Moving from Paper to Scratch to Python

SCRATCH

python

Scheme of Work

Year 9 First Autumn Half Term we do:

- Scratch recap – make a basic game in Scratch
- Designing Algorithms/Programs
- Design a Guessing Game
- Make a Guessing Game in Scratch
- Make the same Guessing game in Python

Scheme of Work

Year 9 Second Autumn Half Term we do:

- Design and make a Maths Quiz with 5 questions first in Scratch then Python
- Recap using Logo
- Creating Procedures in Logo (write your name)
- Functions and for loops in Python
- Use the Python Turtle Module (write your name)

Scheme of Work

Year 9 Spring Term we do:

- Create a Rogue-Like Game

Advanced Topics

- Recursion
- Test Driven Development

Designing a Program

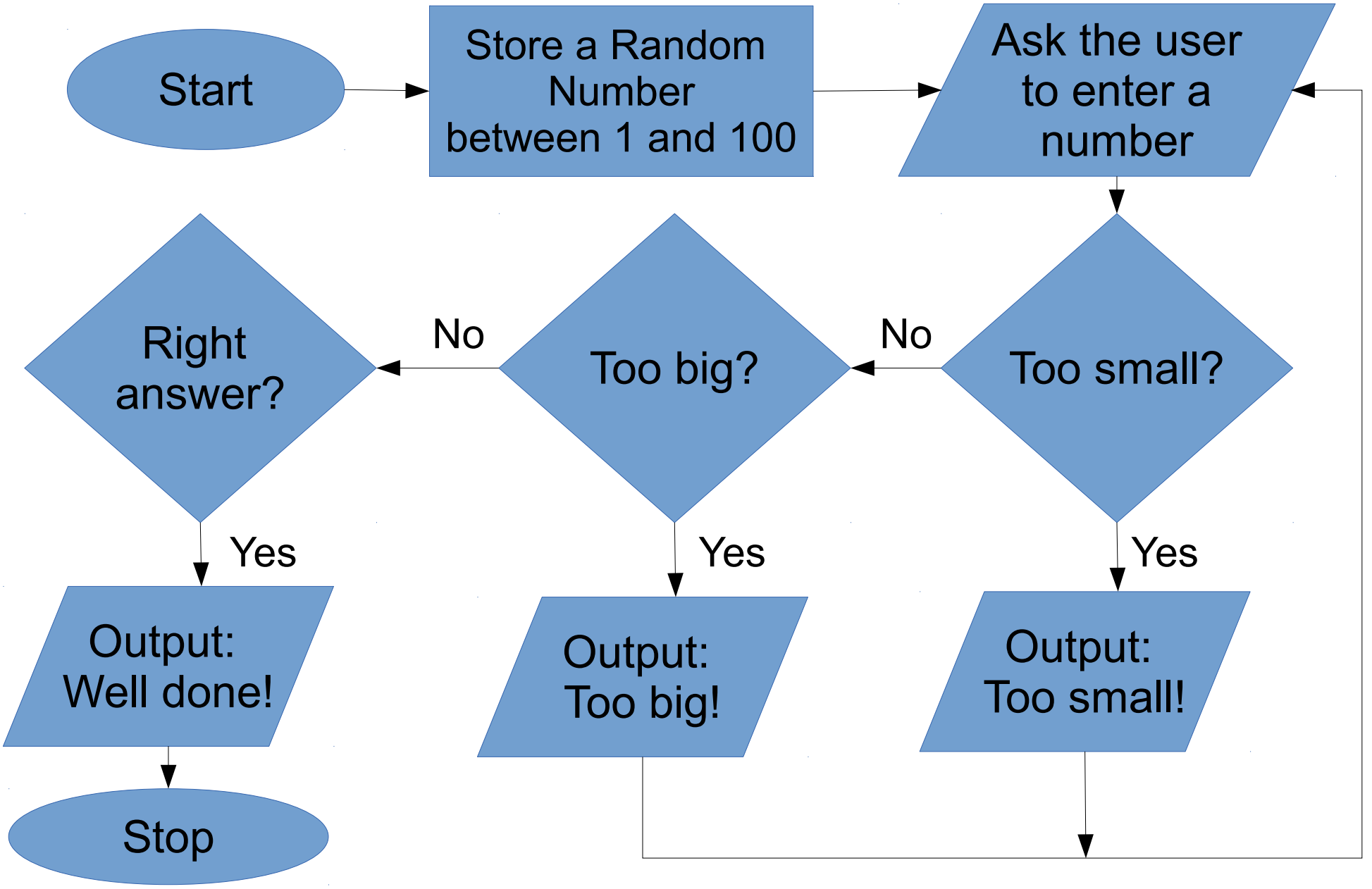
Design a Guessing Game where the Computer generates a random number between 1 and 100 and then responds to your input with one of the following:

- Too big
- Too small
- Well done!

If the correct answer is guessed then the program stops.

Extension: Design a guessing game where the computer guesses what number you're thinking of in the fewest possible steps

Guessing Game Design



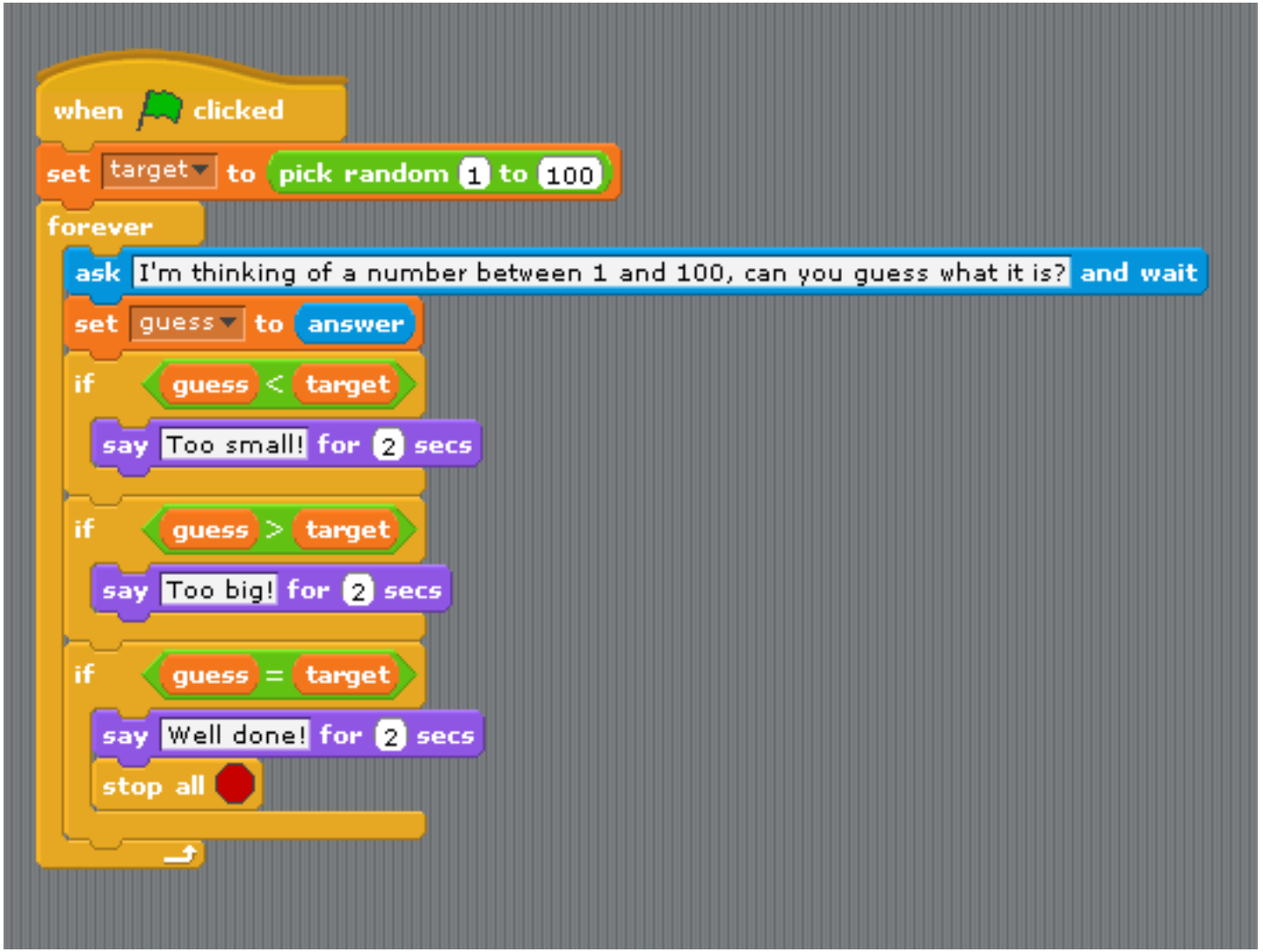
In Scratch

When creating it with students try to match the order of ideas in your diagram and the order of your scripts.

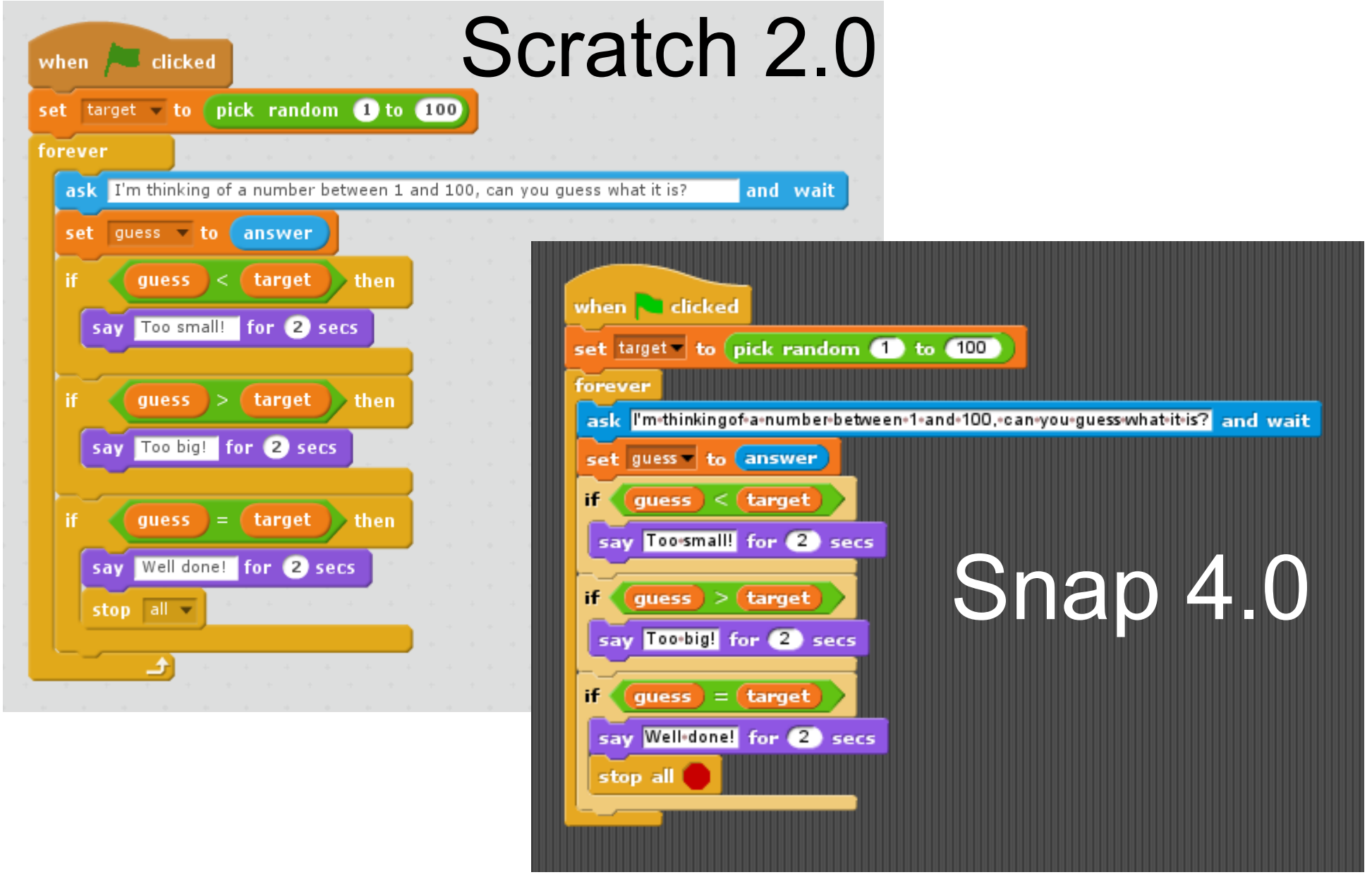
Don't try to get too complicated too soon.

