ICS 240 – Competency List

Object-oriented programming: encapsulation, polymorphism, inheritance Chapter 11

Can develop a subclass from a superclass through inheritance

Can invoke the superclass's constructors and methods using the super keyword

Can override instance methods in the subclass

Explore polymorphism and dynamic binding

Can explain the difference between overriding and overloading

Can describe casting and explain why explicit downcasting is necessary

Can prevent class extending and method overriding by using the final modifier

Abstract Data Types Chapters 9 and 10

Can describe objects and classes, and use classes to model objects

Can use UML notations to describe classes and objects

Can design classes that follow the class – design guidelines

-use privacy modifiers to encapsulate data fields to make classes easy to maintain

-define constructor(s) to assign default values for an object's data fields

-define private data fields with appropriate get and set methods

-override the toString() to produce readable output

-override the equals method to logically determine equality

Linked Lists: static implementation, dynamic implementation, and doubly linked Chapter 24

Can design and implement a linked list

Stacks: static and dynamic implementation Chapters 24 and 20

Can design and implement a stack using an array

Can design and implement a stack using a linked list

Queues: static and dynamic implementation, circular, and priority Chapters 24 and 20

Can design and implement a queue using a circular array

Can design and implement a queue using a linked list

Can design and implement a priority queue using a heap

Generic Objects Chapter 19

Can describe the benefits of generics

Can use generic classes and interfaces

Can define generic classes and interfaces

Can define and use generic methods and bounded generic types

Can describe the benefits and pitfalls of using raw types

Recursion Chapter 18

Can describe what a recursive method is and the benefits of using recursion

Can develop recursive methods for recursive mathematical functions

Can explain how recursive method calls are handled in a call stack

Can solve problems using recursion

Can use an overloaded helper method to derive a recursive method

Can implement selection sort using recursion

Can implement binary search using recursion

Explore the relationship and difference between recursion and iteration

Can explain the difference between tail recursion and head recursion

Algorithm Analysis Chapter 22

Can estimate algorithm efficiency using the Big-O notation

Can describe common growth functions (constant, logarithmic, quadratic, cubic, exponential)

Sorting: quicksort, merge sort, heap sort, shell sort, and radix sort Chapter 23

Can explain time efficiency of various sort algorithms

Can implement and analyze quicksort

Can implement and analyze merge sort

Can implement and analyze heap sort

Can implement and analyze shell sort

Can implement and analyze radix sort

Hashing with collision strategies Chapter 27

Can explain what hashing is and what it is used for

Can obtain the hash code for an object and design a hash function to map a key to an index

Can handle collisions using open addressing

Can explain the differences among linear probing, quadratic probing, and double hashing

Can handle collisions using separate chaining

Trees Chapter 25

Can design and implement a binary search tree

Traversals Chapter 25

Can implement an in-order traversal of a binary tree

Can implement a pre-order traversal of a binary tree

Can implement a post-order traversal of a binary tree

Heaps Chapter 23

Can design and implement a heap