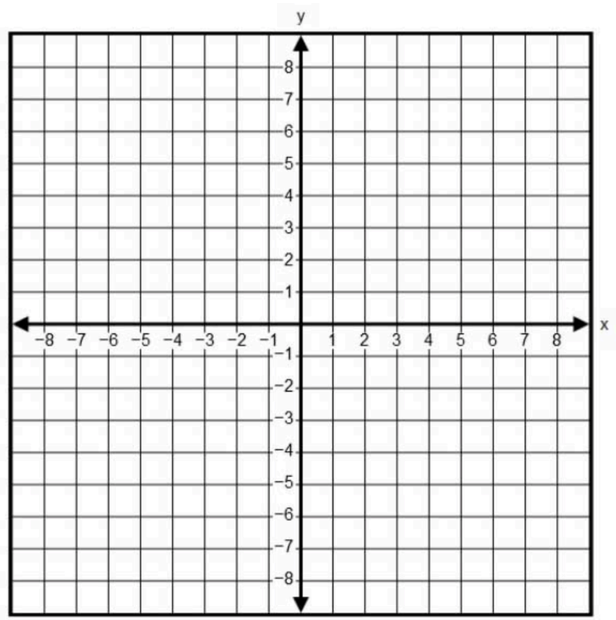
CSC 1002 Week 6

Turtle Graphics



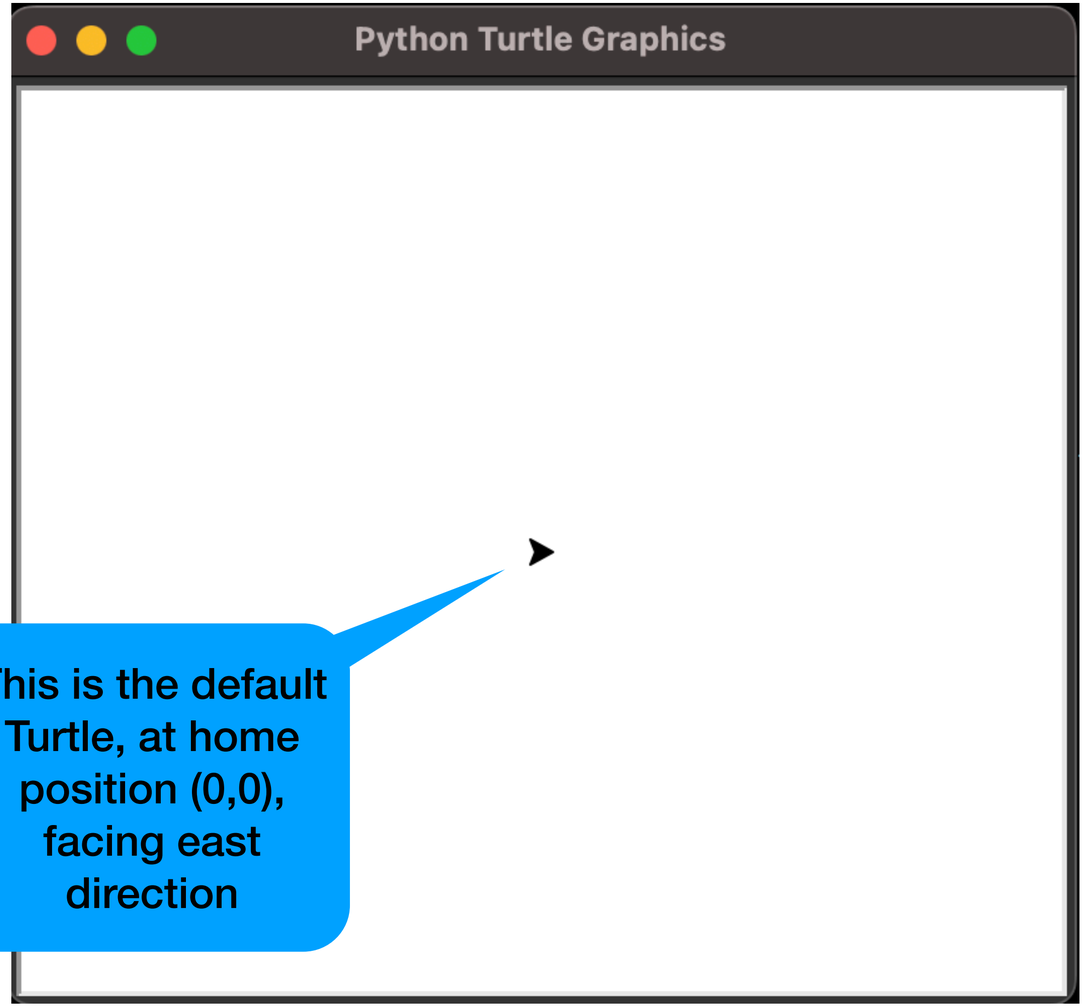
turtle – graphics tools

- Turtle is a graphical tool and it displays a 2-dimensional, XY drawing windows that we can do some simple graphical drawing.
- (Default) The center of the drawing windows is called home and it has a coordinate (0,0)
- It has a drawing pen called “Turtle” initially set at home position
- Draw by moving the pen, the turtle, in either forward and backward direction.
