§2.2: The Anatomy of an Infant Program

devo

14 Sep 2005
CMPT14x
Dr. Sean Ho
Trinity Western University

Reminders:

1) journals in folder

2) HW in folders by lab section

Announcements

- Quiz ch1 and journals handed back today
  - NOT FALSE = TRUE

-

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

Review from 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

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.
```

What's on for today (2.2)

- Components of a baby Modula-2 program
- Modules
- Reserved words
- Library tools
- Identifiers
- Strings, quoting, newlines
- Structure of a program module

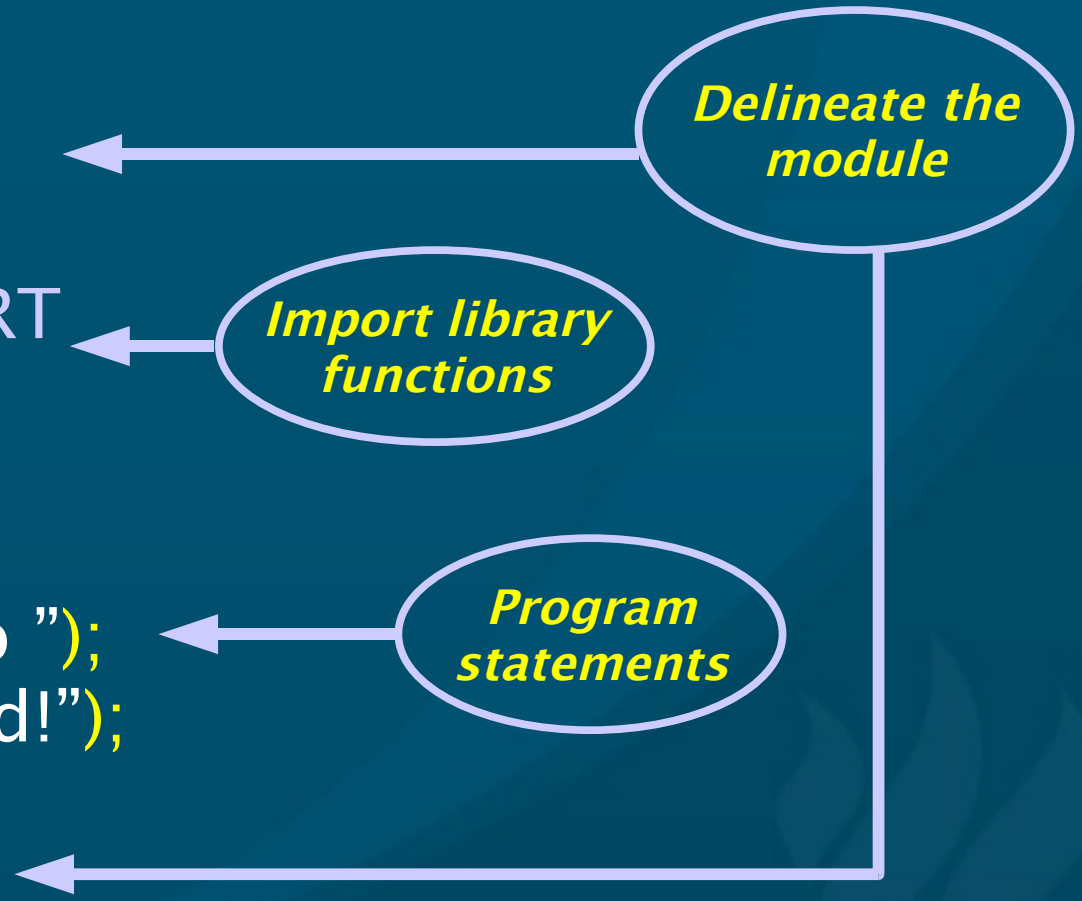
Components of "Hello World"

- MODULE HelloWorld;

  FROM STextIO IMPORT
      WriteString;

  BEGIN
      WriteString ("Hello ");
      WriteString ("World!");

  END HelloWorld.



- Outputs: "Hello World!"

Modules

- Yep, pretty important to ***Modula***–2
- A module is a container holding
  - items and information
  - constituting all or part of an executable program
- HelloWorld is a module that is a complete executable program
- STextIO is a module from which we imported the WriteString function
- WriteString is not a module but a function within a module

Reserved words

- You can name your modules, functions, and variables almost anything you want, except
- Reserved words: special words or markers used to outline the structure of a program
- All uppercase:
  - MODULE, FROM, IMPORT, BEGIN, END ...
  - See Appendix 1 for complete list

Modula-2 Standard Library Tools

- Library functions provided with every standard Modula-2 implementation
- You still have to IMPORT them, though
- Our HelloWorld program used the WriteString standard library tool from the STextIO library

- "The nice thing about standards is there are so many to choose from"
