MCS-177
Introduction to
Computer Science I

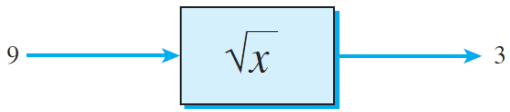
Louis Lei Yu

What We Learned So Far

- Integers and Numbers
  - Arithmetic operations
  - Conversion
- Variables

Abstraction and Functions

- Black Box
- Container for a sequence of actions
- Use the function by name



turtle module

- Many of the additional parts of Python functionality are found in **modules** – an optional part of Python that implements an abstraction that is designed to make program easier.
- The statement you need to use to load a module is *import*

Turtle Module

- Every object in Python has 3 important characteristics:
  - An identity
  - A type
  - A value
- Some Python objects have special values called *attributes*
- Some Python objects have *functions/ methods*

Turtle Attributes

position	The coordinates of the turtle on the screen
heading	The direction the turtle is facing
color	The color of the turtle
tail position	The turtle's tail can be up or down

Table 1.2 Turtle attributes

Summery of turtle methods

Name	Parameter(s)	Description
`Turtle`	None	Creates and returns a new turtle object
`forward`	Distance	Moves the turtle forward
`backward`	Distance	Moves the turle backward
`right`	Angle	Turns the turtle clockwise
`left`	Angle	Turns the turtle counterclockwise
`up`	None	Picks up the turtle's tail
`down`	None	Puts down the turtle's tail
`color`	Color name	Changes the color of the turtle's tail
`fillcolor`	Color name	Changes the color that the turtle will use to fill a polygon
`heading`	None	Returns the current heading
`position`	None	Returns the current position
`goto`	`x, y`	Moves the turtle to position `x, y`
`begin_fill`	None	Remembers the starting point for a filled polygon
`end_fill`	None	Closes the polygon and fills it with the current fill color
`dot`	None	Leaves a dot at the current position

Table 1.3 Summary of simple turtle methods

Defining Functions

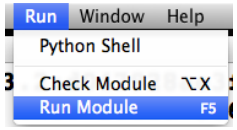
We are not limited to the function the author of Python give us. We can write our own functions to add our own abstractions to the Python language

- Name
- Parameters
- Body

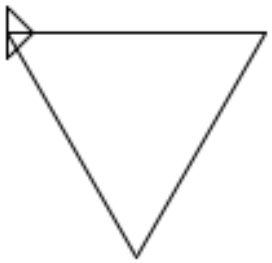
```
def drawSquare(myTurtle, sideLength):
        myTurtle.forward(sideLength)  # side 1
        myTurtle.right(90)
        myTurtle.forward(sideLength)  # side 2
        myTurtle.right(90)
        myTurtle.forward(sideLength)  # side 3
        myTurtle.right(90)
        myTurtle.forward(sideLength)  # side 4
        myTurtle.right(90)
```

Calling a function

```
>>> from ds import*
>>> import cTurtle
>>> t = cTurtle.Turtle()
>>> drawSquare(t, 150)
```



Draw

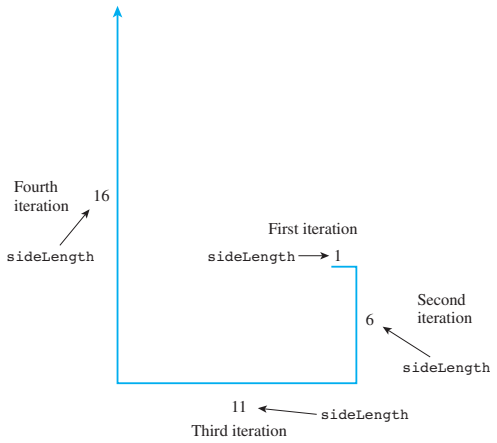


Iteration (for loops)

- Repeat a sequence of steps
- Use a for statement
- Range

```
def drawSquare(myTurtle, sideLength):
        myTurtle.forward(sideLength)  # side 1
        myTurtle.right(90)
        myTurtle.forward(sideLength)  # side 2
        myTurtle.right(90)
        myTurtle.forward(sideLength)  # side 3
        myTurtle.right(90)
        myTurtle.forward(sideLength)  # side 4
        myTurtle.right(90)
```

```
def drawSquare(myTurtle,sideLength):
    for i in range(4):
        myTurtle.forward(sideLength)
        myTurtle.right(90)
```



Drawing a Circle

- Simplify and Generalize
- Polygon with more and more sides

Generalize

- 3 sides – 120 degrees
- 4 sides – 90 degrees
- 5 sides – 72 degrees
- 8 sides – 45 degrees
- N sides - ? Degrees

```
def drawPolygon(myTurtle,sideLength,numSides):
    turnAngle = 360 / numSides
    for i in range(numSides):
        myTurtle.forward(sideLength)
        myTurtle.right(turnAngle)
```

Contract and Docstring

```
#cTurtle object, int -> void
def drawSpiral(myTurtle, maxSide):
    """This method draws a spiral"""
    for sideLength in range(1, maxSide+1, 1):
        turn = maxSide - sideLength
        for i in range(20):
            myTurtle.forward(sideLength/20)
            myTurtle.right(turn/20)


#int, cTurtle object -> void
def tri(sideLength, turtle):
    """This method draws a triangle"""
    turtle.forward(sideLength)
    turtle.right(120)
    turtle.forward(sideLength)
    turtle.right(120)
    turtle.forward(sideLength)
    turtle.right(120)
```

```
>>> help(drawSpiral)
Help on function drawSpiral in module __main__:

drawSpiral(myTurtle, maxSide)
    This method draws a spiral

>>> help(tri)
Help on function tri in module __main__:

tri(sideLength, turtle)
    This method draws a triangle
```