How the Session Works

Outline
• Practical on arrival
• Talk 1
  • Reflect on practical
  • Clarify concepts
• Practical exercises at your own pace
• Talk 2:
  • Further concepts
  • Overall reflection
• Continue practical exercises at home

Getting Started
• Log-on
• Find portable Python on L:\ drive and start IDLE
• Go to https://scratch.mit.edu/
• Find resources on teachinglondoncomputing.org
  • Exercise sheet (and notes) – START NOW
  • Example programs
  • Slides
Transition from Scratch to Python using Turtle Graphics

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

• A first turtle program in two languages
• Discussion: barriers to textual programming
• Learning Programming Progressively
• The turtle language
• Translating between Scratch and Python
• Practical examples
• Summary
from turtle import *

# Added code starts here
pencolor('blue')
pensize(10)

forward(100)
right(90)
forward(100)
right(90)
forward(100)
right(90)
forward(100)
right(90)

# Added code ends here
done()
Discussion

• What are the challenges of learning textual programming?
Challenges of Text Programming

• Accuracy – easy to make mistakes
• Blank sheet problem – lack of starting point
• Motivation – not visual

• Solving a problem: decomposition
• Understanding programming concepts
  • Sequence, choice, repetition, state (variables)
• Debugging
Progress in Learning Programming

• KS1 onwards
  • Computers accept commands: algorithm
  • … turtle often used
  • Decomposition: sequences, choice and repetition

• Challenges
  • Learning Scratch versus learning programming

• Core concepts
  • Problem solving and debugging
  • Programming concepts
Core Programming Concepts

- Sequence: one instruction follows another
- State: variables hold values and can change
- Choice: alternative instructions
- Repetition: repeating instructions
- Input and output

- Values (expressions) versus statements
- Abstraction: procedures / functions
Turtle Language – I

• Role of turtle in Scratch and Python
  • Turtle is a little language inside a more general language

• Essential commands
  • Forward
  • Left
  • Right
  • Pen up
  • Pen down

General language concepts
• Sequence
• Repetition (bounded)
• Function abstraction
Turtle Language – II

- Use of co-ordinates and headings
- Get and set co-ordinates
- Get and set heading
- Also
  - Distance
  - Towards

General language concepts
- Variables
- Choice
Translating Scratch & Python

- Same core concepts
- Overlapping problems that can be easily solved
num = 0
sum = 0
while x <= 10:
    x = x + 1
    sum = sum + num
print('Sum ... is', sum)
Control Statements – Close

```python
for x in range(1, 10):
    print("Hello")

x = 0
while x != 100:
    forward(1)

if mouseDown():
    forward(10)
else:
    right(15)
```
Variables – Set and Change

- Later in scratch

Variables:
- Sprite
- Global

Value of the variables

```
Steps = 0
Steps = Steps + 1
```

Python uses name for both left and right hand side of assignment
# Programming Concepts

<table>
<thead>
<tr>
<th>Concept</th>
<th>Comparison</th>
</tr>
</thead>
<tbody>
<tr>
<td>Arithmetic operators</td>
<td>Very similar</td>
</tr>
<tr>
<td>Logical operators</td>
<td>Very similar</td>
</tr>
<tr>
<td>Selection</td>
<td>Very similar</td>
</tr>
<tr>
<td>Loops</td>
<td>Scratch has more forms</td>
</tr>
<tr>
<td>Variables and types</td>
<td>Scratch does not distinguish strings from numbers</td>
</tr>
<tr>
<td>Assignment</td>
<td>Clearer in Scratch</td>
</tr>
<tr>
<td>Input and output</td>
<td>Scratch equivalents for input / print</td>
</tr>
<tr>
<td>Broadcast</td>
<td>No direct equivalent: decomposition</td>
</tr>
<tr>
<td>Functions</td>
<td>Similar</td>
</tr>
<tr>
<td>Sprites</td>
<td>No direct equivalent</td>
</tr>
</tbody>
</table>
Introduce Assignment

• Exploit different syntax to emphasise that assignment is not equality

• Python

\[
Total = Total + \text{ItemCost} \times \text{Number}
\]

• Means the same as:
Practical Problems
### Equivalence

- Two programs can have same behaviour
  - Different forms of ‘if’ or ‘loop’
  - Logical equivalence
  - Repetition versus loops

- Redundant code
  - Code that makes no difference
  - Easy to include this in Scratch
Problem 1: If & Logic

- Two variables: 'name' and 'age'
- Which versions are the same?
Problem 2: Counting to 5

• Which are the same?
Summary

• Ideas for transferring from visual to textual programming
• Core programming concepts: make the correspondence explicit
• Problem solving, abstraction and decomposition: build on existing ideas