Exam1 on Feb 23rd
Exam1 conflict form – fill by **today**

[220 / 319] Iteration

Meena Syamkumar
Andy Kuemmel
Cole Nelson

Due: P3
Grade posted: P2
Grade posted: Student
Information Survey

Readings:
Chapter 7 of Think Python
Chapter 6.1 to 6.3 of Python for Everybody

Learning Objectives

Implement an iterative algorithm using a while loop
- printing /counting
- validating user input
- performing iterative calculation
- printing grids / character art

Trace iterative algorithms and determine their output

Recognize common while loop errors:
- infinite loops (when unintentional)
- off-by-one mistakes in the loop control variable increment / decrement

Worksheet

State:

N | 4

⬭ 6

total | 0

answer | 0

Code:
1. Put 1 in the "total" box
2. If "N" equals 1, skip to step 6, otherwise continue to step 3
3. Multiply the value in "total" by the value in "N", and put the result back in "total"
4. Decrease the value in "N" by 1
5. Go to step 2
6. Copy the value in total to the answer box

Combination of conditionally skipping forward (2) with going back is (5) is called a "while loop"

Worksheet

State:

N | 4

total | 0

answer | 0

(6)

Code:
1. Put 1 in the "total" box
2. If "N" equals 1, skip to step 6, otherwise continue to step 3

    skip past loop body   *continue to loop body*

3. Multiply the value in "total" by the value in "N", and put the result back in "total"
4. Decrease the value in "N" by 1
5. Go to step 2
6. Copy the value in total to the answer box
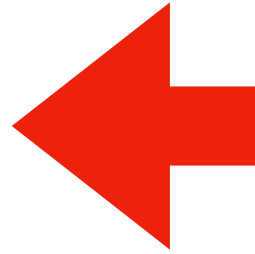
loop body

