§1.8-§2.1: Software Abstractions and Control Structures

12 Sep 2006
CMPT14x
Dr. Sean Ho
Trinity Western University

- ***HW1 due today***

TRINITY WESTERN UNIVERSITY

A note about IDLE: subprocess

- You should see "=== RESTART ===" each time you press F5 or do Run→"Run Module"
  - In the default installation, the right-click context menu "Edit with IDLE" starts IDLE in a slightly different mode
- Details for those interested:
  - "-n" command-line option causes IDLE not to use a separate subprocess for each run
  - Using the "-n" option will cause problems for you later when creating your own modules
- Screen capture on the Macs: Alt-F14
  - Copies into clipboard: Ctrl-V to paste image

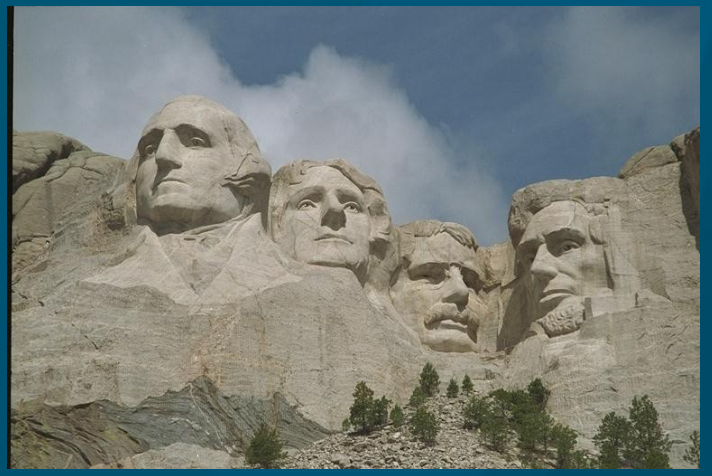
What's on for today (§1.8 - §2.1)

- Expressions and precedence
- Logical operators
- Hardware abstractions
- Software abstractions: levels of translation
- Control/structure abstractions
- Pseudocode
- Library functions

Logical operators

- Logical operators are operators on the bool type:
  - GodLovesMe = True
  - ILoveGod = False
- not: flips True to False and vice-versa
  - not GodLovesMe >>> False
- and: evaluates to True if both operands are True
  - GodLovesMe and ILoveGod >>> False
- or: evaluates to True if at least one operand is True
  - GodLovesMe or ILoveGod >>> True

Operator Precedence



- How would you evaluate this?
  - 5 + 4 * 2
  - (5 + 4) * 2 >>> 18: Addition first
  - 5 + (4 * 2) >>> 13: Multiplication first
- Precedence is a convention for which operators get evaluated first (higher precedence)
  - Usually multiplication has higher precedence than addition
- When in doubt, use parentheses!

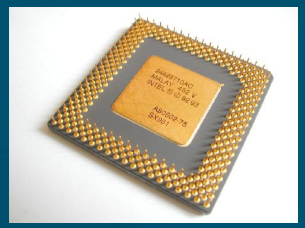
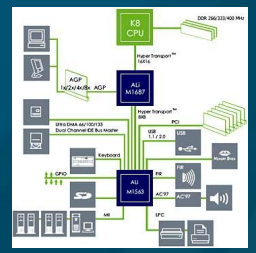
Expression compatibility

- 5 + True doesn't make sense: incompatible types
- What about 5(int) + 2.3(float)?
  - Works because the two types are expression compatible
- The "+" operator is overloaded:
  - It works for multiple types: both int and float
- It turns out that in Python, 5+True does evaluate:
  - 5+True >>> 6
    (interprets True as 1 and False as 0)

Hardware abstractions

- Generally, most computers have these basic hardware components:
  - Input
  - Memory
  - Processing
  - Control
  - Output




- Together with the software, the environment presented to the computer user by these is the virtual machine

