Ch9-10, Py 10-14 Review

- *HW10 due today*

20 Nov 2006
CMPT14x
Dr. Sean Ho
Trinity Western University

TRINITY WESTERN UNIVERSITY

Review last time (12.1-12.5)

- Pointers (in Modula-2 and C)
  - Creating pointers, dereferencing pointers
  - Assignment compatibility
  - Pointer arithmetic
  - NIL (in C: NULL)
- Static vs. dynamic allocation of memory
  - Activation records
  - Stack, stack pointer
- Dynamic variables: NEW(), DISPOSE()

Static vs. dynamic memory

- Static variables are allocated at the beginning of the program run
  - Their size in memory is fixed at compile-time
  - Variables named in declaration section
- Dynamic variables are allocated during the running of a program
  - May also be deallocated during program
  - Size need not be predetermined
  - Reference them via pointers

Dynamic variables

- You can make your own dynamically allocated variables, using NEW() and DISPOSE():

```
VAR
    applePtr : POINTER TO REAL;
BEGIN
    NEW (applePtr);
```

  - Allocates memory for a REAL, and stores the address in applePtr

```
    DISPOSE (applePtr);
```

  - Deallocates the memory, and sets applePtr to NIL
- Dynamic variables are in the heap:
  - Open space for program to allocate/deallocate
- If heap is full, NEW sets pointer to NIL

A note about endianness

- Recall: CPU works on data one word at a time
  - 32-bit CPU: 1word = 4bytes
- 1 CARDINAL on a 32-bit machine takes up 1 word
  - 63_{10} = 00....0111111_{2} = 00 00 00 $3F_{16}$
- But what order are the bytes within a word?
  - Big-endian (big end first): 00 00 00 3F
  - Little-endian (little end first): 3F 00 00 00
- Different CPUs choose different endianness
  - => byte-ordering “holy wars”

Big-endian vs. little-endian

- Big-endian (MSB: most significant byte first)
  - How we write numbers: 4,902
  - Can sort numbers lexicographically like strings
  - CPUs: Sun Sparc, IBM mainframes, SGI MIPS/IRIX, most PowerPC
  - "Network byte order" for IP (Internet)
- Little-endian (LSB: least significant byte first)
  - How we do arithmetic: 236 + 105 (carry)
  - CPUs: Intel x86, AMD, IA64/Linux
- No "one true way", just be aware + byte-swap

What's on for today: review

- Lectures #25-37:

- Sets (M2 ch9)
- Dictionaries (Py ch10)
- Object-oriented programming (Py ch12-14)
- Exceptions (Py tut 8)
- Namespaces and scope (Py tut 9)

Review: Sets (§9.0-9.6)

- Set Theory
  - Membership
  - Union
  - Intersection
  - Difference
  - Symmetric difference
- Sets vs Python lists
    - Don't need to know Python set type
- Bitsets

Review: Dictionaries (Py ch10)

- Dictionaries
  - Keys and values
  - Basic dictionary methods:
    - .keys(), .values(), .items()
  - Iterating through dictionaries
  - Other dictionary methods:
    - len(), del, in, .get(), .copy()
  - Application: hinting
    - Fibonacci example

Review: OO (Py ch12)

- Using Python classes to make records
  - Attributes
    - use instance variables, not class variables
- Objects are instances of classes, created by constructor methods
- alias vs. shallow copy vs. deep copy
- Customizations: __init__, __str__, __mul__, etc.
- Default parameters

Review: Exceptions (Py tut 8)

- Exceptions:
  - Handling
  - Raising
  - else
  - finally
  - User-defined exceptions
  - Passing auxiliary data with an exception

Review: Namespaces (Py tut §9.2)

- Namespaces and Scope
  - Purpose: avoid name conflicts
  - Default (built-in) namespace
  - Global namespace for each file/module
  - Local namespace:
    - Function invocation
    - Class definition
  - Namespaces outside local are read-only:
    - New names (assignment, import) add to local
    - Modifying makes local copy (except global cmd)

Today: review

- Sets (M2 ch9)
- Dictionaries (Py ch10)
- Object-oriented programming (Py ch12-14)
- Exceptions (Py tut 8)
- Namespaces and scope (Py tut 9)

TODO

- Midterm Wed:
  - M2 chs9-10
  - Py ch10-14
- Lab09 due this week:
  - Complex number library
- Lab10 due next week:
  - Implement one of your old labs 2-7 in M2
  - Full lab-writeup (may reuse old writeup)