Week 7
CSC111 — Spring 2015

Dominique Thiébaut
dthiebaut@smith.edu
While loops

Examples

Exercises

Graphics
While loops

- for-loops are **definite**: we know ahead of time how many times we are going to loop

- Sometimes we do not know how many times we need to repeat some work until we are done.

- That's why we have **while-loops**
someVariable = <initial condition>

while <someCondition == True>:
    do some work
    update someVariable
Example 1

Keep on reading numbers from the keyboard until a special number is entered.
sum = 0

x = int( input( "> " ) )
while x != -1:
    # use x in some computation
    sum = sum + x

    # get another value
    x = int( input( "> " ) )

print( "sum = ", sum )
sum = 0

x = int(input("> "))
while x != -1:
    # use x in some computation
    sum = sum + x

    # get another value
    x = int(input("> "))

print("sum = ", sum)
Example 2

\[ \frac{1}{2} + \frac{1}{4} + \frac{1}{8} + \frac{1}{16} + \frac{1}{32} + \frac{1}{64} \ldots = 1 \]

How many terms do we need to add up until the difference between the sum and 1 is less than 0.01?
Example 2

\[
\frac{1}{2} + \frac{1}{4} + \frac{1}{8} + \frac{1}{16} + \frac{1}{32} + \frac{1}{64} \ldots = 1
\]

How many terms do we need to add up until the difference between the sum and 1 is less than 0.01?
Example 2

\[ \frac{1}{2} + \frac{1}{4} + \frac{1}{8} + \frac{1}{16} + \frac{1}{32} + \frac{1}{64} \ldots = 1 \]

How many terms do we need to add up until the difference between the sum and 1 is less than 0.01?
Example 2

\[
\frac{1}{2} + \frac{1}{4} + \frac{1}{8} + \frac{1}{16} + \frac{1}{32} + \frac{1}{64} \ldots = 1
\]

How many terms do we need to add up until the difference between the sum and 1 is less than 0.01?
Example 2

\[
\frac{1}{2} + \frac{1}{4} + \frac{1}{8} + \frac{1}{16} + \frac{1}{32} + \frac{1}{64} \ldots = 1
\]

How many terms do we need to add up until the difference between the sum and 1 is less than 0.01?
Example 2

\[ \frac{1}{2} + \frac{1}{4} + \frac{1}{8} + \frac{1}{16} + \frac{1}{32} + \frac{1}{64} \ldots = 1 \]

How many terms do we need to add up until the difference between the sum and 1 is less than 0.01?
Example 2

\[ \frac{1}{2} + \frac{1}{4} + \frac{1}{8} + \frac{1}{16} + \frac{1}{32} + \frac{1}{64} \ldots = 1 \]

How many terms do we need to add up until the difference between the sum and 1 is less than 0.01?
sum = 0
count = 0
powerOf2 = 2

while 1.0 - sum > 0.01:
    sum = sum + 1/powerOf2
    count = count + 1
    powerOf2 = powerOf2 * 2

print( "Sum of 1/2 to 1/" , powerOf2, " = ", sum, sep = "" )
print( "A total of ", count, "terms were added up" )
Example

• Elevator
• First come, first served
• Weights known

Weights = [ 175, 150, 190, 139, 205, 130, 130, 170]
maxElevator = 1000  #lbs
Example

- Elevator
- First come, first served
- Weights known

Weights = [ 175, 150, 190, 139, 205, 130, 130, 170]
maxElevator = 1000  #lbs

Who gets in the elevator?
weights = [ 175, 150, 190, 139, 205, 130, 130, 170]
maxElevator = 1000

# initialize the sum and index
sum = 0
index = 0

# keep on adding people until the sum of all the weights is too large
# or there aren't any people left waiting
while index < len(weights) and sum + weights[index] <= maxElevator:
    sum = sum + weights[index]
    index = index + 1

# identify who is allowed to get in the elevator
selectedPeople = weights[0 : index]

# reduce the line of people waiting
weights = weights[index:]

# print the result
print("Selected people: ", selectedPeople)
print("sum of weights : ", sum)
print("people left : ", weights)
While loops

Examples

Exercises

Graphics
Computer Graphics is Covered in Chapter 4
Graphics Library

• Can be found here: http://cs.smith.edu/dftwiki/index.php/Zelle%27s_Graphics.py_for_Python_3

• And in the Links and Resources section of the class Web page
# graphics.py

"""Simple object oriented graphics library

The library is designed to make it very easy for novice programmers to
experiment with computer graphics in an object oriented fashion. It is
written by John Zelle for use with the book "Python Programming: An
Introduction to Computer Science" (Franklin, Beedle & Associates).

