Exam 2 Review

CS 3358
Summer II 2013

Jill Seaman

Exam 2

- Wednesday, July 31, 12:00pm to 1:40pm
- Derr 241 (here)
- Closed book, closed notes, clean desk
- 20% of your final grade
- I recommend using a pencil (and eraser)
- All writing will be done on the test paper I will hand out.
- No calculators.

Exam Format

- 100 points total
  - Writing programs/functions/code (about 50-60%)
  - Multiple choice
  - Fill-in-the-blank/short answer
  - Tracing code (what is the output), tracing (demonstrating) stack and queue operations, sorts and sort operations
  - Finding errors in code (recursive functions)

Templates

- Why? What are they for?
  - Type independence, generic programming
- Templated Functions
- Templated Classes
  - Everything goes in the .h file
- Be familiar with the examples
  - swap, MemoryCell, (my) vector
- Be prepared to write code, convert existing function or class to template

Stack ADT

- Know the operations, how they work
  - O(1): push, pop, isFull, isEmpty
  - makeEmpty
- Be able to implement using an array or (singly) linked list or vector or . . .
- Be able to use a stack to solve a problem
- Be familiar with the sample code:
  - IntStack: the static stack class in the notes
  - stack_3358_LL.h: the linked-list implementation on the website
- Array vs Linked List implementations

Queue ADT

- Know the operations, how they work
  - O(1): enqueue, dequeue, isFull, isEmpty
  - makeEmpty
- Be able to implement using an array (with wraparound) or singly linked list or vector or . . .
- Be able to use a queue to solve a problem
- Be familiar with the sample code:
  - IntQueue: the static queue class in the notes
  - queue_3358_LL.h: the linked-list implementation on the website
- Array vs Linked List implementations

Recursion

- How to write recursive functions
  - Base case
  - Recursive case (smaller caller)
- Recursion over non-negative ints and lists
  - arrays, vectors, linked list, List_3358, substr
- Know what's wrong with recursive Fibonacci
- Binary Search: understand recursive version
- You will be asked to write one or two recursive functions.

Sorting

- Understand the different sorts:
  - $O(N^2)$: selection, insertion, bubble
  - O(N log N): merge sort, quicksort
- Know the algorithms really well
  - Will not have to write code for an algorithm
  - May be asked to give descriptions in English (or pseudocode). (see slides 6, 17, 25, 32, 43).
  - May be asked to show steps in the process (show result of a pass, or a merge, or a partitioning).
- Be familiar with runtime analyses and issues

Objectives covered

- Describe the values and operations of Stacks+Queues
- Perform (demonstrate) each of the Stack+Queue ADT operations given an instance of the ADT.
- Implement the Stack and Queue ADTs in C++ using both dynamic and static underlying types, (including singly and doubly linked lists) and using templates (generic programming)
- Summarize the advantages and disadvantages of static vs dynamic implementations of stacks and queues (array vs linked lists)
- Write modular programs in C++ that use stacks/queues
- State a definition of recursion
- List and describe common applications of recursion (problems commonly solved with recursion: gcd, fib)
- Read and write recursive functions in C++
- Describe elementary algorithms for sorting and searching, including linear/binary search, selection/insertion/bubble/merge and quicksort.
- Perform (demonstrate) each of the elementary algorithms (sorting, searching)
- Analyze algorithms, including sorting, searching, and implementations of ADT operations, for efficiency
  - best/worst/avg cases

Exercises: stacks+queues

Given the ADT for the Stack_3358 at the end of the exam, implement the push, pop, isEmpty and isFull functions.

The class declaration would either:
a) include the private member variables or else
b) the question would state which implementation to use and you would provide the private member variables

Given a set of operations over an empty stack (or queue) show the contents of the stack (or queue) after the operations are completed (see Quiz#3).

Exercises: Sorting

Given the following array, what would be the contents
a) after the 4th iteration of the insertion sort?
b) after the 4th iteration of the selection sort?
c) after one iteration of the bubble sort?

```
3  7  2  12 56  1  42  9
```

Given the array above, sort it using the merge sort algorithm: show a trace of the recursive calls and the results of the merge at each step.

Exercises: Sorting

What are the main steps of the merge sort (or quicksort)? Include the base case and recursive case.

Given the following two arrays, show the contents of the temp array after 3 passes of the merge algorithm. Show the final positions of i and j.

```
 1 9 42 56   2 3  7 12
```

Given the following array, show the contents after a valid partitioning using the first element as the pivot point.

```
7  3  2  12 56  1  42  9
```

How to Study

- Review the slides for each lecture
  - understand all the concepts
- Use the book to help understand the slides
  - there will be no questions over material (or code) that is in the book but not on the slides
- Understand the code in the demo(s)
- Understand the homework assignment solutions
  - rewrite yours so it works
  - Practice, practice, practice
- Get some sleep