going back will be implicit in Python, and will happen right after loop body. you can identify the loop body because it will be indented
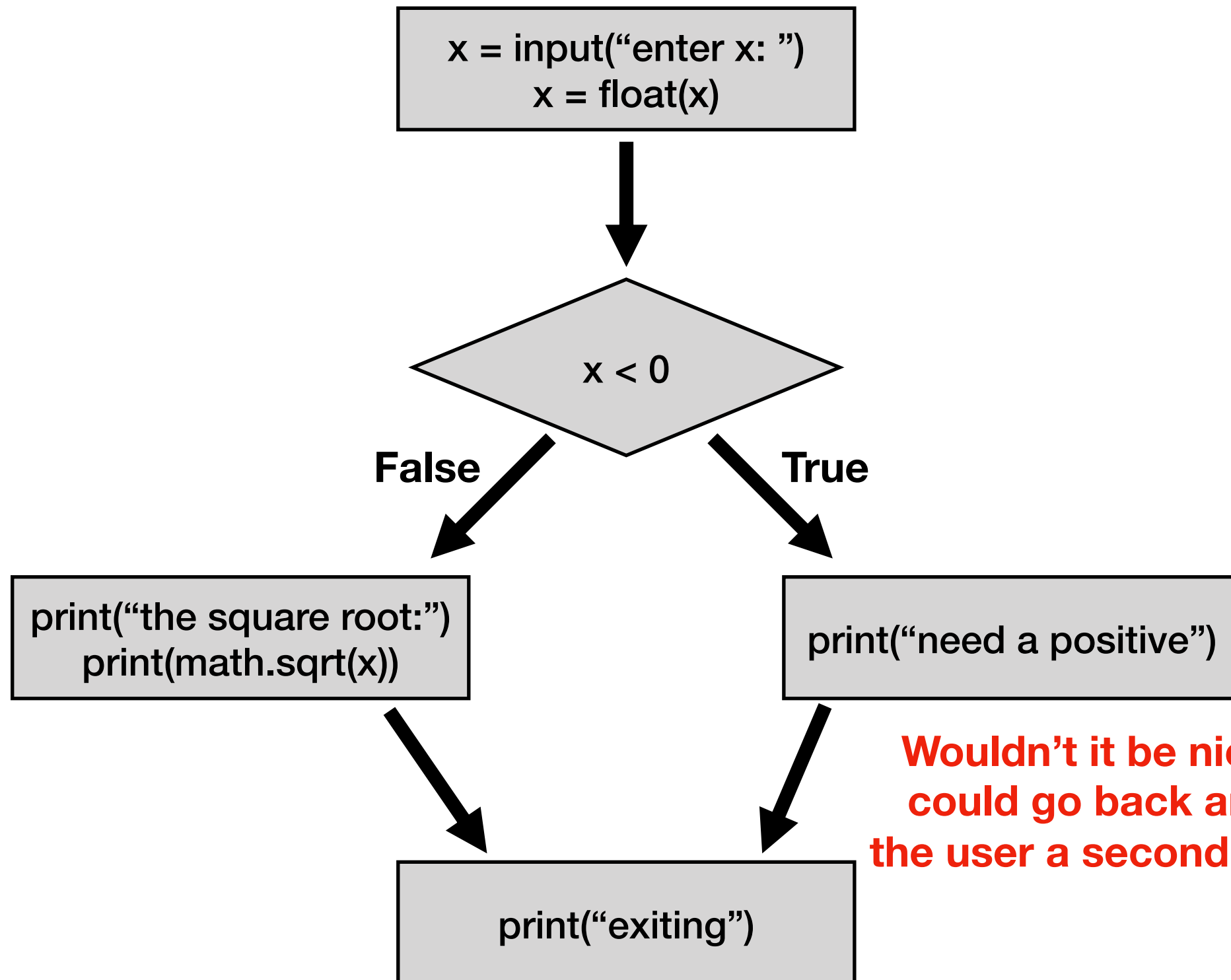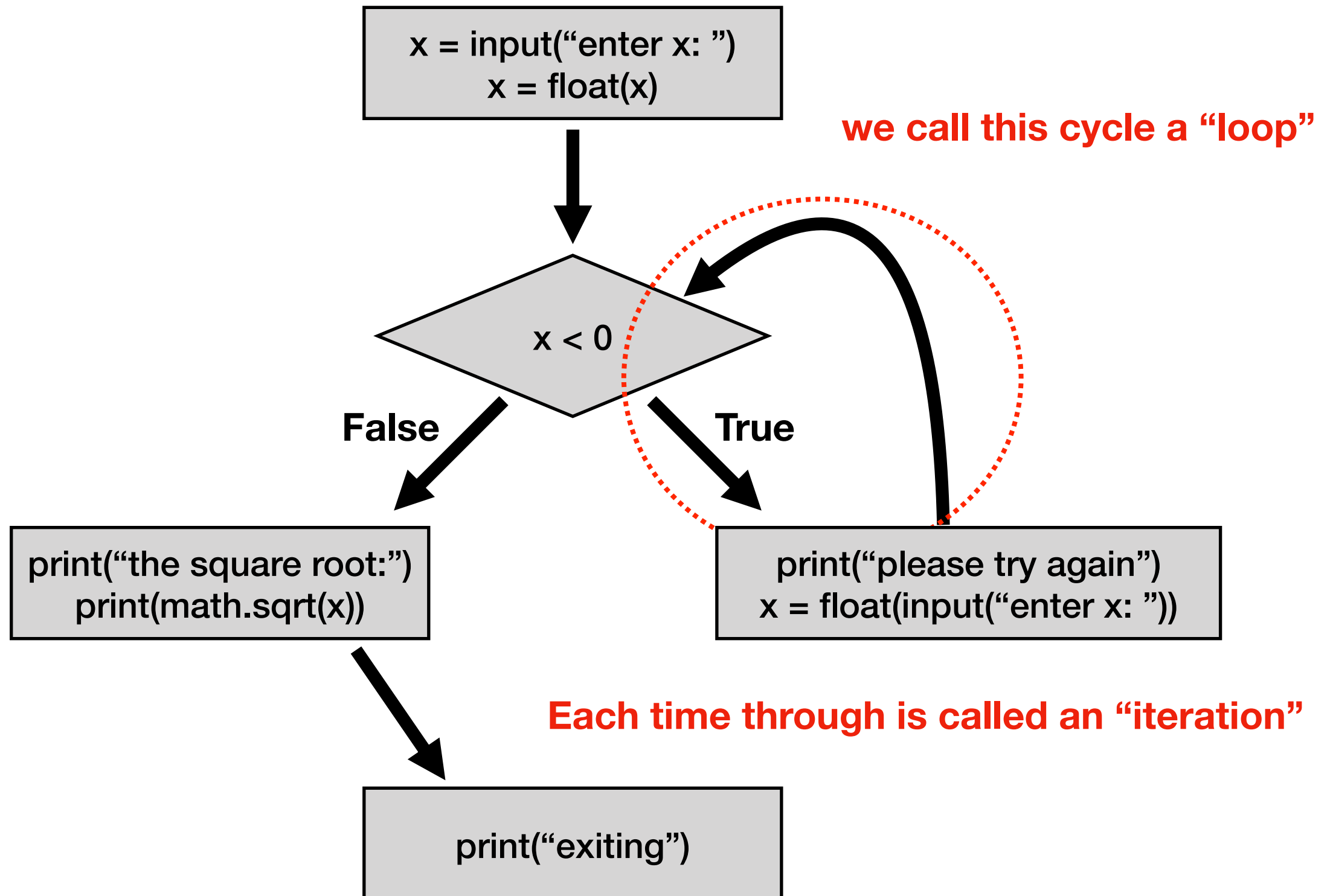
Today's Outline

Control Flow Diagrams ⬅

Basic syntax for "while"

Examples

Control Flow Diagrams: "if"

```
x = input("enter x: ")
x = float(x)
```

```
x < 0
```

False   **True**

```
print("the square root:")
print(math.sqrt(x))
```

```
print("need a positive")
```

```
print("exiting")
```

Wouldn't it be nice if we could go back and give the user a second chance?