Do the whole thing in one sprite unless you **have** to make it more complicated.

Scratch 1.4


when clicked
set target to pick random 1 to 100
forever
ask I'm thinking of a number between 1 and 100, can you guess what it is? and wait
set guess to answer
if guess < target
say Too small! for 2 secs
if guess > target
say Too big! for 2 secs
if guess = target
say Well done! for 2 secs
stop all

Other Versions of Scratch



Pros and Cons

- Scratch 2.0 is written in Flash (not on a Raspberry Pi)
- Snap is written in JavaScript & HTML5 (can be used **just about** on iPads and Android Devices)
- Both can be a bit temperamental on a School network
- Scratch's Website allows sharing, remixing easily
- Snap's Website will do, but not yet
- Both allow creation of own blocks

Python

We do some basic stuff with variables, while loops and if statements first

Then we spend a few lessons working through the Guessing Game in Python

Most able add features:

- Play Again
- Keep Score
- High (Low) Score Table (uses a list)

Code

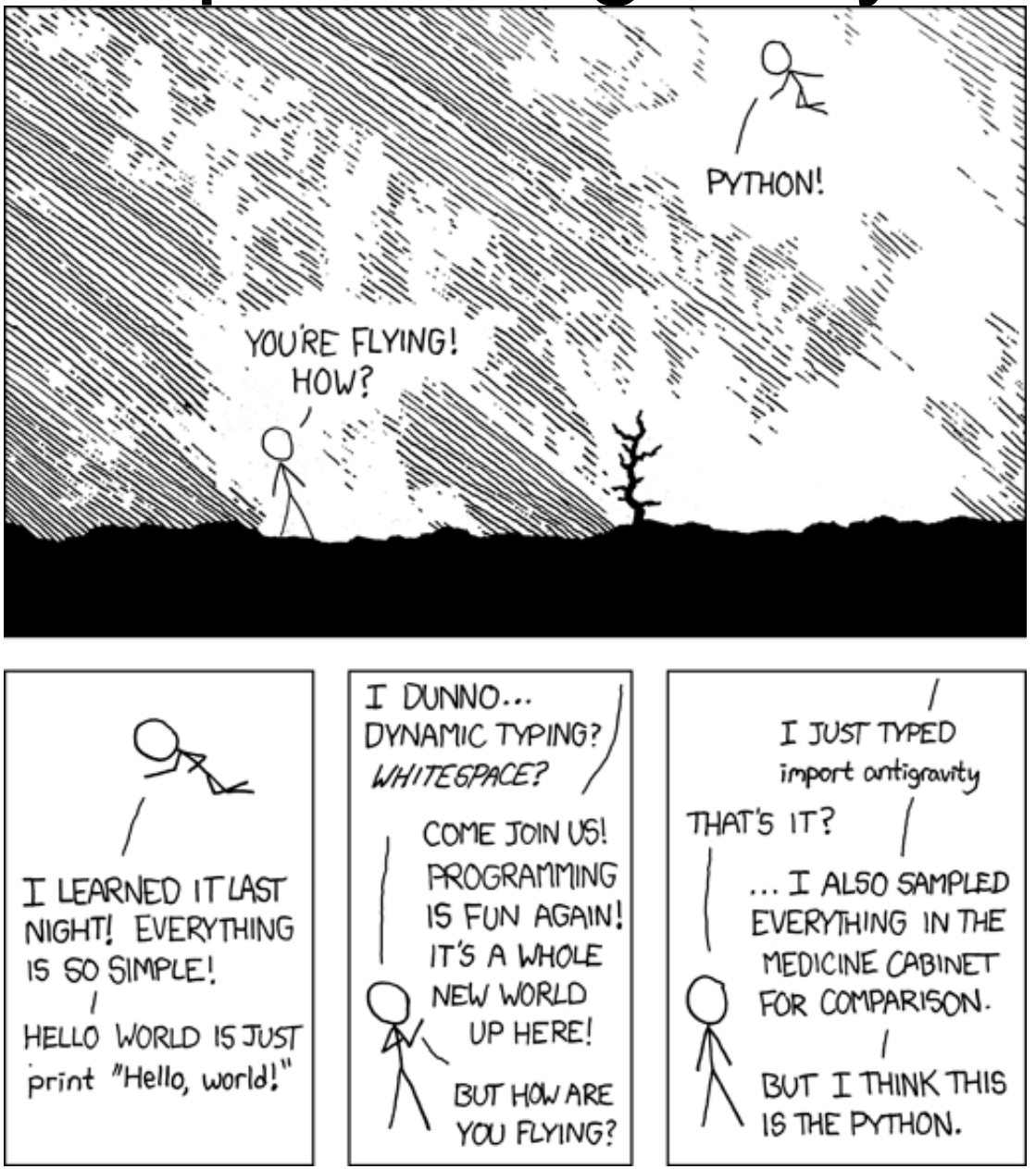
```
1   import random
2
3   target = random.randint(1,  100)
4
5   while True:
6       print("I'm thinking of a number between 1 and 100 can you guess what it is? ")
7       guess = int(input())
8       if guess < target:
9           print("Too small.")
10      if guess > target:
11          print("Too big")
12      if guess == target:
13          print("Well done!")
14          break
15
16
```

The order of each section is the same as in the Scratch version

Python 2 or 3?

- There are lots of resources for both on the web
- We chose Python 3 because it is in active development
- There is a script which will attempt to convert Python 2 files to Python 3 (py2to3) for you automatically
- Not every library converted to Python 3
- **import antigravity**

import antigravity



xkcd.com

IDEs

- IDLE – Built in – minimalist
- Pyscripter Portable – used with class
- Eric/Eric5 – used with class
- Komodo Edit – problem using on our network
- iPython3
- iPython3 Notebook – launches in web browser
- Notepad
- Notepad++

Where to go next?

Logo

- Use it to introduce re-usable code – aka Procedures or Functions
- Create something using Procedures in Logo
- Use the Turtle Module to recreate it in Python

We get them to create Procedures for each letter of their name, one to make a gap and one to write their whole name built from the others in Logo.

Then recreate it using Functions in Python and the Turtle Module.

