Teaching London Computing

Programming for GCSE Topic 8.3: Python Turtle Graphics

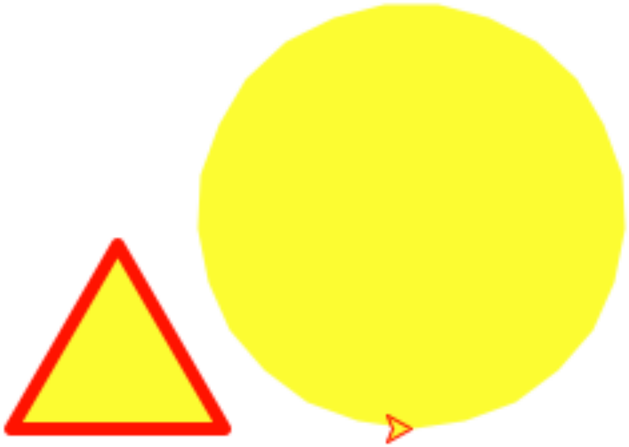
William Marsh
School of Electronic Engineering and Computer Science
Queen Mary University of London

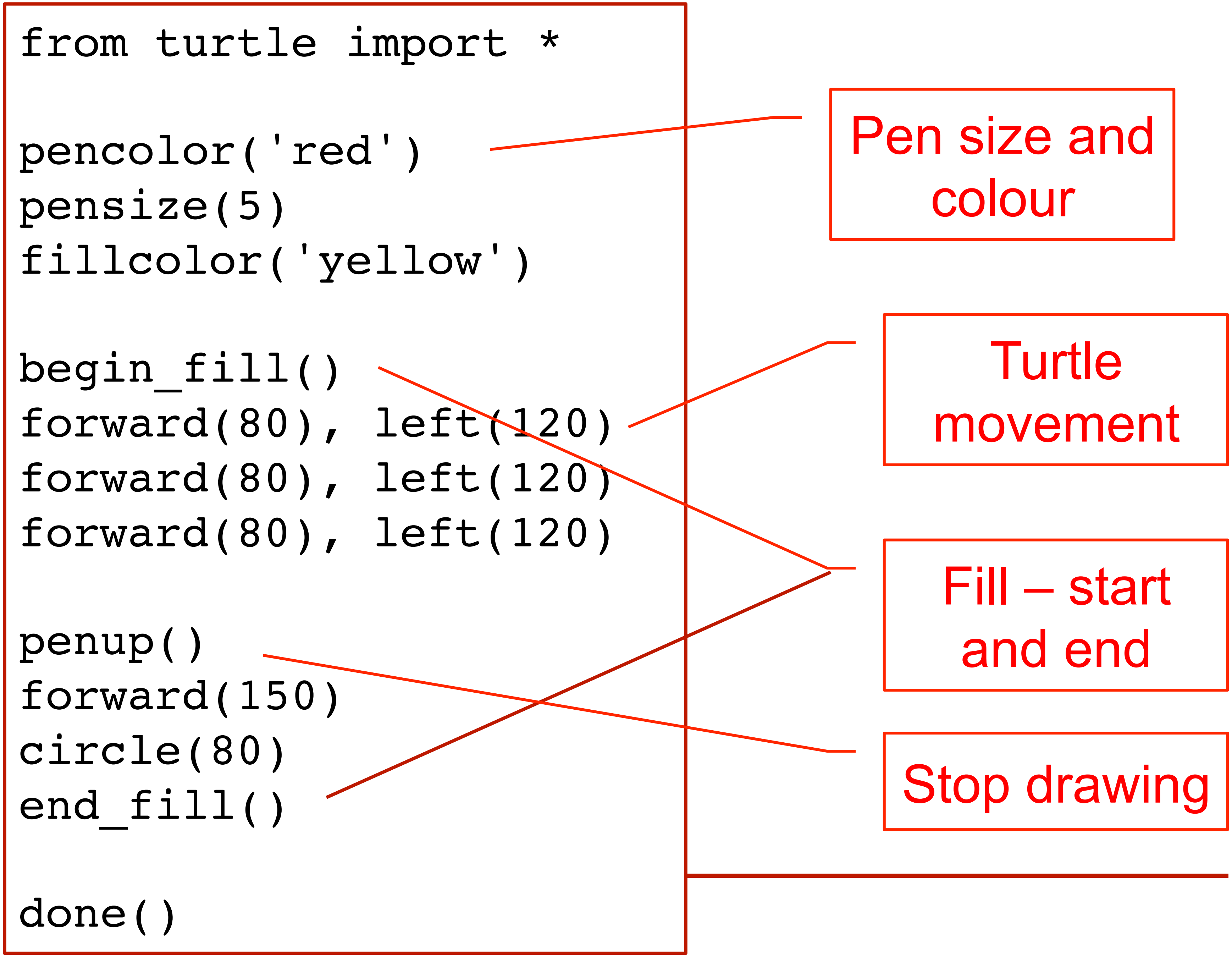
Aims

- Outline the idea of turtle graphics
- Evaluate turtle graphics for learning programming

Principles of Turtle Graphics

```
from turtle import *

pencolor('red')
pensize(5)
fillcolor('yellow')

begin_fill()
forward(80), left(120)
forward(80), left(120)
forward(80), left(120)

penup()
forward(150)
circle(80)
end_fill()

done()
```





```
from turtle import *

pencolor('red')
pensize(5)
fillcolor('yellow')

begin_fill()
forward(80), left(120)
forward(80), left(120)
forward(80), left(120)

penup()
forward(150)
circle(80)
end_fill()

done()
```

Pen size and colour

Turtle movement

Fill – start and end

Stop drawing

Documentation

- See chapter 23 of Python library
  - Functions: simpler, as shown
  - Object-orient methods: more flexible
- Simplified extract in the practical sheet
  - Omits synonyms
  - More complex features

How Useful Is Turtle Graphics?

Could use turtle graphics as an introductions

Advantages

- Visual
  - Picture and simple animation
  - Text I/O included
  - Not really for graphical UI or games
- Problem solving and creativity
- Functions
  - Obvious need for decomposition into functions

Limitations

- Less focus on variables: very important concept
- Many problems can be solved (badly) without control structures (if and loops)
- Not a great match to GCSE syllabus

Example Challenge Problems

- Handle shapes as data
  - Save in file
  - E.g. ['square', 100, 30, 40] – size, x, y
- Do transformations on shapes
  - Rotate
  - Reflect
- Display data
  - E.g. bar or line graph

Summary

- Turtle graphics is simple and visual
- Consider it as an alternative route for transition between visual (e.g. Scratch) and textual programming