CPSC 301: Computing in the Life Sciences
Lecture Notes 3: Tracing, summarizing and writing Scratch code

Complete notes for 2017-01-12

Jessica Dawson
jqdawson@cs.ubc.ca
http://www.cs.ubc.ca/~jqdawson

University of British Columbia
Department of Computer Science

2016 W2
Admin – 2017-01-12

• **Waitlisted students**
  – sign in again today after class

• **Lab enrollment**
  – Email Jessica if you don’t have a lab yet and cannot fit Mon 5 – 7pm

• **Lab02 – Posted**
  – Make sure to complete the before lab parts BEFORE lab
  – Starting with lab 02 you will do pair programming in labs
  – HOWEVER, you both still need to complete the before lab
    • You can work together on it. But you won’t be able to contribute equally in lab if only one of you has done the preparation.

• **Next week: Getting started with python**
  – Textbook reading already posted
  – Slides and pre-class quiz up later today
Last time - Tracing

• **Recall:** Point is to pretend to be the computer: step through each line, keeping track of how that line changes the data
  – Need to make sure you focus on understanding exactly what the code will do (not what you intend it to do)
  – Important when trying to fix broken code
Summarizing Scratch Code

• Describes the action of the code fragment at a higher level than tracing in a form that humans understand
  – A good test of understanding: if you cannot understand the code, you are unlikely to be able to translate it into human language form

• The goal is to develop an overall picture of a piece of code
  – A practical purpose is to decide whether you can reuse that code, or whether you can modify it to suit your needs

• Even in Scratch, programs can be very long
  – When explaining, break them down into appropriate chunks
  – Size of chunk will depend on the degree of detail that is desired: maybe entire sprites, entire scripts, parts of a script, but rarely individual blocks
Summarizing Example: Dog Sprite

• Consider again the two scripts from the dog sprite in Scratch project “chasing the mouse” from class web site

• Summary of first script: After the flag is clicked, sprite will keep moving relatively slowly towards the mouse-pointer.

• Summary of second script: After the flag is clicked, sprite will keep barking for 1 second and then staying quiet for 1 second as long as it does not touch the mouse-pointer.
Summarizing Example: Cat Sprite

• Try a high-level summary of the Cat sprite in the Scratch project “chasing the mouse” from class web site

Summary:
When the space key is pressed, cat will keep moving by 20 steps to the right and 10 steps up until it touches the mouse pointer.

Summary:
When the space key is pressed, cat will keep pointing towards the mouse pointer every 3 seconds.
Summarizing Example: Fur Elise

• Consider the project Fur Elise from Scratch gallery (posted on the web site)
  – Summary of the whole project:
    Plays the right hand theme of the first stanza (verse, part,...) of Beethoven’s Fur Elise displaying the piano keys that play the notes.
  – Summary of the first script of the Piano sprite:
    – Plays the right hand theme of the first stanza (verse, part,...) of Beethoven’s Fur Elise.
  – Summary of the of the second script of the Piano sprite:
    – Displays the piano keys that play the right hand theme of the first stanza (verse, part,...) of Beethoven’s Fur Elise.
Summarizing Example: **Ping Pong**

- Another simple project from the Scratch gallery
- Summarize the action of each of these scripts:
  - First script:
    - When the simulation starts, put the ball at a particular position. (Note that direction is not set.) Stop the simulation if the ball ever touches the colour red.
  - Second:
    - For as long as the simulation is running, the ball will bounce if it hits an edge of the stage and will move at rate four in its current direction.
  - Third script:
    - For as long as the simulation is running: When the ball touches the paddle, make a water-drop sound and bounce the ball up with a small random angular perturbation of up to 20 degrees.
Writing Scratch Code

- When working with a new language, it is often easiest to start from something that works and modify it
  - Working from the existing code helps you learn by example
- This fact is particularly true for Scratch
  - The Scratch gallery gives you lots of examples of working code
  - The Scratch environment lends itself to experimentation
- We can view the existing code as an incorrect attempt to solve your task
  - It may (or may not) be correct for its original task
- So the modification of Scratch code to accomplish a different task is actually an example of fixing incorrect Scratch code
Summarizing and writing exercise

• Consider the following script for a calculator

• Group exercise:
  – Write a summary of the existing script
  – Write to code to improve the script to handle invalid operations
Summary

• Three basic programming skills are useful in any language for any task:
  • Code Tracing
    – Simulate the code and record the changes it performs on the data
    – Helps to find errors in the code
  • Code Summarization
    – Describe the code in regular human terms
    – Helps the programmer understand the code
  • Code Writing
    – We usually start from existing code and modify it
    – Tracing and summarization help here as well.