Object-oriented Analysis and Design
Final Exam, Fall 2015

Date: December 10, 2015   **Marks: 90**   **Time: 180 mins.**

Section _____   Roll No. _____   Name_____

Question 1 (Max. Marks = 15)

For each of the statements given in the table below, <u>underline</u> either True or False. Think before you <u>underline</u> the correct option. Cutting/scratching will result in disqualification of answer.

S#	Statement	True/False
1	An object has its own value for each attribute but shares the attribute names and operations with other instances of its class.	True False
2	An implementation of a method by a specific class is called an operation.	True False
3	Analysis deals with "how" whereas design deals with "what".	True False
4	State diagrams should be constructed for all classes with meaningful temporal behavior.	True False
5	The interaction model describes the aspects of an object that change over time.	True False
6	Abstraction is the selective examination of certain aspects of a problem.	True False
7	A class model captures the dynamic structure of a system by characterizing the objects in the system, the relationships between the objects, and the attributes and operations for each class of objects.	True False
8	The term identity means that objects are distinguished by their inherent existence and not by descriptive properties that they may have.	True False
9	Class diagrams are useful both for abstract modeling and for designing actual programs.	True False
10	Association is a relationship among at least two classes.	True False
11	Association end names make it possible to create multiple references of same class.	True False
12	Aggregation is a form of composition with two additional constraints.	True False
13	Polymorphism acts as the only basis of reuse in object-oriented applications.	True False
14	Declaring attributes with class scope leads to inferior design as it is similar to declaring global variables.	True False
15	Multiplicity constrains the number of related objects.	True False

Question 2 (Max. Marks = 10 + 5 = 15)

Write C++ code in the space provided to show (compile-time and run-time) error free implementation of following two relationships shown in a UML 2 design class diagrams.
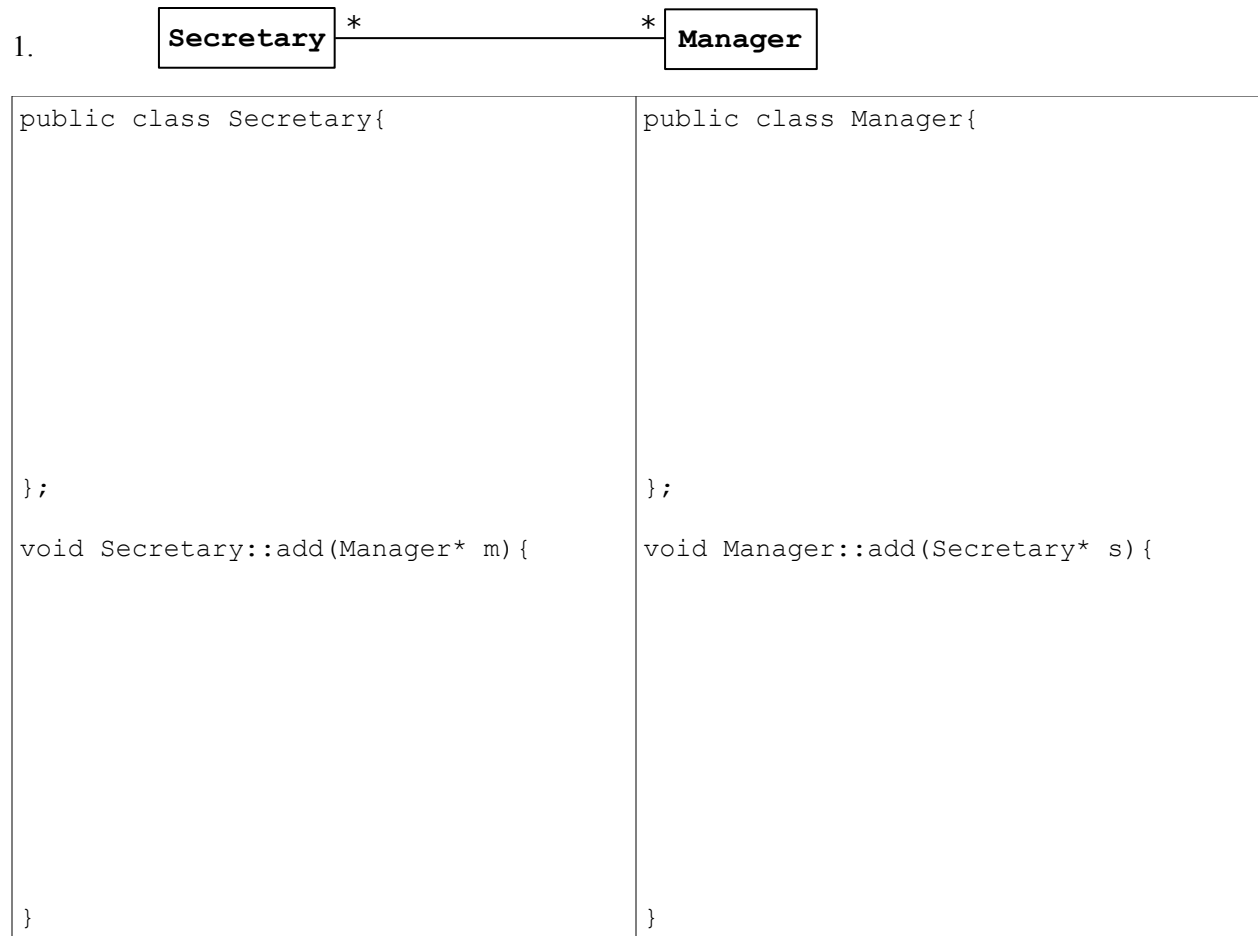
1.   Secretary `*` ———————— `*` Manager

```
public class Secretary{



};

void Secretary::add(Manager* m){




}
```

```
public class Manager{



};

void Manager::add(Secretary* s){




}
```

