Final

Note

This is an open-book exam. You may consult any books, papers, or notes, but discussion is strictly forbidden.

Problems

1. Please answer the following questions truthfully:

   (0) None (1) UML
   (2) Design Patterns (3) Verification

   (a) Which parts of the course are useful to you? (Select all that apply)

   *Answer:*_____ (5 points)

   (b) Which parts of the course are too difficult for you? (Select all that apply)

   *Answer:*_____ (5 points)

2. In a typical word processor, we may use the Command pattern to implement undo's. Consider the following implementation of the Command pattern:

```
abstract class Command {
  public ~Command () { /* ... */ }
  abstract public void Execute ();
  abstract public void Unexecute ();
  protected Command () { /* ... */ }
}

class CopyCommand extends Command {
  public CopyCommand (String) { /* ... */ }
  public void Execute () { /* ... */ }
  public void Unexecute () { /* ... */ }
}

class PasteCommand extends Command {
  public PasteCommand (String) { /* ... */ }
  public void Execute () { /* ... */ }
  public void Unexecute () { /* ... */ }
}
```

In the class, we implement `MacroCommand` naïvely. The naïve design cannot build new macros based on existing macros easily. You will apply the Composite pattern to resolve the problem. Please define the class `CompositeCommand` and rewrite the existing class(es) if necessary. (10 points)

3. Consider the following definition of binary trees:

```
abstract class Tree {
  abstract public print (void);
}

class Leaf extends Tree {
  public Leaf (int) { /* ... */ }
  public print (void) { /* ... */ }
}

class Node extends Tree {
  public Node (Tree, Tree) { /* ... */ }
  public print (void) { /* ... */ }
}
```

Recall that **inorder tree walk** prints the root of a subtree *between* its left subtree and right subtree. Similarly, **preorder tree walk** prints the root of a subtree *before* its left and right subtrees.

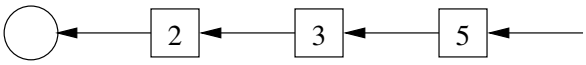
   (a) Please use the Visitor pattern to define the class `Visitor`. Modify the existing class(es) if necessary. (6 points)

   (b) Please define a `Visitor` subclass `InorderVisitor`. Modify the existing class(es) if necessary. (7 points)

   (c) Please define a `Visitor` subclass `PreorderVisitor`. Modify the existing class(es) if necessary. (7 points)

4. You will use the Decorator pattern to implement a prime number generator (this method is called the **sieve of Eratosthenes**). Initially, the generator is a counter generating natural numbers starting from 2. When the generator generates a number, we add a sieve of that number on top of the current generator. For instance, Figure 4 shows a counter (the circle) followed by three sieves (labeled 2, 3, 5). Each sieve obtains a number from its left neighbor and tests if the number is a multiple of its label. If so, it asks for another number. Otherwise, it return the number to its right neighbor.

   (a) Apply the Decorator pattern to define the classes `Component`, `Counter` and `Sieve` for the sieve of Eratosthenes. (10 points)



(b) Use the classes you defined to implement a prime number generator. (5 points)

5. Let p and q be atomic propositions. Answer the following questions about Computation Tree Logic (CTL):

   (a) Are $(\mathsf{AX}\ p) \wedge (\mathsf{AX}\ q)$ and $\mathsf{AX}\ (p \wedge q)$ equivalent? If not, please draw two computation trees to distinguish them. (5 points)

   (b) Are $(\mathsf{AF}\ p) \wedge (\mathsf{AF}\ q)$ and $\mathsf{AF}\ (p \wedge q)$ equivalent? If not, please draw two computation trees to distinguish them. (5 points)

   (c) Are $(\mathsf{EG}\ p) \wedge (\mathsf{EG}\ q)$ and $\mathsf{EG}\ (p \wedge q)$ equivalent? If not, please draw two computation trees to distinguish them. (5 points)

