1.8-2.1: Software Abstractions and Control Structures

devo

12 Sep 2005
CMPT14x
Dr. Sean Ho
Trinity Western University

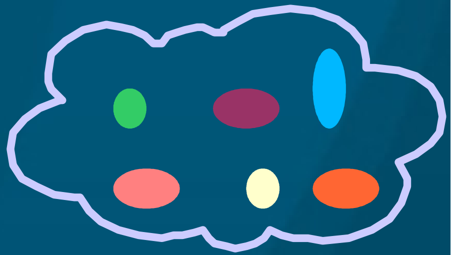
Reminder: journals in folder up front

TRINITY WESTERN UNIVERSITY

Announcements

- Suggested further reading:
  Fred Brooks, “The Mythical Man-Month”, 1975, 1995 (20th anniv. ed.), Addison-Wesley.
- Quiz today (after review)

Review from 1.5-1.7



- Atomic vs. compound data (examples?)
- Data types (examples?)
  - What's the difference: 5, 5.0, '5', “5”, (5), {5}
- Operators, operands, ADTs, implementations
- Variables vs. constants
- Logical operators: NOT, AND, OR
- Operator precedence
- Expression compatibility (what types?)

= NOT *
AND / < -
+ OR

Quiz ch1

- Get out a blank sheet of paper
- In the top right corner, write
  - Your name
  - Student #
  - CMPT14x
  - Quiz 1
  - Today's date (12 Sep 2005)
- Copy the quiz questions onto your sheet and provide short answers as best you can
- Closed book, closed notes, closed laptops/calcs

Quiz ch1 (4 questions, 20 marks, 10 minutes)

- Copy this sentence and fill in the blanks:
  - "Computers are t____, and computer scientists are t_________."
- What are the five steps of top-down problem solving?
  - (it's okay if you don't get the exact words, but write the concepts)
- What's the difference between 3, 3.0, and "3.0"?
- Write down the three logical operators and evaluate them on your choice of TRUE and FALSE operands

Quiz chapter 1: solutions (#1-2)

- "Computers are tools, and computer scientists are toolsmiths." (2+2)
- Five steps of top-down problem solving: (5)
  - Write everything down
  - Apprehend the problem
  - Design a solution
  - Execute your plan
  - Scrutinize the results

Quiz chapter 1: solutions (#3-4)

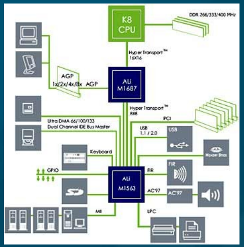
- 3, 3.0, “3.0”: difference is type: (2)
  - 3 is cardinal type (or integer) (1)
  - 3.0 is real type (aka float) (1)
  - “3.0” is string type (1)
- Three logical operators: (2+2+2)
  - NOT: e.g., NOT TRUE = FALSE
  - AND: e.g., TRUE AND FALSE = FALSE
  - OR: e.g., TRUE OR FALSE = TRUE

What's on for today (1.8-2.1)

- Hardware abstractions
- Software abstractions: levels of translation
- Control/structure abstractions
- Pseudocode
- Syntax vs. semantics
- Debugging