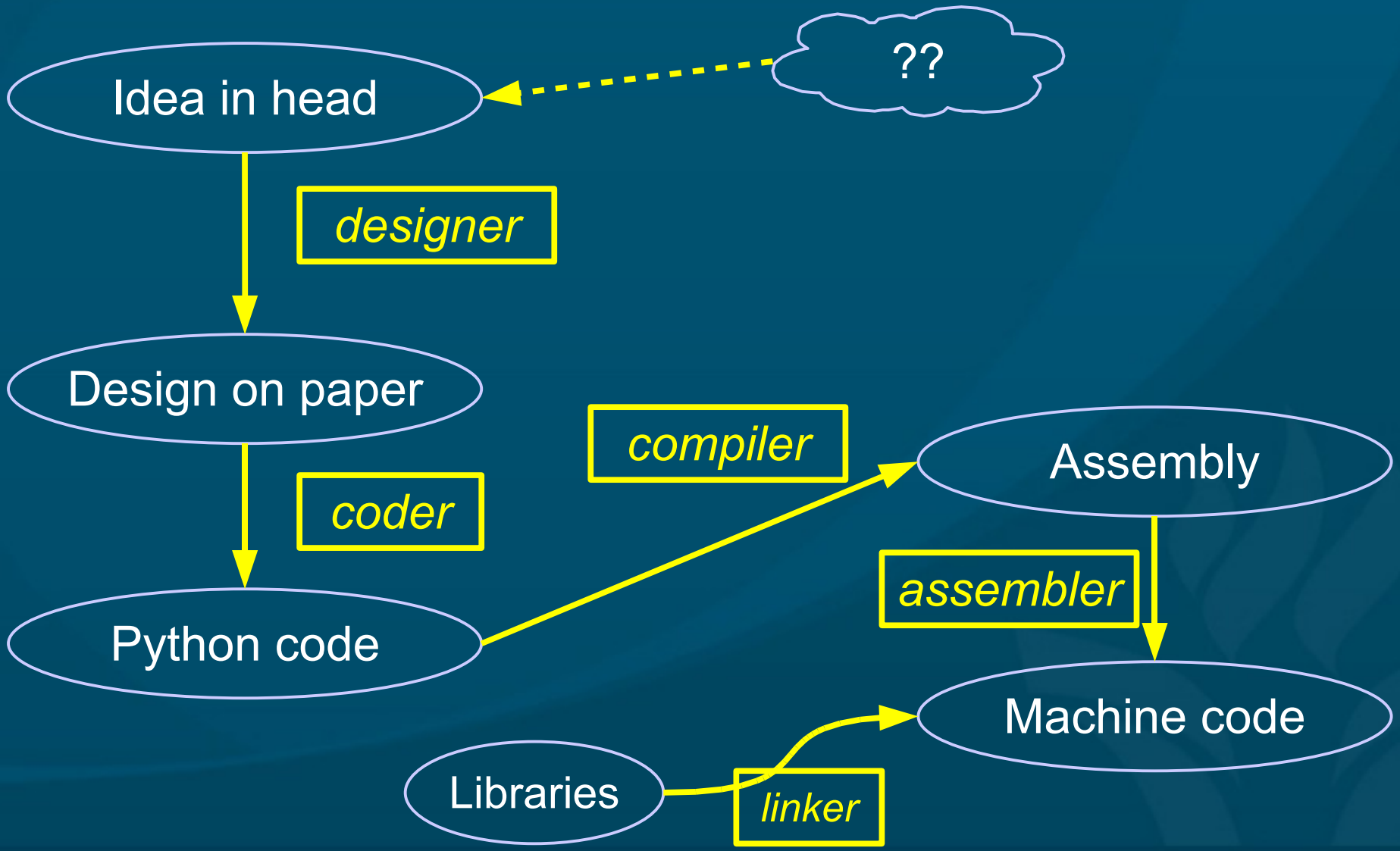
Software abstractions

- Instructions: basic commands to computer
  - e.g., ADD x and y and STORE the result in z
- Programming language: set of all available instructions
  - e.g., Python, C++, machine language

- Program: sequence of instructions
  - e.g., your “Hello World” program
- Software: package of one or more programs
  - e.g. Microsoft Word, Microsoft Office

- Operating system: software running the computer: provides environment for programmer
  - e.g., Windows XP, Mac OSX, Linux, etc.