Control Flow Diagrams: "while"

```
x = input("enter x: ")
x = float(x)
```

we call this cycle a "loop"

x < 0

False   **True**

```
print("the square root:")
print(math.sqrt(x))
```

```
print("please try again")
x = float(input("enter x: "))
```

Each time through is called an "iteration"

```
print("exiting")
```

Control Flow Diagrams: "while"

x = input("enter x: ")
x = float(x)

We keep executing the loop body **while** the condition is true, so this is called a "while" loop

loop condition

x < 0

False

True

print("the square root:")
print(math.sqrt(x))

print("please try again")
x = float(input("enter x: "))

loop body

print("exiting")

Control Flow Diagrams: "while"

```
x = input("enter x: ")
x = float(x)
```

x < 0

False

True

```
print("the square root:")
print(math.sqrt(x))
```

```
print("please try again")
x = float(input("enter x: "))
```

```
print("exiting")
```

what does this loop do? (note crossed out line)

runs forever! called an "infinite loop"

Control Flow Diagrams: "while"

x = input("enter x: ")
x = float(x)

To avoid infinite loops, make sure something will/can eventually happen in the body to change the condition

x < 0

False

True

print("the square root:")
print(math.sqrt(x))
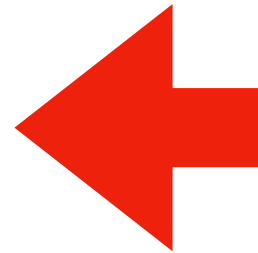
print("please try again")
x = float(input("enter x: "))

print("exiting")

Today's Outline

Control Flow Diagrams

Basic syntax for "while"   ⬅

Examples

Syntax

```
x = int(input("enter x: "))

if x < 0:
    x = int(input("please try again: "))
```

Syntax for "if"

Syntax

```
x = int(input("enter x: "))

while x < 0:
    x = int(input("please try again: "))
```

Syntax for "while loop" is just like for "if", just replace "if" with "while"

**This example gives user an arbitrary number of tries
until they get it right**
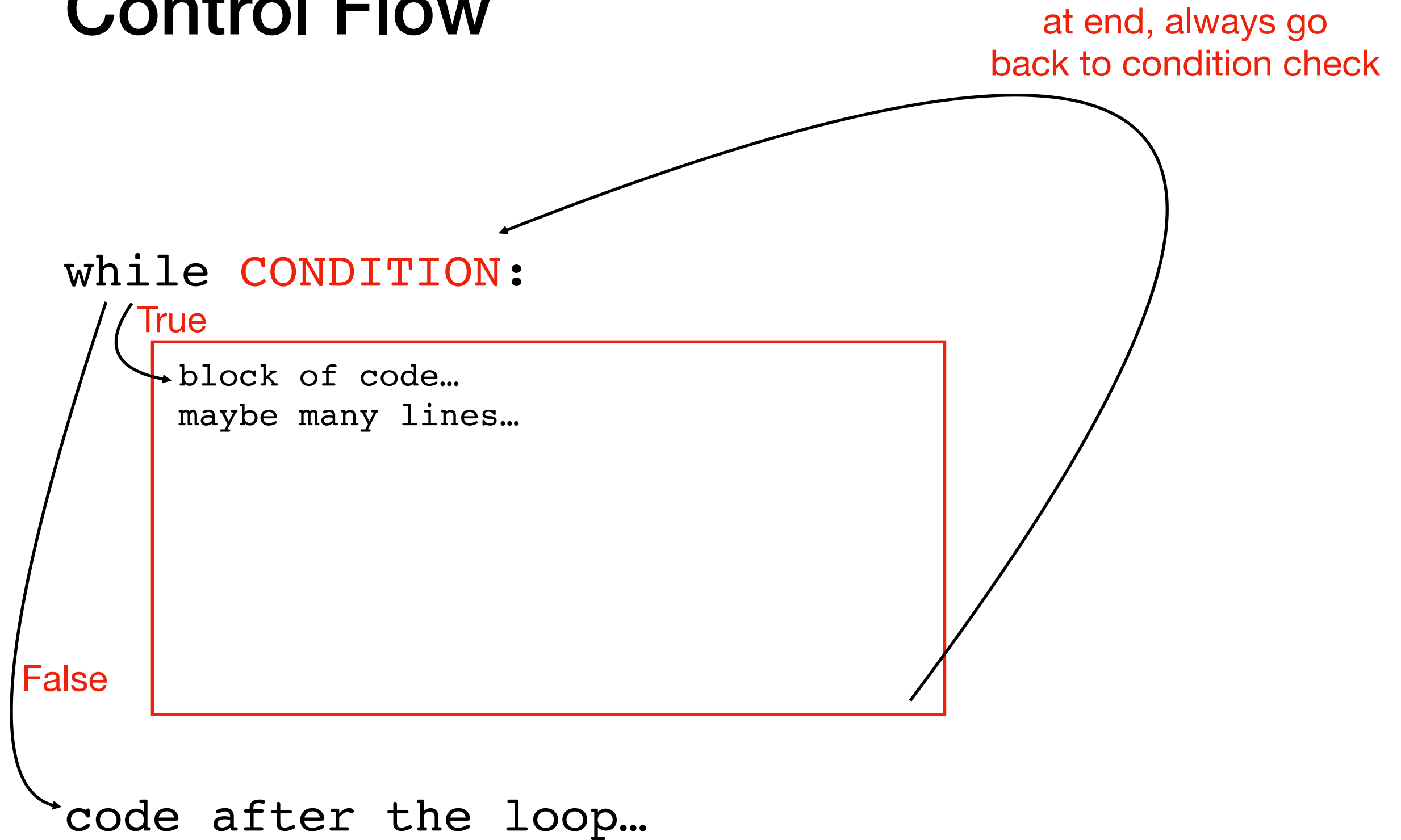
Terminology

```
x = int(input("enter x: "))

while x < 0:
    x = int(input("please try again: "))
```

