CMPUT 301 Winter 2012 Final

Name:_____

CCID:_____

Object Oriented Analysis: Potential Classes and Methods [3 marks]

Read the following paragraph and pull out potential nouns that may lead to classes and verbs that may lead to relationships and methods according to Object Oriented Analysis.

There are TVs all over campus, they all play the campus TV station. The Campus TV station will play videos provided to them via an API. I want to make a system for the student newspaper where new classified ads are converted into a short 15 second video slide show. This slide show is then automatically uploaded to the Campus TV station.

List the potential Classes (appropriate nouns):

TV, station, video, API, ads, slide show

List the potential Actions/Methods/Relationships (appropriate verbs):

TV plays station

API provides videos

Station plays videos

Ads are converted to slideshow

slide show is uploaded to station

CMPUT 301 Winter 2012 Final

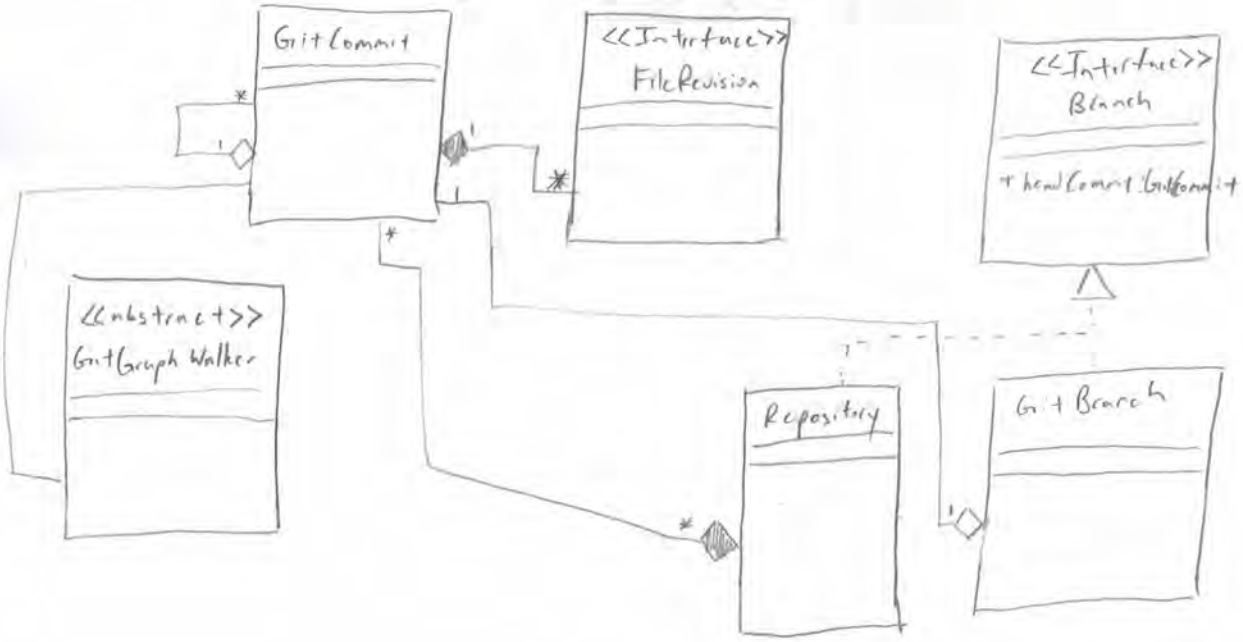
Name:_____

CCID:_____

UML: Composition or Aggregation? [3 marks]

Convert this Java code to a **UML class diagram**. This Java code meant to represent a Hand and its identified fingers that were extracted from a image of a person's hand. Draw a well-designed UML class diagram to represent this information. Provide the basic abstractions, attributes, methods, relationships, multiplicities, and navigabilities as appropriate.

```
public class GitCommit {
   List<GitCommit> parents;
   Author author;
   List<FileRevision> revisions;
}
public interface FileRevision {}
public class GitBranch implements
Branch {
   GitCommit head;
   public GitCommit headCommit();
}
public class Repository implements
Branch {
   GitCommit head;
   List<GitCommit> revisions;
   public GitCommit headCommit();
}
public interface Branch {
   public GitCommit headCommit();
}
abstract class GitGraphWalker {
   void walkParents(GitCommit g);
   void operation(GitCommit g);
}
```