- The pen/turtle has a full-circle heading from 0 to 360 degrees, counter-clockwise, east being the 0 degree. Default heading is 0 degree (facing east)
- Common commands: Move the turtle to a specific (x,y) position, clear the drawing content, send the pen to home, hide or un-hide the pen, set the line color and width, and so on.

Python Turtle Graphics

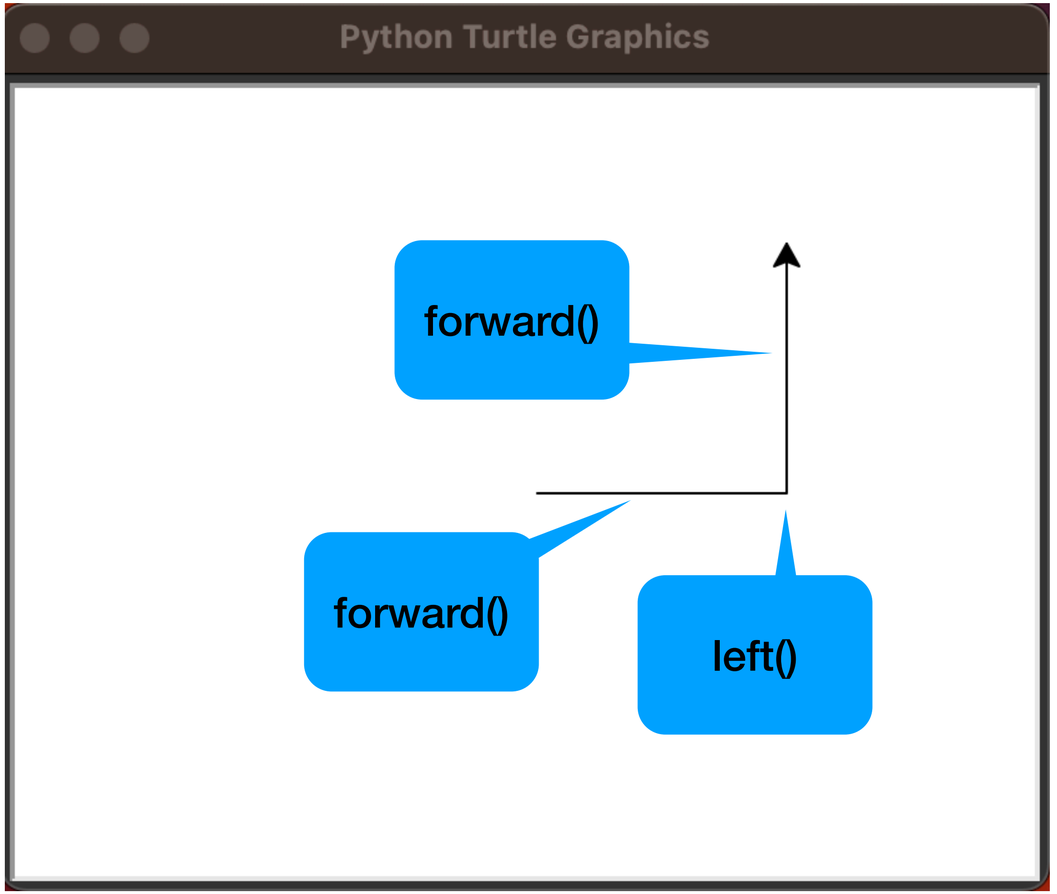
- creating turtle GUI

```
>>> import turtle
>>> tt=turtle.Turtle()
```