LICENSE: This is open-source software released under the terms of the
GPL (http://www.gnu.org/licenses/gpl.html).

PLATFORMS: The package is a wrapper around Tkinter and should run on
any platform where Tkinter is available.

INSTALLATION: Put this file somewhere where Python can see it.

OVERVIEW: There are two kinds of objects in the library. The GraphWin
class implements a window where drawing can be done and various
GraphicsObjects are provided that can be drawn into a GraphWin. As a
simple example, here is a complete program to draw a circle of radius
10 centered in a 100x100 window:

from graphics import *

def main():
    win = GraphWin("My Circle", 100, 100)
    c = Circle(Point(50,50), 10)
    c.draw(win)
    win.getMouse() # Pause to view result
    win.close()     # Close window when done

main()

GraphWin objects support coordinate transformation through
setCoords method and pointer-based input through getMouse method.

The library provides the following graphical objects:

Point
Line
Doing Graphics:

1. **Open** a graphic window

2. **Draw** on it

3. **Close** it

4. Terminate the program
from graphics import *

def main():
    win = GraphWin("CSC111", 600, 400)
    c = Circle(Point(50,50), 10)
    c.draw(win)

    win.getMouse() # Pause to view result
    win.close() # Close window when done

main()
from graphics import *

def main():
    win = GraphWin("CSC111", 600, 400)
    c = Circle(Point(50,50), 10)
    c.draw(win)

    win.getMouse()  # Pause to view result
    win.close()  # Close window when done

main()
Objects to Play With:

• Points
• Circles
• Rectangles
• Labels (text)
Points

• Used to anchor other objects (circles or rectangles)

• Defined by $x$ and $y$ coordinates

```python
# create a point at location (50, 50)
p1 = Point(50,50)

# create a point at location (50, 50)
p2 = Point(150,50)
```
Circles

- Defined by a **center** and a **radius**
- The center is a **Point**

```python
# create a circle centered at (50, 50) with radius 70
c1 = Circle( Point(50,50), 70 )
c1.draw( win )
```
Rectangles

• Defined by a **top-left**, and a **bottom-right point**

```python
# create a rectangle with top-left corner
# at (5,5) and bottom-right at (50,50)

r3 = Rectangle( Point(5,5), Point( 50, 50) )
r3.draw( win )
```
Labels

• Defined by an anchor **Point**, and

• A **string** that is displayed, centered on the anchor point.

```python
# Create a text label centered at (100,100) # and containing "Smith College"

label3 = Text( Point(100,100), "Smith College" )
label3.draw( win )
```
# create a rectangle with top-left corner
# at (5,5) and bottom-right at (50,50)

r3 = Rectangle( Point(5,5), Point( 50, 50) )
r3.setFill( "red" )
r3.draw( win )
Lot's of Colors to Choose from

http://cs.smith.edu/dftwiki/index.php/Tk_Color_Names
Demo Time
from graphics import *

def main():
    win = GraphWin( "Lab 7 Moving ball", 600, 400 )

    # create and draw a red circle
    center = Point( 100, 100 )
    circ = Circle( center, 30 )
    circ.setFill( 'red' )
    circ.draw( win )

    # set initial direction of ball
    dx = 1.5
    dy = 0.25

    # move ball on screen
    while win.checkMouse() == None:
        circ.move( dx, dy )
        #x = circ.getCenter().getX()
        #y = circ.getCenter().getY()

    win.close()    # Close window when done

main()
Understanding Motion

Center at (3,2)

circ
Understanding Motion

Center at (3,2)

deltaX = 2.5
deltaY = 1
Understanding Motion

Center at (3,2)

circ

deltaX = 2.5

deltaY = 1

circ.move( deltaX, deltaY )
Understanding Motion

\[ \text{circ.move}(\text{deltaX}, \text{deltaY}) \]

Center at (5.5, 3)

\[ \text{deltaX} = 2.5 \]
\[ \text{deltaY} = 1 \]
Understanding Motion

circ.move( deltaX, deltaY )
circ.move( deltaX, deltaY )

deltaX = 2.5

deltaY = 1

circ

Center at (5.5, 3)
Understanding Motion

deltaX = 2.5

deltaY = 1

```
circ.move( deltaX, deltaY )
circ.move( deltaX, deltaY )
```
We stopped here last time...
Concepts of Graphics & Animation
Every element is an OBJECT