loop condition

loop control variable

loop body

Control Flow

at end, always go
back to condition check

while CONDITION:
True

block of code…
maybe many lines…

False

code after the loop…

Steps to follow

Whenever you write a while loop, keep these in mind:

1. **Initialize** your loop condition variable

2. a) **Update** your loop condition variable in loop body

    b) Make **progress towards** eventually turning your loop condition to **False**

Congrats!

You now understand the 4 key **Flow of Execution** ideas,
 in the context of Python.

1. **generally, proceed forward, one step at a time**

2. sometimes go run a "mini program" somewhere else before continuing to
 the next line
• This is a function call

3. sometimes skip forward over some lines of code
• Conditional or while loop, when the condition is false

4. sometimes go back to a previous line of code
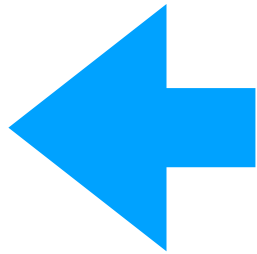• while loop. When at the end of body, always go back to condition

three primary exceptions to the general case (1)

Today's Outline

Control Flow Diagrams

Basic syntax for "while"

Examples

Example: Countdown Timer

use `time.sleep(1)`

```
How many seconds? 5
5 seconds left
4 seconds left
3 seconds left
2 seconds left
1 seconds left
DING DING DING DING DING!
```
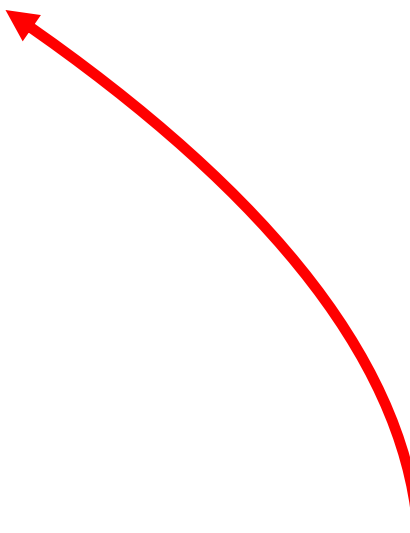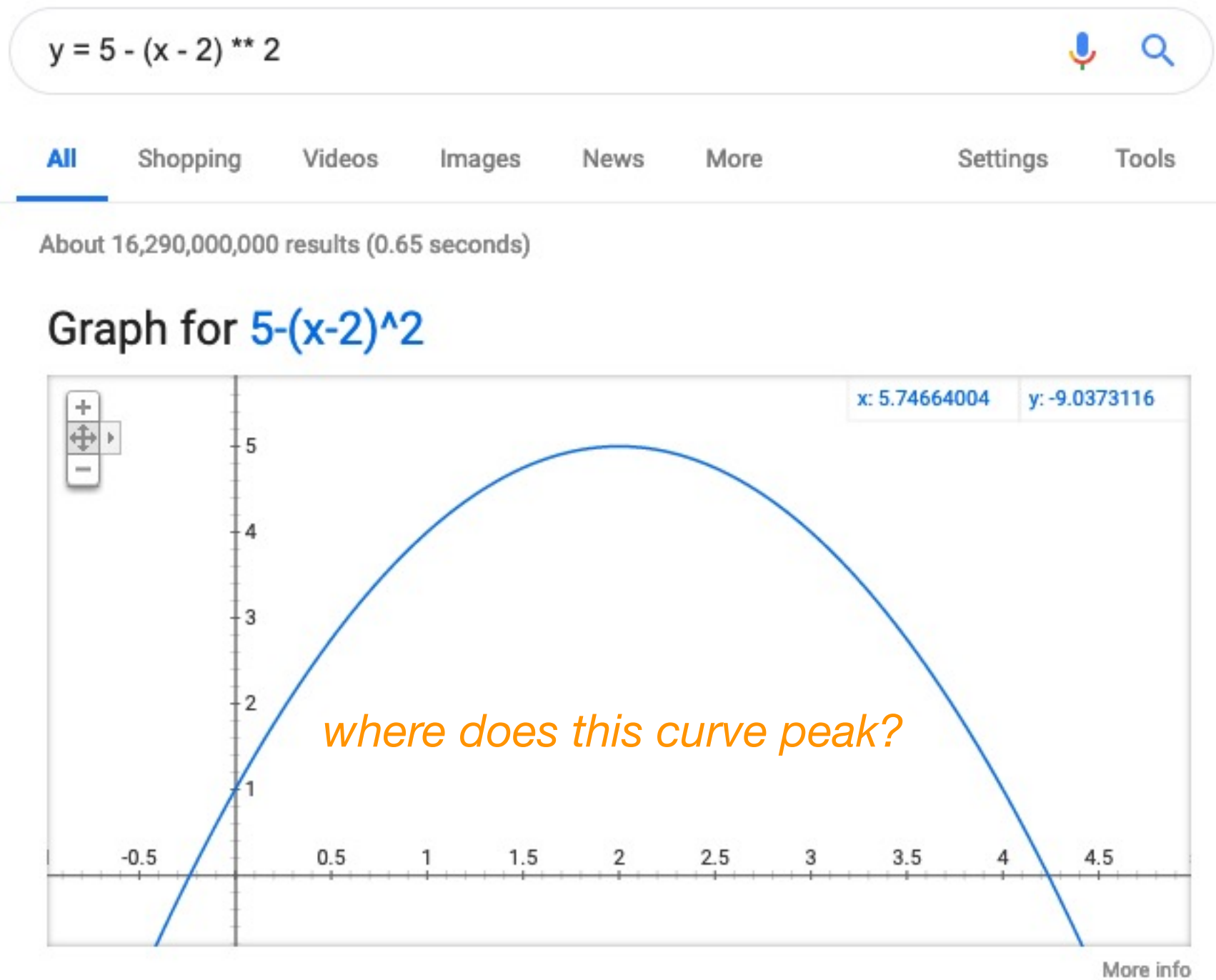
for with range

0
3
6
9
12

```
for item in range(5):
    print(item * 3)
```

using range(N) with a for loop will iterate with these values for item:
0, 1, 2, …, N-2, N-1

Example: Maximum (Finding the Peak)

y = 5 - (x - 2) ** 2

All   Shopping   Videos   Images   News   More   Settings   Tools

About 16,290,000,000 results (0.65 seconds)

Graph for $5-(x-2)^2$

where does this curve peak?

Example: Integration (Riemann Sum)