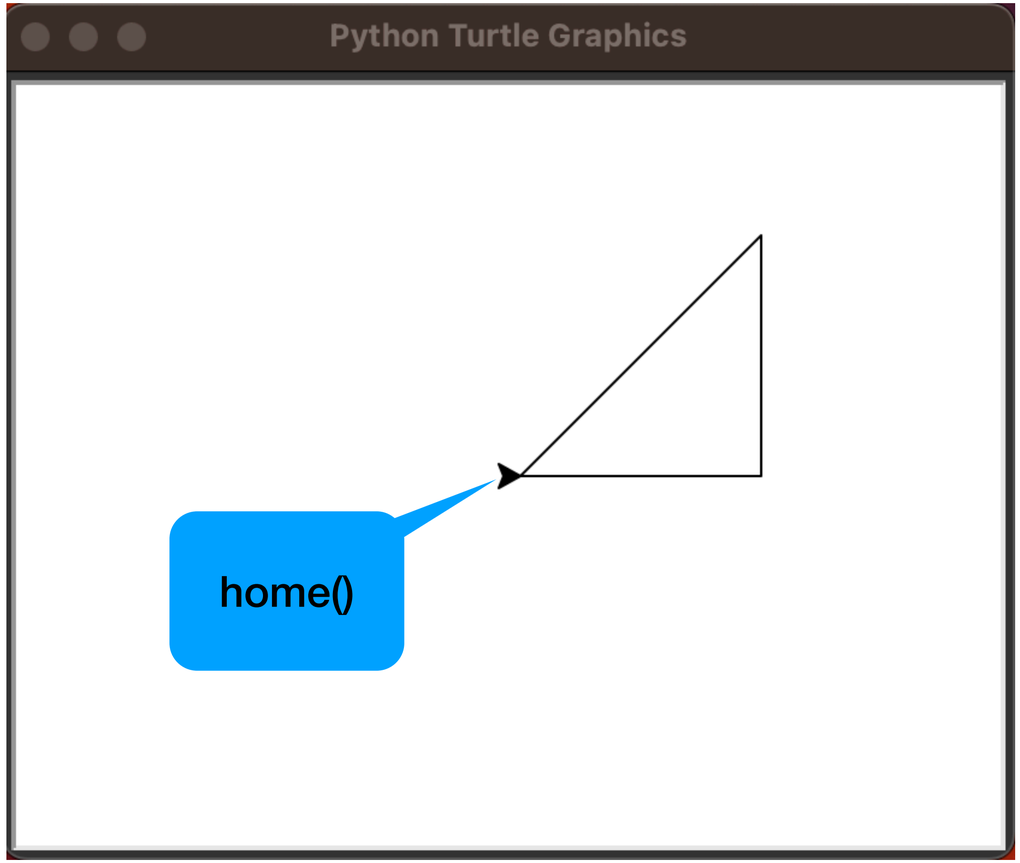
drawing – forward() & left()

```
>>> tt.forward(200)
>>> tt.left(90)
>>> tt.forward(200)
```



drawing – home()

```
>>> tt.forward(200)
>>> tt.left(90)
>>> tt.forward(200)
>>> tt.home()
```



Turtle() and Screen()

- Turtle() – defines the drawing pen used to move around the drawing area. You can create multiple turtle objects, each moved and controlled separately. Each turtle object can have a different shape (square, circle, triangle, turtle) or even a custom shape.
- Screen() – defines the drawing area. It's a singleton object. It defines the boundary of the canvas and enables the interactive mode via mainloop().

Turtle - basic functions

- up(), down(), left(), right(), setheading()
- goto(), forward(), backward()
- clear(), reset(), home()
- pensize(), pencolor()
- shape(), color(), shapesize()
- refer to https://docs.python.org/3/library/turtle.html for more info on turtle graphics

Turtle – more details

- shape() sets or returns the current turtle shape: “square”, “triangle”, “circle”, “standard” and so on.
- color(border color, fill color) or color(fill color)
  - passing an empty string “” for the color parameter will set the color transparent (matching the screen’s background color), but it can no longer be dragged, nor clicked.
- setheading(degree)
- forward(distance), backward(distance), left(degree), right(degree)
- undo() – undo last motion or draw actions
- clone() – make a copy of an existing turtle object (not drawings)
- setposition(x,y)/goto(x,y)

Turtle – States

- pos()/position(), xcor(), ycor() – current turtle's position (x,y)
- get_shapepoly() – dimension of the turtle shape
- distance(turtle or x,y) – returns the distance between 2 turtles
- towards(turtle) – the angle between 2 turtles
- hideturtle() or showturtle(), isvisible()
- up(), down(), isdown()
- shapesize(stretch_x, stretch_y, border_width)

