Computational thinking; Hello Python; Errors

COSC 101, 2018-01-24

Announcements
● For Friday
  ○ Read sections 2.1-2.7, 2.9
  ○ Complete reading questions

Outline
● Warm-up
● Computational thinking
● Programming
● Hello, Python!
● Errors

Warm-up
● What is an example of an algorithm you lookup and execute in real life?
  (An algorithm is a step by step list of instructions that solve a problem.)
  ○ Recipe
  ○ Driving directions

Computational thinking
● Computational thinking is the thought processes involved in formulating a problem and expressing its solution(s) in such a way that a computer—human or machine—can effectively carry out.
  -- Jeannette M. Wing
  ○ i.e., computational thinking is the thought process we follow to write an algorithm
● Example: an algorithm for brushing your teeth
● Decomposition --- break problem into pieces
  1. Wet the toothbrush
  2. Put toothpaste on the toothbrush
  3. Brush each side of your teeth for 30 seconds
  4. Spit
  5. Rinse
● Could a human execute this algorithm? --- Yes/maybe
● What about if they have never brushed their teeth before? --- Maybe/no
● Abstraction --- “hiding” some of the details of the solution
  ○ Useful for developing a solution
  ○ Need to provide underlying steps --- or rely on something else to provide the steps
● What parts of this algorithm are too abstract?
  ○ Wet the toothbrush --- With water? Vinegar?
  ○ Put toothpaste on the toothbrush --- On the handle? Bristles?
  ○ Brush each side of your teeth for 30 seconds --- What is a “side”? With what part of the toothbrush?
  ○ Spit --- On the floor? In the sink?
  ○ Rinse --- With water? Vinegar?
● Decompose these steps to make the algorithm unambiguous
   1. Turn on faucet
   2. Place toothbrush under water for 5 seconds
   3. Unscrew cap from tube of toothpaste
   4. Put 5mL of toothpaste on bristles
   5. Screw cap back on tube of toothpaste
   6. Open mouth
   7. Move toothbrush back and forth across teeth on upper-left side for 30 seconds
   Etc.

Computational thinking practice
1. Turn away from your partner so you cannot see their paper and draw something. Do not look at what your partner is drawing!
2. Write an algorithm explaining how to recreate the drawing for someone who has not seen it.
3. Exchange instructions with your partner. Follow your partner’s instructions to recreate their drawing.
4. Compare your drawings. How well does what you drew match up with what they originally drew? How can you improve your instructions?

Programming
● Programming --- representing algorithms in a notation (language) that can be followed by a computer
● Natural language is inherently ambiguous
   ○ A robot could not follow your teeth brushing (and drawing) algorithm
● Programming languages are unambiguous --- an instruction has exactly one meaning
● What are the basic types of instructions used by Python (and nearly every programming language)?
   ○ Receiving input --- e.g., from keyboard
   ○ Providing output --- e.g., display text on screen
   ○ Math and logic
   ○ Conditional execution
   ○ Repetition

Hello Python
● Two modes of programming in Python
   ○ Type instructions into interpreter --- instructions are executed as they are entered
   ○ Type instructions in a file --- instructions are executed when file is read by interpreter
● First program
      >>> print("Hello, Python!")
Hello, Python!
Errors

- Debugging --- process of correcting errors (or bugs) in your program
- A dead moth on a relay in one of the first computers caused a problem, and the name *bug* stuck

What are the three kinds of errors that can occur in a program? When are they detected?

- Syntax error --- a grammatical error
  - Detected when you enter an instruction into the interpreter
- Runtime error --- infeasible instructions
  - Detected when the interpreter
- Semantic error --- wrong instructions
  - Detected when a human examines the program’s output

What type of error exists in each of the following programs?

a. `print("six plus seven is...")
   6 - 7
   Semantic --- output is unexpected`

b. `print("Lucky number 13")`
   *Syntax --- missing parenthesis*

c. `13 / 0`
   *Runtime --- cannot divide by zero*

d. `PRINT("13 men who offered $13 and 13 prayers")`
   *Syntax --- Python is case-sensitive*