GeoGebra

Riemann Sum Calculator

Author: megan.ann.martinez

Topic: Area, Upper and Lower Sum or Riemann Sum

Left Riemann Sum = 12.16

Right Riemann Sum = 8.96

Function = $5 - (x - 2)^2$

a = 1   b = 5   n = 10
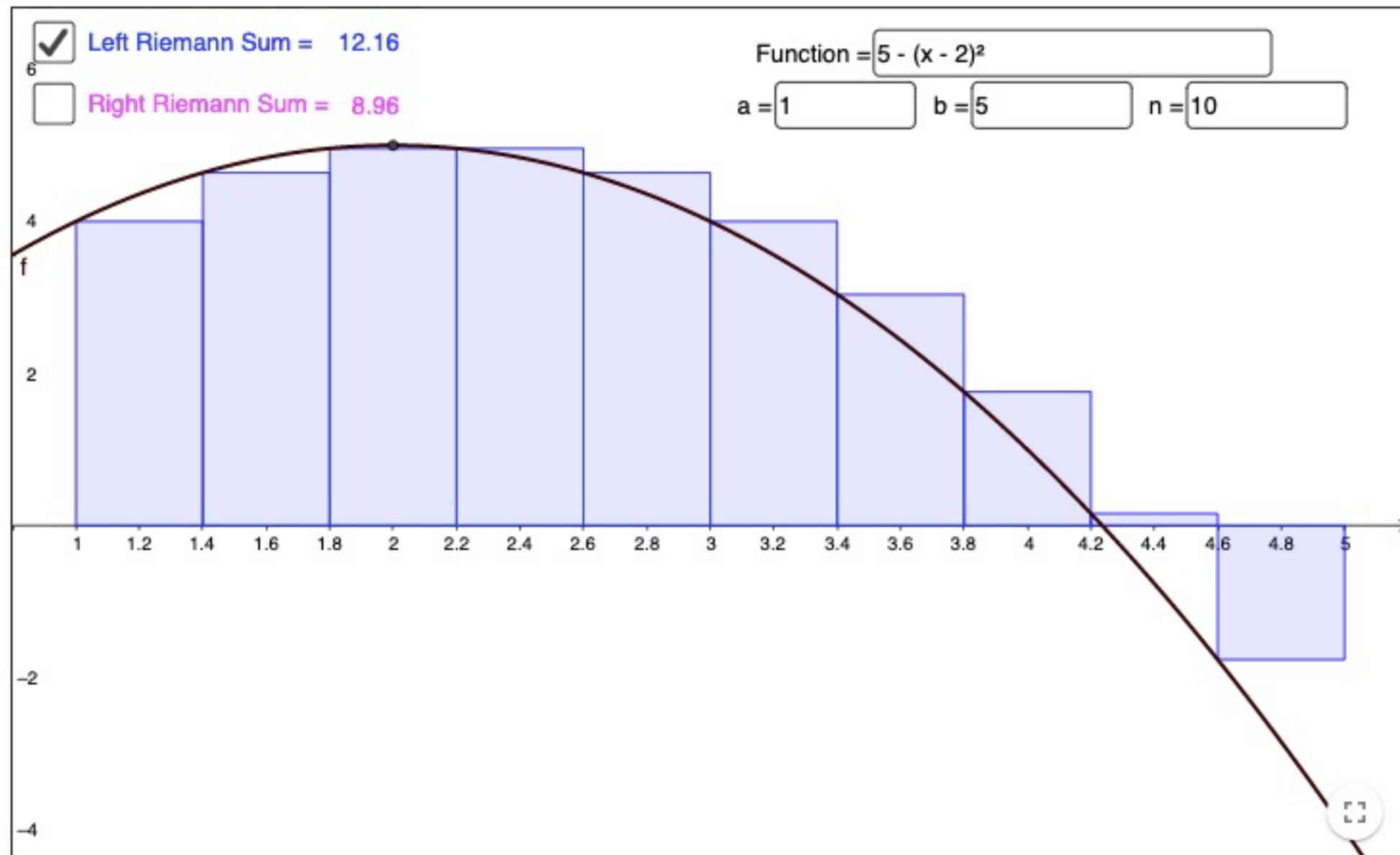
Example: Prime Finder

```
Prime numbers:
2 is prime
3 is prime
4 is not prime
5 is prime
6 is not prime
7 is prime
8 is not prime
9 is not prime
…
```
