeatedly read and process multiple groups of values:
  - Write a program that can compute the average exam grade for multiple students.
  - Each student has the same number of exam grades
  - Prompt the user for the number of exams
  - When you finish a student prompt the user to see if there are more students to process
- What do we know?
- What do we need to compute?
- What is our algorithm / approach?

Step One: Understand the Problem

- To compute the average grade for a student, we must read and tally all of the grades for that student
  - We can use a loop to do this. ***(we have working code to do this portion)***
- We need to compute grades for multiple students
  - That implies a set of nested Loops
    - The outer loop processes each student
    - The inner loop process the student's grades

Step Two

Code Academy

- Compute the grade for one student
- Set up the variables and loop
- We know how many grades to process, so we can use a count-controlled loop

```
total score = 0
For i in range (1, number of exams + 1) :
  Read the next exam score
  Add the exam score to the total score
Compute the exam average
Print the exam average
```

Step Three

Code Academy

- Repeat the process for each student
- Since we don’t know how many students there are, we will use a while loop with a sentinel value
  - For simplicity we will use “Y” as the sentinel value

Step Four: Translate to Python

```
1   ##
2   #  This program computes the average exam grade for multiple students.
3   #
4
5   # Obtain the number of exam grades per student.
6   numExams = int(input("How many exam grades does each student have? "))
7
8   # Initialize moreGrades to a non-sentinel value.
9   moreGrades = "Y"
10
11  # Compute average exam grades until the user wants to stop.
12  while moreGrades == "Y" :
13
14      # Compute the average grade for one student.
15      print("Enter the exam grades.")
16      total = 0
17      for i in range(1, numExams + 1) :
18          score = int(input("Exam %d: " % i))   # Prompt for each exam grade.
19          total = total + score
20
21      average = total / numExams
22      print("The average is %.2f" % average)
23
24      # Prompt as to whether the user wants to enter grades for another student.
25      moreGrades = input("Enter exam grades for another student (Y/N)? ")
26      moreGrades = moreGrades.upper()
```

Program Run

```
How many exam grades does each student have? 4
Enter the exam grades.

Exam 1: 80

Exam 2: 88

Exam 3: 76

Exam 4: 90
The average is 83.50

Enter exam grades for another student (Y/N)? y
Enter the exam grades.

Exam 1: 45

Exam 2: 56

Exam 3: 67

Exam 4: 89
The average is 64.25

Enter exam grades for another student (Y/N)? n

In [3]:
```