2.

```
       Vehicle
  ─────────────────
  +move(): void
  +stop(): void
```

```
         Car
```
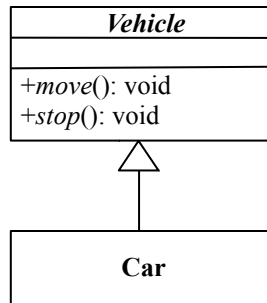
// Vehicle code goes here	// Car code goes here

Question 3 (Max. Marks = 10 + 10 + 10 = 30)

"Rename File" is one of the most important use cases of the FASTFile file management system. Needless to say a file must exist in order to be renamed. Moreover, FASTFile allows only administrators to rename files. When an administrator selects a file to be renamed, FASTFile highlights the selected file. Once the administrator has right-clicked the highlighted file, FASTFile displays a pop-up menu showing the various file operations. After the administrator has selected the "Rename" option, FASTFile makes the name of the file editable. If the administrator presses "Escape" at this point, FASTFile makes the file name uneditable and the use case ends. Typically, the administrator types the new name at this point and presses "Enter". FASTFile first checks whether the file being renamed is open. If the file is open, FASTFile displays "File Open" error message. If the file is closed, FASTFile checks whether the new name has any invalid characters (e.g. ?, ", /, :, <, etc.). If an invalid character is present, FASTFile displays "Invalid Character" error message. If no invalid character is present, FASTFile checks whether the new name matches the name of an existing file. If the new name matches an existing name, FASTFile displays "Name Match" error message. Otherwise, FASTFile updates the file name and the use case ends.

Note: The use case ends after any type of error message is displayed.

Use the information provided above to answer the following three parts of this question.

a. Document the entire "Rename File" use case using the **template** given below.

Identifier	
Purpose	
Priority	
Actor(s)	
Pre-conditions	
Post-conditions	

	Typical Course of Action	
S#	**Actor Action**	**System Response**

	Alternate Course of Action 1	
S#	**Actor Action**	**System Response**

	Alternate Course of Action 2	
S#	**Actor Action**	**System Response**

	Alternate Course of Action 3	
S#	**Actor Action**	**System Response**

	Alternate Course of Action 4	
S#	**Actor Action**	**System Response**

b. Model the entire "Rename File" use case using a UML 2 **swimlane activity diagram**.

c.   Model the entire "Rename File" use case using a <u>single system-level</u> UML 2 **sequence diagram**.

Question 4 (Max. Marks = 15 + 15 = 30)

Consider an Appointment Scheduling System that facilitates appointment scheduling between a client and a service provider. Service provider specifies the services (s)he offers and the corresponding venue and availability hours for the specific service. Client requests an appointment with the service provider by selecting the service and available time-slots and provides the necessary contact details. System resolves any scheduling conflicts in case multiple clients select the same slot simultaneously. Once the appointment is confirmed, system notifies the client through email. Client may also choose to be notified through SMS while making the appointment. Any subsequent changes in the appointment (e.g. possible rescheduling, cancellation, etc) are also notified accordingly. Service Provider can specify templates (message layouts) for sending Email and SMS notifications. Email template supports rich text editing whereas SMS template only supports plain text.

Use the information provided above to answer the following two parts of this question.

a. Draw the complete UML 2 design class diagram of the Appointment Scheduling System after incorporating all of the relevant design patterns.

b. Provide complete code containing the proper (error-free) implementation of the UML 2 design class diagram of part a.

//continue writing code here