Programming is translation



Control abstractions

- Sequence: first do this; then do that
- Selection (branch): IF ... THEN ... ELSE ...
- Repetition (loop): WHILE ... DO ....
- Composition (subroutine): call a function
- Parallelism: do all these at the same time

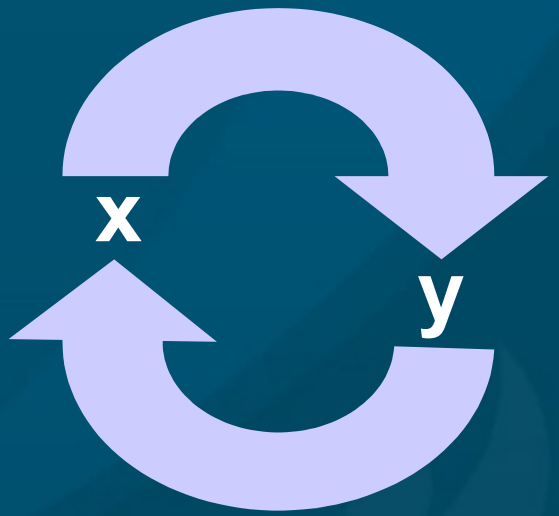
- These are the basic building blocks of program control and structure

Pseudocode

- Pseudocode is sketching out your design
  - General enough to not get tied up in details
  - Specific enough to translate into code
- Use the five control abstractions
- Usually several iterations of pseudocode, getting less abstract and closer to real code
- Don't worry about syntax; worry about semantics
  - Repetition can be done with WHILE ... DO ... or LOOP ... UNTIL ....:
  - Similar semantics; different syntax

Example pseudocode: swap

- Problem: swap the values of x and y
- Initial solution:
  - x <--- y
  - y <--- x
- Will this work?
- Try again:
  - temp <--- x
  - x <--- y
  - y <--- temp



Example pseudocode: add 1..20

- Problem: add the integers between 1 and 20
- Initial solution:
  - Initialize sum to 0
  - Initialize counter to 1
  - Repeat:
    - Add counter to sum
    - Add one to counter
  - Until counter = 20
- Will this work?

Example: add 1..20 (second try)

- Try again:
  - Initialize sum to 0
  - Initialize counter to 1
  - Repeat:
    - Add counter to sum
    - Add one to counter
  - Until counter = 21
- Alternate version:
  - Initialize sum to 0
  - Initialize counter to 1
  - While counter <21, repeat:
    - Add counter to sum
    - Add one to counter
- Same semantics, different syntax
- Top-of-loop test vs. bottom-of-loop test

Pseudocode: you try (group effort!)

- Problem: print the largest of a sequence of numbers
  -

Importing library functions

- Library functions are building blocks:
  - Tools that others wrote that you can use
- Functions are grouped into libraries:
  - If you want to use a pre-written function, you need to specify which library to import it from

```
import math
math.sqrt( 2 )          >>>1.4142135623730951
math.pow( 3, 5 )            >>>243.0
math.pi                     >>>3.1415926535897931
```

Review of today (1.8-2.1)

- Expressions and precedence
- Logical operators
- Five abstract components of hardware
- Software: instructions, languages, programs, operating system
- Designer -> coder -> compiler -> assembler + linker
- Five control/structure abstractions of programs
- Pseudocode
- Importing library functions

Writeups for Labs 1-2 *(L1 due next wk)*

- Full writeups required starting with Lab3
- Labs1-2 can have short writeup:
  - Design (10 marks)
    - Name, student#, CMPT14x, lab section, Lab#1, date
    - Statement of the problem
    - Discussion of solution strategy
  - Code (30 marks)
    - Name, etc. again in code header
    - Well-commented code, formatted and indented
  - Output (10 marks)
    - A couple runs with different input

TODO items

- Go to Neu9 computer lab:
  - Make sure you can login
  - Python/IDLE intro on course www (due Wed)
    - Nothing to hand in on this intro
- Homework due next class (Wed):
  - §1.11 # 25, 31, 40
- Reading: through §2.2 for Wed
- Lab1 due next week MTW (in lab section)
- Remember your quiet time journals