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

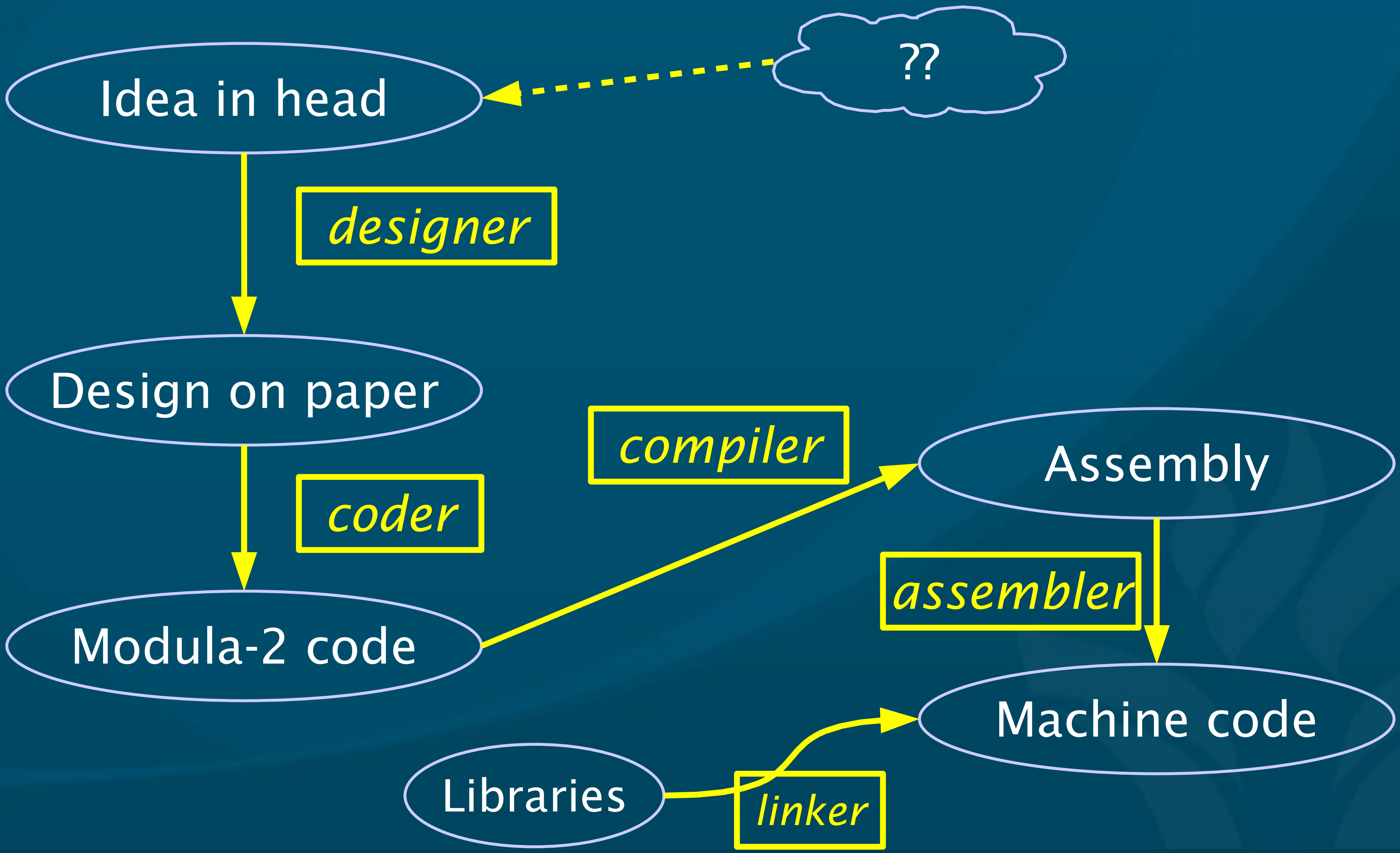
Software abstractions

- 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., Modula-2, C, machine language

M2

- 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

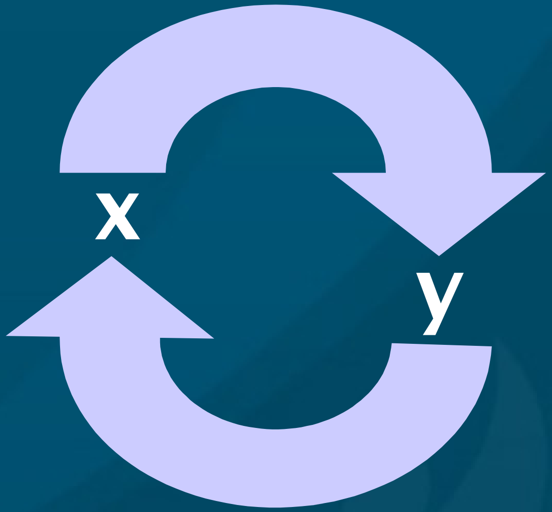
Example pseudocode: swap

- 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

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

Bugs and debugging

- Project stays “90% done” for 90% of the time
- Debugging takes up most of your time; allocate time for it!

- Spend a little more time on design and you'll save a lot of time debugging
- Syntax errors are easy to catch (compiler helps)
- Semantic (logical) errors come from poor design:
  - Much harder to catch, let alone fix!

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
-
  ```
  MODULE HelloWorld;

  FROM StextIO IMPORT
      WriteString;

  BEGIN
      WriteString (“Hello World!”);

  END HelloWorld.
  ```

Review of today (1.8-2.1)

- Five abstract components of hardware
- Software: instructions, languages, programs, operating system
- Designer -> coder -> compiler -> assembler + linker
- Five control/structure abstractions of programs
- Pseudocode
- Syntax vs. semantics
- 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
  - Stonybrook intro on course www (due Wed)
    - Nothing to hand in on Stonybrook 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