COMP1021
Introduction to Computer Science

Using Timers

Gibson Lam and David Rossiter

Outcomes

- After completing this presentation, you are expected to be able to:
  1. Use turtle timers to control exact timing in a Python program
  2. Use turtle timers to create time intervals

Problems with Exact Timing

- We saw previously that we can change the animation speed using `turtle.speed()`
- However `turtle.speed()` cannot control the exact timing of the animation
- For example, you cannot use it to make a stopwatch program, where the seconds hand moves one tick every second exactly

How About time.sleep()?

- We can use `time.sleep()` to control exact timing
- It tells Python to do nothing while waiting for an exact amount of time
- We could use it to build a stopwatch inside a loop, i.e.

```
import time

seconds = 0
while True:
    print(seconds)
    seconds = seconds + 1

    time.sleep(1)
```

```
>>>
0
1
2
3
⋮
```

Problem with time.sleep()

- Using `time.sleep()` is not suitable in the next part of the course
- We want to control timing but we also want to do several things *at the same time*
- We cannot use only `time.sleep()` for clever timing of several things at the same time because it totally stops Python from doing anything else

Using a Turtle Timer

- To properly control timing, we can use a turtle timer event, which can be created using the `ontimer()` function

`turtle.ontimer(` *timer function* `,` *time* `)`

This is the event handling function for the timer

The amount of time to wait (in milliseconds) before the timer function is executed

- A turtle timer allows the Python program to do something else while waiting for the timer

A Timer Example

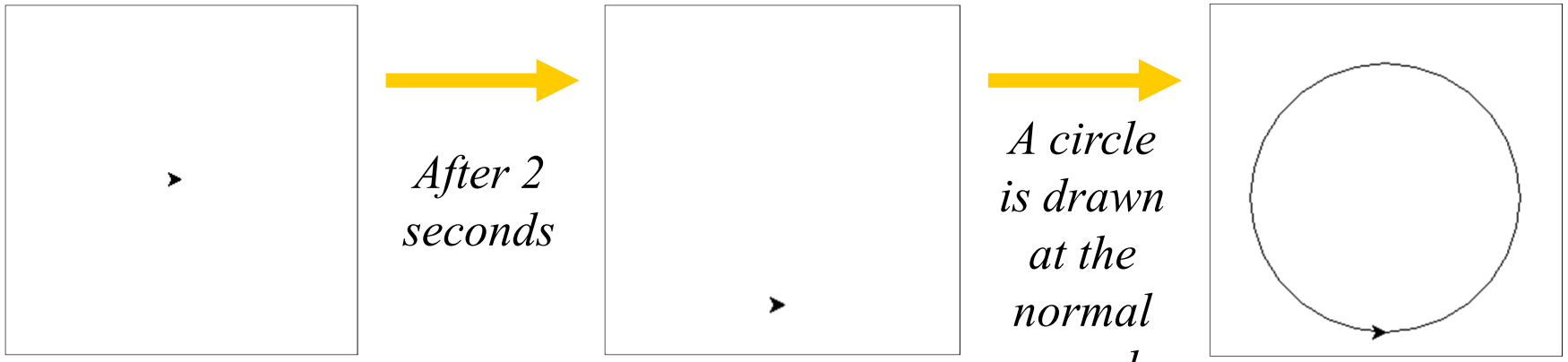
```
import turtle

def draw():
    turtle.up()
    turtle.goto(0, -100)
    turtle.down()
    turtle.circle(100)

turtle.ontimer(draw, 2000)

turtle.done()
```

Set a timer to run the draw function after 2 seconds (=2000 milliseconds)

- In this example the turtle does not do anything at the start
- Two seconds after the program begins it moves the turtle and starts to draw a circle

Result of the Timer Example

- This is the result of the example shown on the previous slide:



The program starts

The turtle starts to draw a circle

Drawing is finished

Creating Time Intervals

- Sometimes, we need to use the timer repeatedly
- For example, perhaps you need your code to do something every second, i.e. a stopwatch
- That means you need a timer to fire repeatedly every second, not just once
- One way to do this is to start the timer again, at the end of the code which is triggered by the previous timer

A Stopwatch Using Timer

- In this example, a turtle timer is used to create a stopwatch program similar to the one we saw before which was created using `time.sleep()`
- Note that this example does not draw anything on the turtle window

```
import turtle

def stopwatch():
    global seconds
    print(seconds)
    seconds = seconds + 1
    turtle.ontimer(stopwatch, 1000)

seconds = 0
stopwatch()
turtle.done()
```

The function sets the timer up to run itself again after one second

Start the stopwatch

Something That Does Not Work

- Sometimes it is useful if you could pass values to the function you want to run later, e.g.

