

# CS 249 MidTerm Exam - Closed Book In-Class Exam

Time allotted: 75 minutes February 12, 2002

Answer all questions. Each question is worth the points as indicated. You can answer in point form if you prefer when English is called for. You can use sketch code that is not necessarily complete AS LONG AS IT CLEARLY INDICATES you understand the approach or technique we are after. We are not looking for a full essay on each question, but rather a short concise set of points or sketch code that responds to the question and indicates you understand the point being explored.

1. (10 Points) For each aspect of SOS, i.e.

- (a) source code representation
- (b) outside-in development
- (c) short-cycle development

state what you consider the most important aspect of object-oriented programming that supports it, and briefly describe why.

2. (10 Points) Suppose you have an interface class

```
class Airplane : public Ptr<NamedInterface> {
public:
    virtual Kilograms mass() const = 0;
    . . .
};
```

- (a) List 5 ways in which this class takes advantage of the key aspects of object-oriented programming.
- (b) A consultant, Mr. Fancy, claims you are not using the full power of C++ operators, and suggests you provide `Airplane::operator=`, which sets all settable attributes of the assignee to the corresponding values of the rvalue. I.e. it does not try to make the two airplanes have the same name or serial number, but just other attributes such as mass, flapMaxAngle, etc. What are two major coding concerns with doing so.

3. (10 Points) Ex-Mayor Giuliani suddenly shows up as a photo-op in your company and surprises all by whipping out the following interface class for a new data structure.

```
class PQRTree {
public:
    RefCount references() const = 0;
    virtual const Ptr<PQRTree> newRef() const = 0;
    virtual deleteRef() const = 0;
    void newItem( Key, Data ) = 0;
    ... // More interface
};
```

Reporters are very impressed and ask you in front of Giuliani whether you can use this interface, assuming your existing software uses the style of this course.

- (a) Explain whether it is possible for you to use it or not, and why.
- (b) Assuming the reporters and the mayor have now left, describe how you would write this interface class yourself, and the advantages over Giuliani's version.

4. (10 Points) Assume you are providing a sophisticated new memory allocator implementation, `SopImpl`, using the approach described in the memory management chapter. Write each of the class definitions you would expect to define and briefly document the purpose of each. (The definition need only indicate name, and its base class(es), not the member functions.)
5. (10 Points) Assume you are managing a programming team building a GUI application and you come across the following interfaces developed by a member of your team in a code review:

```
class Viewer : public NamedInterface {
    . . .
    virtual void run() = 0;
    virtual void reset() = 0;
    virtual void suspend() = 0;
    virtual bool executing() const = 0;
    virtual int onControllerStart() = 0;
    virtual int onControllerStop() = 0;
    . . .
};

class Controller : public NamedInterface {
    . . .
    virtual int running() const = 0;
    virtual void onOff( bool ) = 0;
    virtual void freeze() = 0;
    virtual void unfreeze() = 0;
    virtual int onViewChange() = 0;
    . . .
};
```

- (a) Revise these two interface classes so they are consistent with the programming approach of the course (You do not need to provide additional interface classes in your answer.)
  - (b) Estimate the impact on “complexity” of the software in terms of the number of classes, amount of documentation, lines of code.
6. (10 Points) Consider a simulation of an airplane using the activity mechanism to advance time.

```
class CallbackImpl : public Notifieee<Activity> {
public:
    void onEvent( int );
    . . .
void
CallbackImpl::onEvent( int i ) {
    airplane_->onTimeAdvance( i );
}

void
AirplaneImpl::onTimeAdvance( i ) {
    // Update airplane state
    . . .
    t = computeNextTimeToUpdate(); // ***
    activity_->nextTime( t );
}
```

- (a) Write a numbered point list describing the sequence of functions that are executed from the statement `***` through to a second invocation of `***` with a brief comment indicating what each

one does. For simplicity, you can assume there is only this one activity. Identify the activity manager as AM, for brevity.

- (b) Write a similar list describing the sequence of actions that occurs to reclaim the activity when an airplane instance is deleted.

The End