- The names for STextIO and WriteString may be different in different Modula-2 packages
  - it's the same with every language

Identifiers

- Identifiers are names for stuff: e.g.,
  - Modules (“HelloWorld”)
  - Libraries (“STextIO”)
  - Functions (“WriteString”)
  - Variables (“x”)
- Identifiers are sequences of
  - non-blank letters or digits
  - Must start with a letter
- OK: GreatGooglyMoogly, x, My21stBirthday
- Not OK: “hi ya”, h@Xz0r, 21stBirthday
- Case sensitive! WriteString ≠ writestring

Strings and quoting

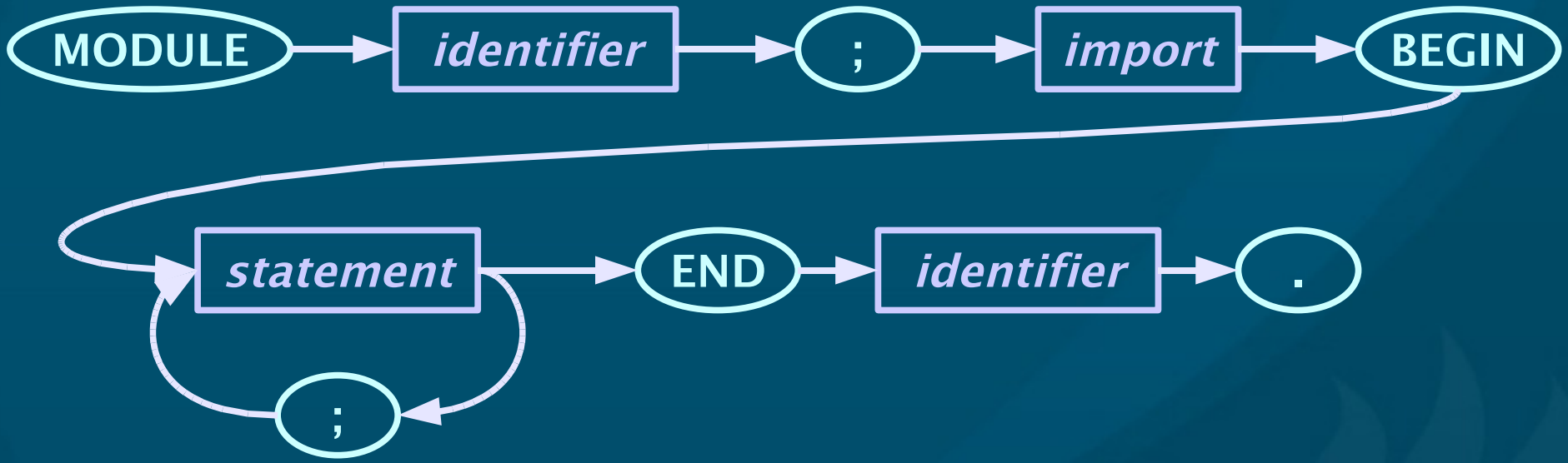
- Strings in Modula-2 can be in either
  - Single quotes: 'In the beginning was the Word'
  - Double quotes: “and the Word was with God”
- What if you want a quote mark in your string?
  - “It is I; don't be afraid”
  - 'Jesus said, “I am the way, and the truth, and the life.”'
- Can't have a newline (carriage return) in string:
  - “This is an illegal
    string because it has a newline in it.”

Splitting up strings; WriteLn

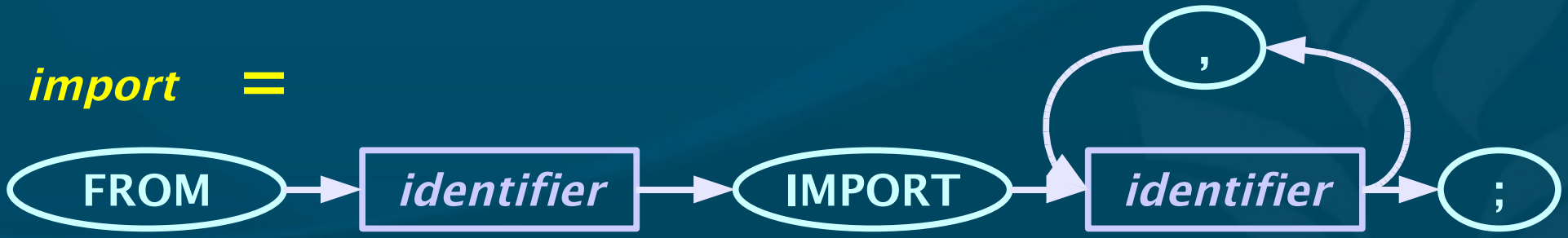
- WriteString (“Therefore go and”);
  WriteString (“make disciples”);
  - Therefore go andmake disciples
- WriteString (“Therefore go and ”);
  WriteString (“make disciples”);
  - Therefore go and make disciples
- WriteString (“Therefore go and”);
  WriteLn;
  WriteString (“make disciples”);
  - Therefore go and
    make disciples

Structure of a program module

program module =



import =



Review of today (2.2)

- Components of a baby Modula-2 program
- Modules
- Reserved words
- Library tools (what are some we know already?)
- Identifiers (what are some legit examples?)
- Strings, quoting, newlines
- Structure of a program module (railroad diagram)

TODO items

- Stonybrook intro today (nothing to hand in)
- Homework due Friday:
  - §1.11 # 35
- Reading: through §2.4 for Thu; §2.5 for Fri
- Quiz next Mon
- Lab 1 due next MTW in lab section