Demo - Turtle

Screen() – singleton object

- Defines the dimension of the drawing canvas area by calling setup(x, y), (0,0) at center
  - window_width() and window_height() returns the width and height respectively
  - setworldcoordinates(lower left corner coordinate, upper right corner coordinate)
- Manual Screen refresh – tracer() & update()
- Title of the screen - title()
- Mode of the heading – mode()
  - standard – east, counterclockwise, logo – north, clockwise
- Active turtles – turtles()
- Screen events
  - onkey() – bind a function on a key press such as “Spacebar”, “Up”, “Down”, “Left”, “Right”, “Return”, and so on.
  - listen() - enable the onkey() events
  - onclick() – bind a function on a mouse-click event
  - ontimer() – bind a function to a timer
  - mainloop() – start event process for turtle graphics, last statement in program

Screen Refresh – Auto vs Manual

- use screen's tracer(0) to turn off auto screen refresh
  - in this case when the turtle is moved (via forward() or goto()) the position of the turtle will not be updated until update() or tracer(1) is called.
- use screen's update() to manually refresh the drawing canvas
  - NOTE: it will redraw all turtle shapes in the order they were created.
- use screen's tracer(1) to resume the normal auto refresh
  - use the second parameter 'delay' to control the refresh rate
    - ex: tracer(1, delay=200)

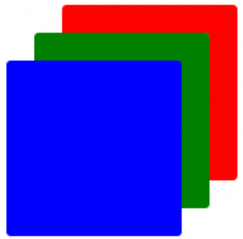
Turtle – Auto vs Manual

- In Auto mode:
  - Any actions applied to a turtle, its shape will be redrawn and displayed on top of other shapes if any.
    - Including: onclick, fd, back, shapsize, color, …. etc.
    - Excluding: write, up, down
- In Manual mode:
  - Any actions (excluding write) applied to any turtles, no changes will be applied to the screen until update() is called.
  - NOTE: update() or tracer(1) will redraw all turtle shapes in the order they were created.
  - NOTE: write() will ALWAYS show the static text on screen (in auto or manual mode)

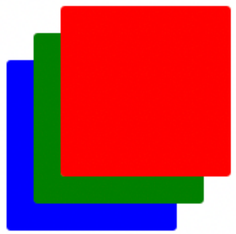
Turtle – Auto vs Manual

- Given 3 turtles created in following sequence (blue, green and red) :

- Then one by one, move and overlap the turtles in reverse sequence: red -> green -> blue
- In Auto mode:



- In Manual mode (after update() or tracer(1)):



Event Loop (Screen)

- In order to keep the turtle graphics screen active, call the screen's mainloop() function
- It must be the last statement of your main process/program
  - turtle.Screen.mainloop()

Screen Events: Key Press & Mouse-Click

- The onkey(func, key) function of the screen object binds a function to a keyboard key:
  - Non-printable keys: “Return”, “Up”, “Down”, “Left”, “Right”, “space”, “Tab”
  - Printable keys: pass to the function as it is, ex: “a”, “1”, “K”, ….
- The onscreenclick()/onclick() binds/unbinds a function to a mouse-click event against the turtle graphics’ canvas.
- Note on Key Press: call listen() to enable the process
- place listen() before mainloop()

Timer Event (Screen)

- use ontimer(func, delay) to call a function after a fixed delay (in milliseconds)
- the ontimer() is set for one time call only
- you need to call ontimer() within the function to repeat the event if needed

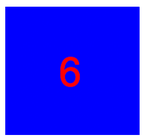
Displaying text (Turtle)

- turtle graphics doesn’t provide a way to write a text string on the screen freely
- need a turtle object and use write() to “draw” a text string where the turtle is.
  - write(string, move, align, font)
- text is static image
  - calling write() again with a blank string at the same position will not erase the text
  - moving the turtle object will not move the text
- either calling undo() to undo the last turtle action or invoking clear() which clears out ALL the drawing content (lines, shapes, string, etc) for that turtle
- use hide() to hide the turtle shape, as needed
- write() always output the string whether the screen is in Auto or Manual refresh mode
- text color follows color of the turtle object (shape)

Write() – Auto vs Manual

- Create one blue square turtle as follows:

- Then create another turtle in red color, make it hidden then move to the center of the blue square turtle, and call write() to display ’6’
- In Auto mode:



- In Manual mode (after update() or tracer(1)):