Here we are trying to pass a number to the function which will be executed 9 seconds later

```
turtle.ontimer( addmoney(50), 9000 )
```

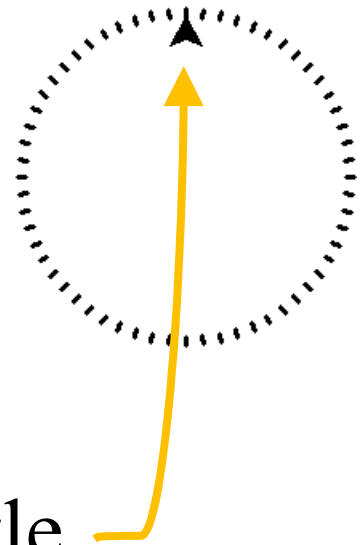
- Unfortunately this does not work in Python
- You will have to use another approach to pass values to the function, such as *global variables* (see the example in the previous slide)

Yet Another Stopwatch Example

- Let's make a more 'beautiful' stopwatch program using turtle graphics
- In this example, the stopwatch is visually shown in the turtle window

Stopwatch – Drawing the Ticks

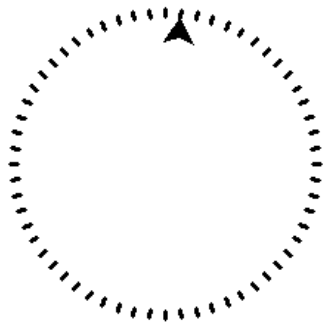
- Some simple turtle graphics code is used to create the ticks of the stopwatch like this:



- After drawing the ticks, the turtle remains at the top of the stopwatch and it acts as the seconds hand in the watch

Stopwatch – Moving the Turtle

- The turtle then moves one tick each second using this code:



Negative number makes the turtle goes backward

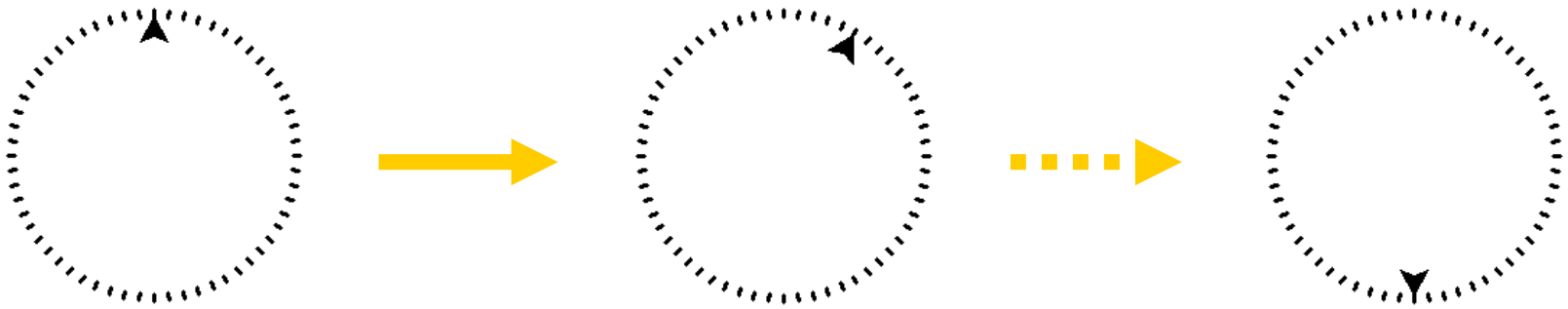
```
def tick():
    turtle.up()
    turtle.left(90)
    turtle.circle(100, -6)
    turtle.right(90)
    turtle.down()

    turtle.update()

    turtle.ontimer(tick, 1000)
```

Move the turtle to the next tick position, i.e. move in the clockwise direction 6 degrees (=1 second)

Move the turtle again one second later

Running the Example

- Here are the examples of the stopwatch in action:



The program starts

5 second after the program starts

30 seconds after the program starts