Ch1-8 Review

- *Quiz ch7-8 today*

23 Oct 2006
CMPT14x
Dr. Sean Ho
Trinity Western University

TRINITY WESTERN UNIVERSITY

Quiz06 ch7-8 (10 minutes)

- In C, why should you always allocate strings (arrays of char) to be at least one char longer than the longest string you'll need to store? [2]
  - What could happen if you don't? [2]
- Contrast a header file with an implementation file [5]
- Convert 1001 1011 from binary to both hexadecimal and octal (use Python form). [5]
- If chr(0101) == 'A', what is chr(0112)? [3]
- Express 4Mb/sec in bytes/sec. *(binary, not SI)* [3]
  - (you may express your answer in powers of 2)

Quiz06 ch7-8: answers #1(a), (b)

- In C, why should you always allocate strings (arrays of char) to be at least one char longer than the longest string you'll need to store?
  - Store the null character terminating the string
- What could happen if you don't?
  - Reading from the string might not stop at the end of the array; we may keep reading into junk memory elsewhere.

Quiz06 ch7-8: answers #2-5

- Contrast a header file with an implementation file
  - Header: public interface, no function bodies
  - Implementation: contains bodies of functions
- Convert 1001 1011 from binary to both hexadecimal and octal (use Python form).
  - Hex: 0x9B; oct: 0233
- If chr(0101) == 'A', what is chr(0112)?
  - 'J'
- Express 4Mb/sec in bytes/sec. *(binary, not SI)*
  - 2**19 bytes/sec

Review of §9.0-9.6

- Sets
  - Membership
  - Union
  - Intersection
  - Difference
  - Symmetric difference
- Implementing sets in Python
- Bitsets

Addendum: set() in Python

- It turns out Python does have a built-in set() type:
  - Use set(seq) to create a set:
    - vowels = set(['a', 'e', 'i', 'o', 'u']) # or just set('aeiou')
    - myName = set(['s', 'e', 'a', 'n'])
  - Set operations:
    - vowels | myName # union
    - vowels & myName # intersection
    - vowels – myName # set difference
    - vowels ^ myName # symmetric set difference
    - if vowels <= myName: # subset
  - **Ref: http://docs.python.org/lib/types-set.html**

Today: Chapters 1-8 Review

- Ch1: Problem-solving
- Ch2: Your first program
- Ch3: Program structure
- Ch4: Procedures/functions
- Ch5: Arrays/lists
- Ch6: Library modules
- Ch7: Applications
- Ch8: Number bases and memory/storage

Ch1: Problem solving

- Computing scientists as toolsmiths
- Top-down vs. bottom-up; WADES
- Client --> Designer --> Implementer
  - Requirements doc, Design spec, Code
- Design, pseudocode, documentation
- Abstract data types
  - Atomic vs. compound
  - What's the difference: 5, 5.0, '5', (5), {5}
- 5 hardware abstractions, 5 control/flow abstractions

Ch2: A basic Python program

- Components of a baby Python program
- Literals, identifiers and reserved words (examples?)
- Strings, quoting, newlines
- Statically-typed vs. dynamically-typed
- Declaring and initializing variables
- Keyboard input: input(), raw_input()
- Expressions, operators, and precedence rules
- Formatted output: %d, %f, %s

Ch3: Basic Program Structure

- Statement sequences
- Selection (if, else, elif)
- Repetition/loops (while, for)
  - Top-of-loop vs. bottom-of-loop testing
  - Sentinel variables
  - continue, break, else
- Sequence concatenation (+) and repetition (*)
    - Works on strings, lists, tuples

Ch4: Functions

- Procedures (functions, subroutines)
  - No parameters
  - With parameters
  - Formal vs. actual parameters
  - Scope
  - Global variables (why not to use them)
  - Call-by-value vs call-by-reference
    - Python is call-by-object, which is like call-by-value for immutable types and call-by-reference for mutable types

Ch5 (+Py ch8): Arrays and Lists

- Call stack, backtrace
- Python lists vs. M2/C arrays
- Lists as function parameters
- Multidimensional arrays/lists
- Python-specific list operations
  - Membership (in)
  - Concatenate (+), repeat (*)
  - Delete (del), slice ([s:e])
  - Aliasing vs. copying lists

Python type hierarchy (partial)

- Atomic types
  - Numbers
    - Integers (int, long, bool): 5, 500000L, True
    - Reals (float) (only double-precision): 5.0
    - Complex numbers (complex): 5+2j
- Container (aggregate) types
  - Immutable sequences
    - Strings (str): "Hello"
    - Tuples (tuple): (2, 5.0, "hi")
  - Mutable sequences
    - Lists (list): [2, 5.0, "hi"]
  - Mappings
    - Dictionaries (dict): {"apple": 5, "orange": 8}

Ch6: Standard I/O and Libraries

- Working with files: file objects, open(), close()
  - Input: read(), readline(), readlines()
  - Output: write(), flush()
  - The file position pointer: seek(), tell()
- Standard I/O channels: sys.stdin, stdout, stderr
- Python standard math library

- Libraries: interface (DEF) vs implementation (IMP)
- Accessor (set/get) functions

Ch7: Applications

- Null-terminated strings; lexical sorting
- fractions.py ADT library:
  - Set/get functions to hide tuple implementation
- substitution.py cipher library:
  - How it works, encode/decode
- pseudorandom.py RNG library:
  - Seed, iterative process
    - (Understand concepts enough to code it)
  - Testing via histograms

Ch8: Number bases and storage

- Number bases:
  - Binary, hexadecimal (0xBEEF), octal (0115)
- Bitwise operators: &, |, ^, <<, >>
- Units of measure of memory:
  - Bits, nibbles, bytes, words, pages
- Units of measure for hard disks:
  - C/H/S geometry
- SI units vs binary units, KB vs. Kb, etc.

TODO items

- Lab06 due this week: ch7 choose one:
  - #22: word search
  - #32: pseudorandom plot (hint: try 'turtle')
  - #37: matrix library
  - #43: secure encryption
- CMPT140 Final ch1-8 this Wed-Thu 25-26Oct
  - Everyone attend Thu class, even 141/143!
- Register for CMPT145 if you haven't already!