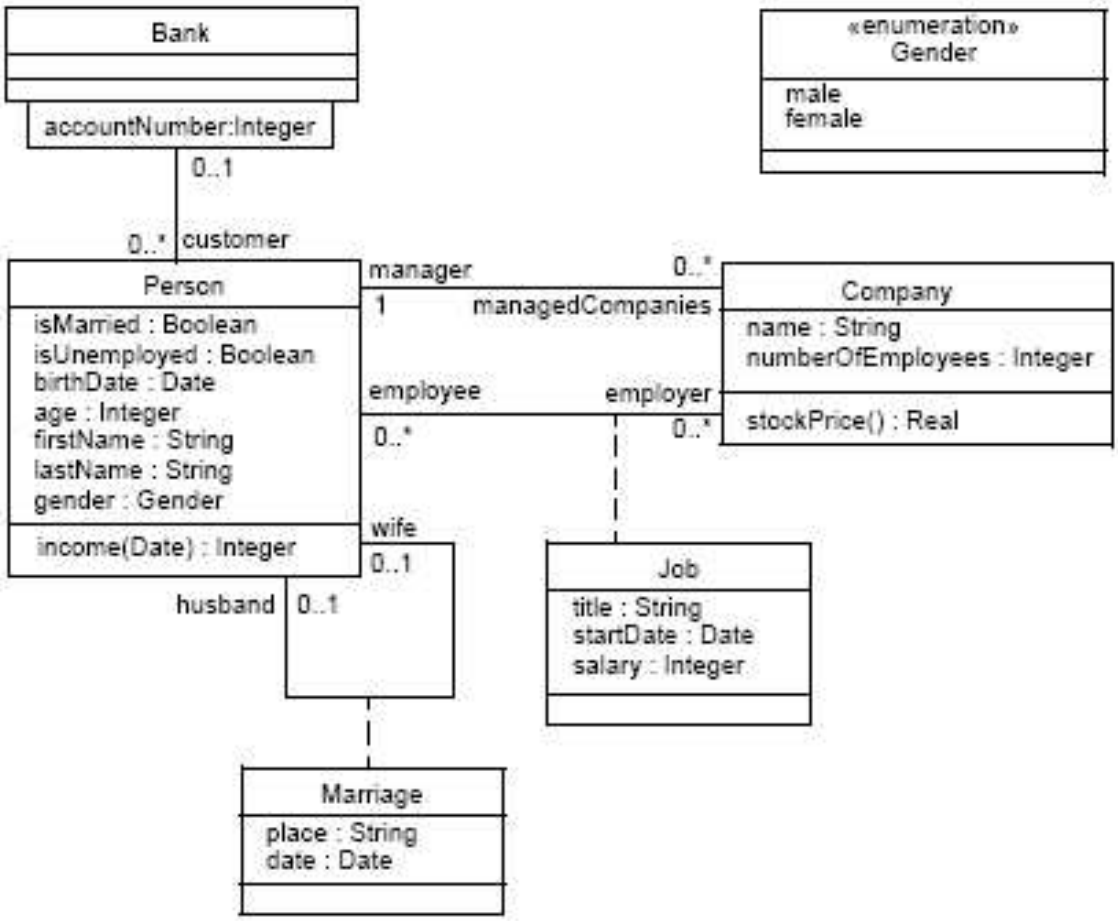
6. Consider the design of a very simple train control system. We want to make sure that the doors are always closed when the train is moving and they can be opened only when the train has come to a complete stop. Let *doors* (of type boolean) denote the close/open (*false*/*true*) state of the doors (all of which are in the same state at any time) and *speed* (of type non-negative real) denote the speed of the train.

   (a) Write a system specification as an LTL formula that characterizes the overall behavior of the train control system; make your own assumption about the initial state. (5 points)

   (b) Try to separate the control into two modules such that one controls the doors while the other controls speed of the train. Write LTL specifications for the two modules (such that their conjunction is equivalent to the system specification in (a)). (5 points)

   (c) Show that the conjunction in (b) is indeed equivalent to the system specification in (a). (5 points)

7. This problem refers to the Class Diagram Example in Chapter 7 of the UML 2.0 OCL Specification, shown below.



(a) Write an OCL expression specifying that a job with "CIO" as the title must pay at least (NT$) 100,000 for the salary. (2 points)

(b) Write an OCL expression specifying that a person can be employed only if the person is at least 18 years old. (3 points)

(c) How do you enforce that the values of "isMarried" and "isUnemployed" of a person are consistent with the state of the person? (5 points)

(d) Write an OCL expression specifying that any company with 5 employees or more must hire a female employee. (5 points)