Rogue-Like Game

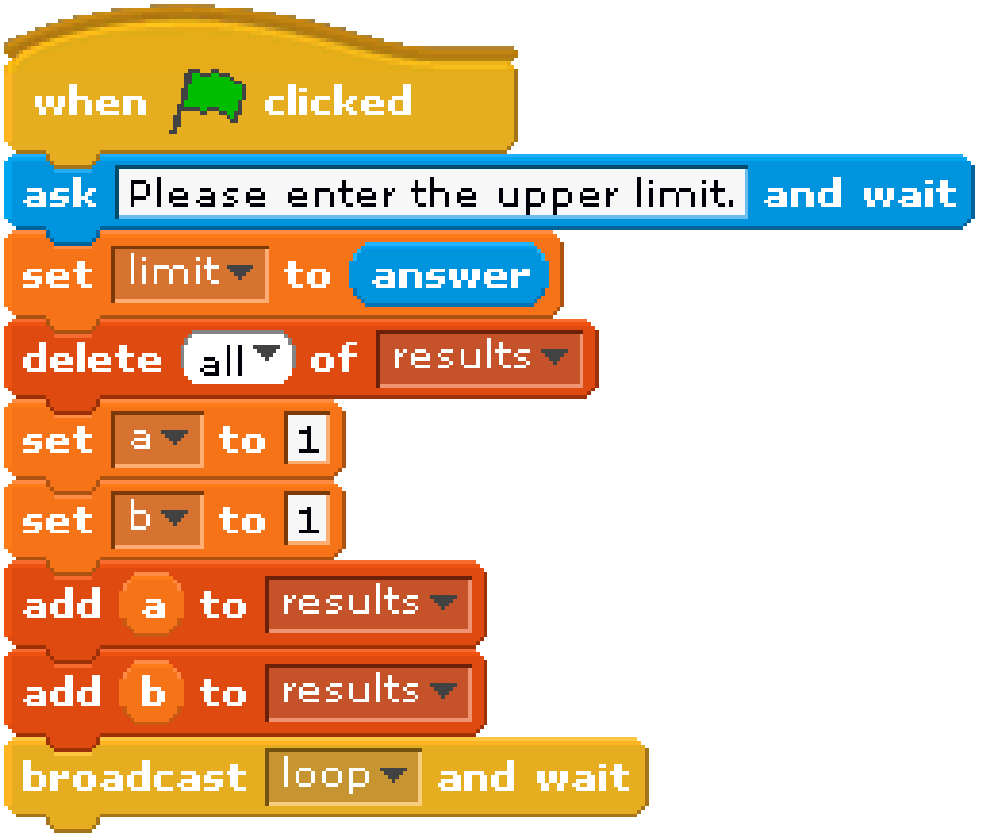
- In order to give them some experience of creating larger projects we work through creating a rogue-like game with them
- Follow similar steps to the GCSE Controlled Assessments – modelling what they need to do to create the project
- Iterative design process
- Introduces Testing and Pseudocode
- How to write up their project
- They build a version and I build one

Recursion

Creating a Fibonacci Generator

- Iterative Version in Scratch
- Almost Recursive Version in Scratch
- Iterative Version in Python
- Proper Recursive Version in Python
- Proper Recursion in Scratch 2.0 or Snap?

Scratch – Almost Recursive



Scratch 2.0 Version?
Snap 4.0 Version?

Any volunteers?

Python Versions

```python
#Iterative Version
def fibonacci(value):
    lastValue = 1
    currentValue = 1
    for x in range (2,  value):
        temp = currentValue
        currentValue = lastValue + currentValue
        lastValue = temp
    return currentValue

#Recursive Version
def recursiveFibonacci(value):
    if value == 1:
        return 1
    elif value == 2:
        return 1
    else:
        #Inefficient
        return recursiveFibonacci(value-1) + recursiveFibonacci(value-2)
```

Conclusions

- Start simple
- Always design it on paper first
- Can just do top level design – no details first
- Iterate
- Do it in Scratch (or Logo) first
- Recreate it in Python
- If you get stuck ask: CAS has a Forum – No question too simple, small or stupid
- StackOverflow
- Enjoy!

Contact

Dave Ames – CAS Website

dave.ames@computingatschool.org.uk

http://dave-ames.net

@davidames