Examples

Organization of a graphic program

Something completely different...
Every Graphic Element is an Object
p = Point( 50, 150 )
$p = \text{Point}(\ 50, \ 150 \ )$

$p.getX()$ returns 50
Circle

Circle Object
Circle

Circle Object

getCenter()

center
radius

x, y
Circle

Circle Object

getCenter()
Circle

Circle Object

center
radius

getCenter()

Point Object

getX()

50 number

D. Thiebaut, Computer Science, Smith College
Circle

Circle Object

center
radius

circle

getCenter()

Point Object

getX()

x, y

50
number

getCenter()

Point Object

getX()
Rectangle

Rectangle Object

getP1()

getP2()
Rectangle

Rectangle Object

getP1()

getP2()

Point Object

P1

P2

x, y
Rectangle

Rectangle Object

getP1()

getP2()

P1

P2

Point Object

getX()

x, y
Rectangle

Rectangle Object

getP1()

getP2()

Point Object

getX()

50 number

x, y

P1

P2
Text (label)

Text Object

getAnchor()

getText()
Text (label)

Text Object

getAnchor()

getText()
Text (label)

Text Object

anchor
text

getAnchor()

getText()

Point Object

getX()

x, y
Text (label)

Text Object

anchor
text

getAnchor()

getText()

Point Object

getX()

x, y
Text (label)

Text Object

anchor

text

getAnchor()

getText()

Point Object

getX()

x, y

245 number

D. Thiebaut, Computer Science, Smith College
Where is all this Defined?

Zelle's Graphics.py for Python 3

--D. Thiebaut (talk) 11:12, 8 March 2015 (EDT)

The file below, copyrighted by John Zelle, was downloaded from http://mcsp.wartburg.edu/zelle/python/graphics.py on 3/8/15, and mirrored here for convenience.

```
gravity.py
"""Simple object oriented graphics library

The library is designed to make it very easy for novice programmers to experiment with computer graphics in an object oriented fashion. It is written by John Zelle for use with the book "Python Programming: An Introduction to Computer Science" (Franklin, Beedle & Associates).

LICENSE: This is open-source software released under the terms of the GPL (http://www.gnu.org/licenses/gpl.html).

PLATFORMS: The package is a wrapper around Tkinter and should run on any platform where Tkinter is available.

INSTALLATION: Put this file somewhere where Python can see it.

OVERVIEW: There are two kinds of objects in the library. The GraphWin class implements a window where drawing can be done and various GraphicsObjects are provided that can be drawn into a GraphWin. As a simple example, here is a complete program to draw a circle of radius 10 centered in a 100x100 window:
```

http://cs.smith.edu/dftwiki/index.php/Zelle%27s_Graphics.py_for_Python_3
Every element is an OBJECT

Examples

Organization of a graphic program

Something completely different...
Examples
p = Point( 50, 150 )

getX()

x = point1.getX()
y = point1.getY()
if x <= 0 or y <= 0:
    # the point is outside the window
    ...

point1 = Point( 50, 150 )
Circle Object

point1 = Point( 50, 150 )
circ1 = Circle( point1, 30 )

center1 = circ1.getCenter()
x = center1.getX()
y = center1.getY()
if x <= 0 or y <= 0:
    # the center is outside the window
    ...

Point Object

50 number
point1 = Point(50, 150)
circle1 = Circle(point1, 30)

center1 = circle1.getCenter()
x = center1.getX()
y = center1.getY()
if x <= 0 or y <= 0:
    # the center is outside the window...

x = circle1.getCenter().getX()
y = circle1.getCenter().getY()
if x <= 0 or y <= 0:
    #
Rectangle Object

```python
r = Rectangle( Point( 50, 150 ),
               Point( 150, 150 ) )
r.move( dx, dy )

x1 = r.getP1().getX()
y1 = r.getP1().getY()
x2 = r.getP2().getX()
y2 = r.getP2().getY()

mouseP = win.checkMouse()
if mouseP != None:
    x = mouseP.getX()
y = mouseP.getY()
    if x1 <= x <= x2 and ...
```
Every element is an OBJECT

Examples

Organization of a graphic program

Something completely different...
Organization of a Graphic Program
# open the graphic window

# define and initialize the graphic objects

# start animation loop. Stop on specific user interaction
while win.checkMouse() == None:
    # move/update each object according to its speed
    # and direction

