COMP2911 Send Help

What’s in the exam?

* Object Orientated Design
  + Refer to assignment 1
  + Aspects of the design process, e.g. CRC cards, UML diagrams
* Programming by Contract
  + Pre-post-conditions
  + Invariants & their interaction with inheritance
  + What is a contract?
* Generic Types and Polymorphism
  + Similar to lab with Set<E> or Graph<E>
  + Bit of assignment 2 maybe?
* Design Patterns
  + Iterator, Observer, Decorator, Strategy, Composite
  + <http://www.tutorialspoint.com/design_pattern/index.htm>
* Concurrency

No: Searching (A\*, Djikstra’s, Heuristics), agile development (scrum, sprints, development process), GUI

Object Oriented Design

Design Phase:

* Identify the classes, responsibilities of these classes and relationships between these classes
* The choice of data structures isn’t in the design phase – it’s in the implementation phase
* End product: description of classes + their responsibilities, diagrams of relationships between classes and diagrams of important scenarios

Objects:

* State: collection of all information held by the object – affects the behaviour of an object (e.g. an empty mailbox may return an error when asked to return messages, a full one would reject additional messages)
* Behaviour: operations or methods that an object supports
* Identity: there could perchance be two mailboxes with the same state + behaviour but they are separate objects

Identifying Classes:

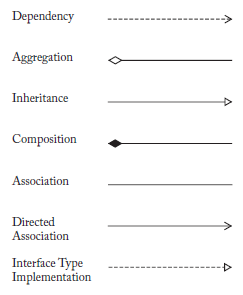
* Rule of thumb: look for nouns in the specification
* Classes that are required to do the work e.g. storage of messages in a mailbox ie. Class MessageQueue
* Agents: Operations e.g. Scanner
* Event and Transaction classes: model records of activities to identify what has been done and what needs to be done e.g. MouseEvent
* User and role classes: for systems often used by more than one person or one person needs to perform distinct tasks
* System classes: to perform initialisation and shutdown

Identifying Responsibilities

* Look for verbs e.g. recorded, played, deleted… users log in
* A responsibility must be assigned to one class
* A mid-level abstraction class shouldn’t deal with processing keystrokes (low levelled) or initialising the system (high levelled)

Relationships between classes

* Dependency (“Uses”), Aggregation (“Has”), Inheritance (“Is”)
* Dependency
  + A class is dependent on another class if it manipulates objects of the other class in any way e.g. Mailbox **uses** Message class
  + Asymmetric because a Message doesn’t depend on Mailbox but Mailbox depends on Message
  + Minimise the number of dependencies because changes in one can force changes in others
* Aggregation
  + A special case of dependency (stores multiple instances rather than just manipulates)
  + If objects of one class contains objects of other classes over a period of time e.g. MessageQueue has Message objects, and we say MessageQueue aggregates the Message class
  + Usually corresponds to fields in a class
    - Primitive types such as number or date are considered to be **attributes** instead
* Inheritance
  + If objects of a class are special cases of objects of the other class
  + i.e. Superclasses and subclasses

****