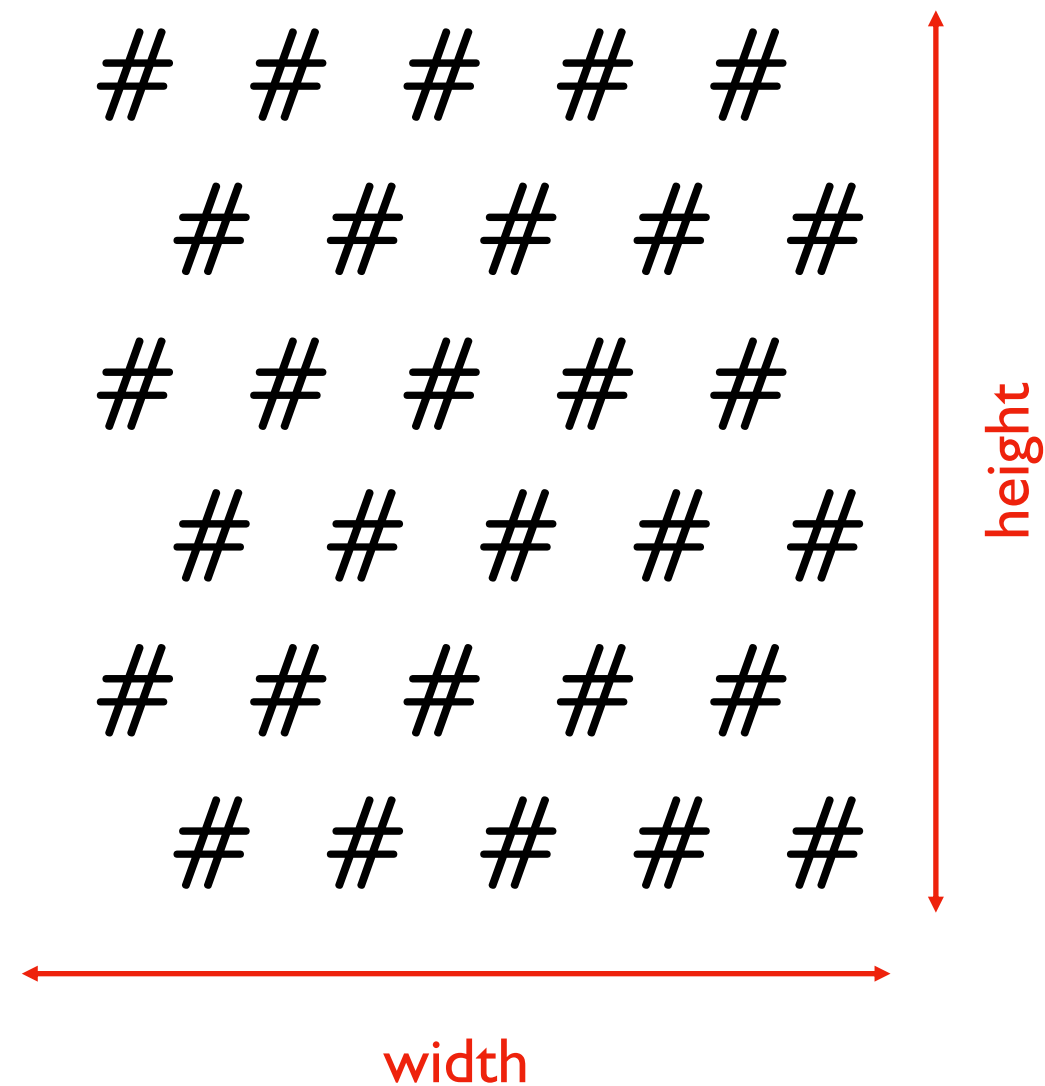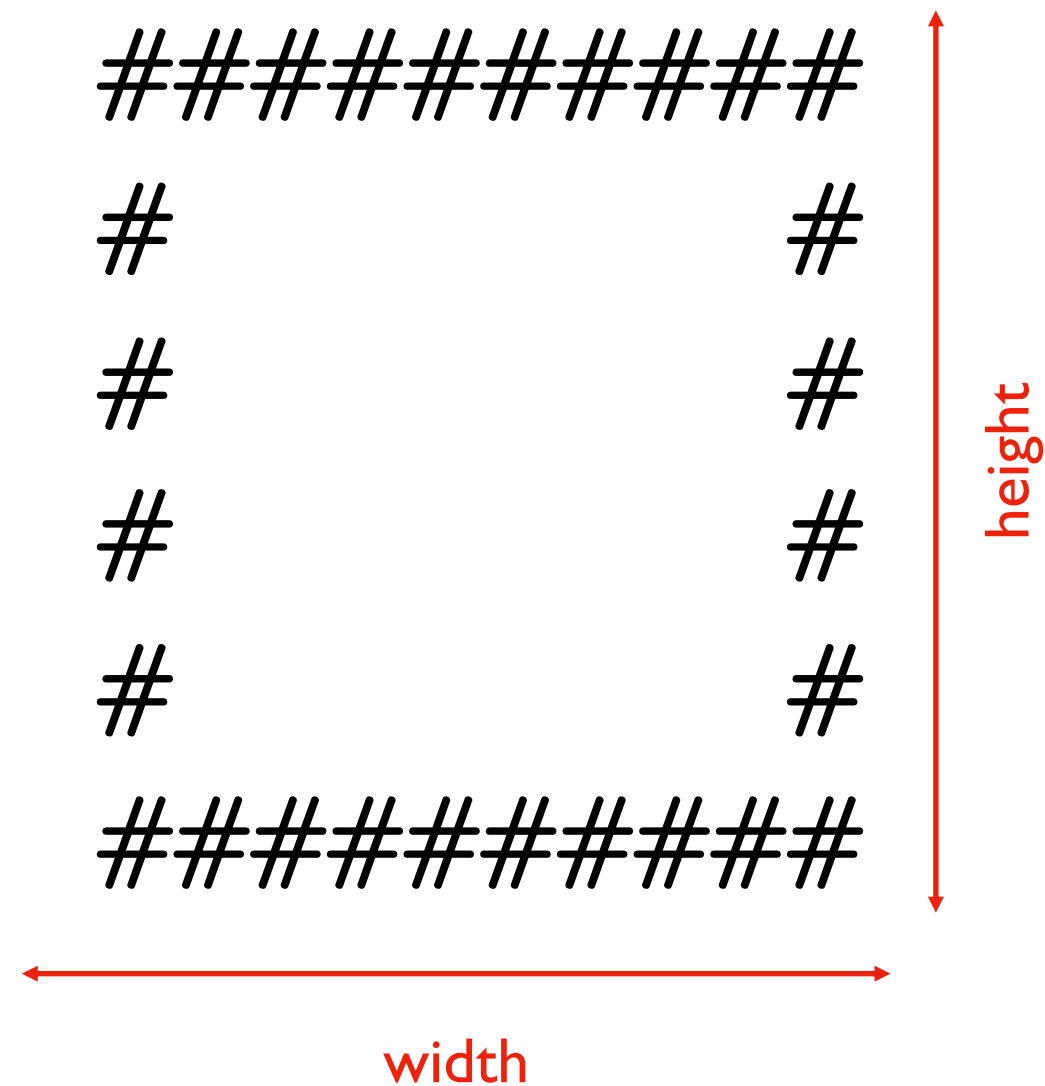
Practice: Checkers
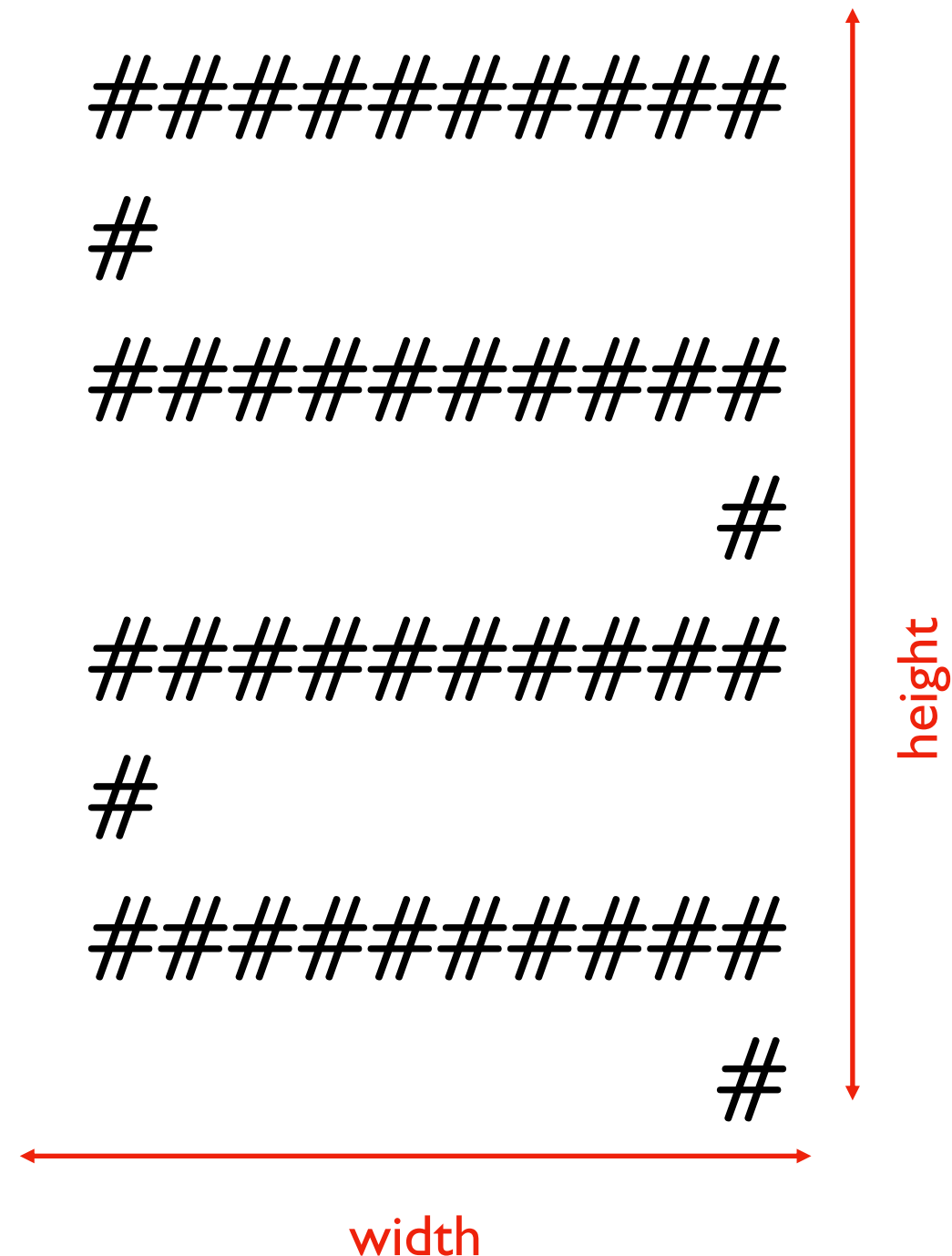
write a loop to draw the following:

```
#  #  #  #  #
 #  #  #  #  #
#  #  #  #  #
 #  #  #  #  #
#  #  #  #  #
 #  #  #  #  #
```

height

width

Practice: Border

write a loop to draw the following:

```
##########
#        #
#        #
#        #
#        #
##########
```

height

width

Practice: Snake

write a loop to draw the following:

```
##########
#
##########
         #
##########
#
##########
         #
```

height

width

Challenge: Countdown Timer

use `time.sleep(1)`

```
how many seconds? 5
5
4
3
2
1
DING DING DING DING DING!
how many seconds? 2
2
1
0
DING DING DING DING DING!
how many seconds? q
good bye!
```

exit program

this program should involve a nested loop!!!