# Loop is over.
# close the graphic window
# open the graphic window
win = GraphWin("Demo", 600, 400 )

# define and initialize the graphic objects

# start animation loop. Stop on specific user interaction

  # move/update each object according to its speed
  # and direction

# Loop is over.
# close the graphic window
# open the graphic window
win = GraphWin( "Demo", 600, 400 )

# define and initialize the graphic objects
circ = Circle( Point( 100, 100 ), 30 )
circ.setFill( 'red' )
circ.draw( win )
dx, dy = 3, 2

# start animation loop. Stop on specific user interaction

# move/update each object according to its speed
# and direction

# Loop is over.
# close the graphic window
# open the graphic window
win = GraphWin( "Demo", 600, 400 )

# define and initialize the graphic objects
circ = Circle( Point( 100, 100 ), 30 )
circ.setFill( 'red' )
circ.draw( win )

dx, dy = 3, 2

# start animation loop. Stop on specific user interaction
while win.checkMouse() == None:
    # move/update each object according to its speed
    # and direction

# Loop is over.
# close the graphic window
# open the graphic window
win = GraphWin( "Demo", 600, 400 )

# define and initialize the graphic objects
circ = Circle( Point( 100, 100 ), 30 )
circ.setFill( 'red' )
circ.draw( win )
dx, dy = 3, 2

# start animation loop. Stop on specific user interaction
while win.checkMouse() == None:
    # move/update each object according to its speed
    # and direction
    circ.move( dx, dy )

# Loop is over.
# close the graphic window
# open the graphic window
win = GraphWin( "Demo", 600, 400 )

# define and initialize the graphic objects
circ = Circle( Point( 100, 100 ), 30 )
circ.setFill( 'red' )
circ.draw( win )
dx, dy = 3, 2

# start animation loop. Stop on specific user interaction
while win.checkMouse() == None:
    # move/update each object according to its speed
    # and direction
    circ.move( dx, dy )

# Loop is over.
# close the graphic window
# open the graphic window
win = GraphWin( "Demo", 600, 400 )

# define and initialize the graphic objects
circ = Circle( Point( 100, 100 ), 30 )
circ.setFill( 'red' )
circ.draw( win )
dx, dy = 3, 2

# start animation loop. Stop on specific user interaction
while win.checkMouse() == None:
    # move/update each object according to its speed
    # and direction
    circ.move( dx, dy )

# Loop is over.
# close the graphic window
# open the graphic window
win = GraphWin( "Demo", 600, 400 )

# define and initialize the graphic objects
circ = Circle( Point( 100, 100 ), 30 )
circ.setFill( 'red' )
circ.draw( win )
dx, dy = 3, 2

# start animation loop. Stop on specific user interaction
while win.checkMouse() == None:
    # move/update each object according to its speed
    # and direction
    circ.move( dx, dy )

# Loop is over.
# close the graphic window
# open the graphic window
win = GraphWin( "Demo", 600, 400 )

# define and initialize the graphic objects
circ = Circle( Point( 100, 100 ), 30 )
circ.setFill( 'red' )
circ.draw( win )
dx, dy = 3, 2

# start animation loop. Stop on specific user interaction
while win.checkMouse() == None:
  # move/update each object according to its speed
  # and direction
  circ.move( dx, dy )

# Loop is over.
# close the graphic window
# open the graphic window
win = GraphWin( "Demo", 600, 400 )

# define and initialize the graphic objects
circ = Circle( Point( 100, 100 ), 30 )
circ.setFill( 'red' )
circ.draw( win )
dx, dy = 3, 2

# start animation loop. Stop on specific user interaction
while win.checkMouse() == None:

    # move/update each object according to its speed
    # and direction
    circ.move( dx, dy )

# Loop is over.
# close the graphic window
win.close()
# open the graphic window
win = GraphWin( "Demo", 600, 400 )

# define and initialize the graphic objects
circ = Circle( Point( 100, 100 ), 30 )
circ.setFill( 'red' )
circ.draw( win )
dx, dy = 3, 2

# start animation loop. Stop on specific user interaction
while win.checkMouse() == None:
    # move/update each object according to its speed
    # and direction
    circ.move( dx, dy )

# Loop is over.
# close the graphic window
win.close()
Every element is an OBJECT

Examples

Organization of a graphic program

Something completely different...

(see class Web page…)

D. Thiebaut, Computer Science, Smith College