CMPUT 301 Winter 2012 Final

Name:_____

CCID:_____

Use Cases and Use Case Diagram [3 marks total]

Background:

Kickstarter allows you to donate money to build a product and then receive credit, a prize, a gift, or credit for the product. Kickstarter does not require that projects be open-source or creative commons licensed. I want to make a Kickstarter clone that ensures all projects are under open licenses.

Goals:

Project owners want to create, describe and update the project and project descriptions.
Project owners want to define gifts/prizes/credits for different kinds of donations (e.g. For a software project $1 gives you a credit in a README, $10 gives you a signed hat from the developers).
Donators have to be able to sign up accounts.
Donators can donate money to projects using Paypal.com or Google Checkout.
Project owners need the contact and shipping information of the donators.

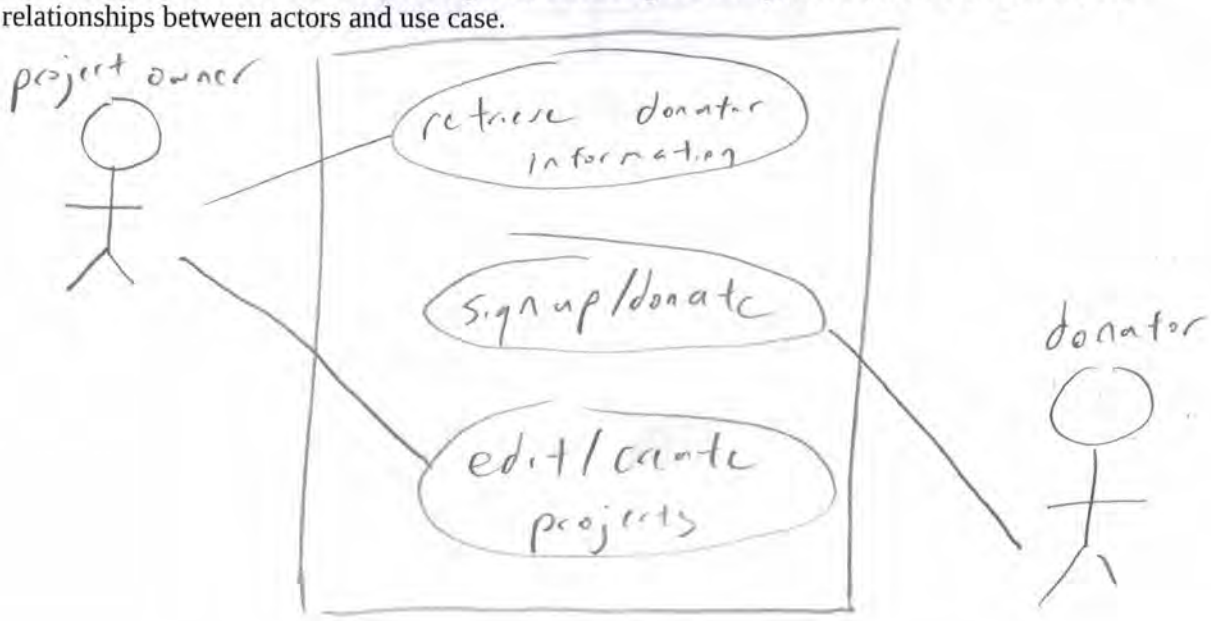
What are **three** primary use cases of the following situation:

Use case 1: project owners can edit/create projects

Use case 2: donators can signup/donate

Use case 3: project owners can retrieve donator information

Now complete this **UML use case diagram**, including boundary, actors, use case bubbles and relationships between actors and use case.



CMPUT 301 Winter 2012 Final

Name:_____

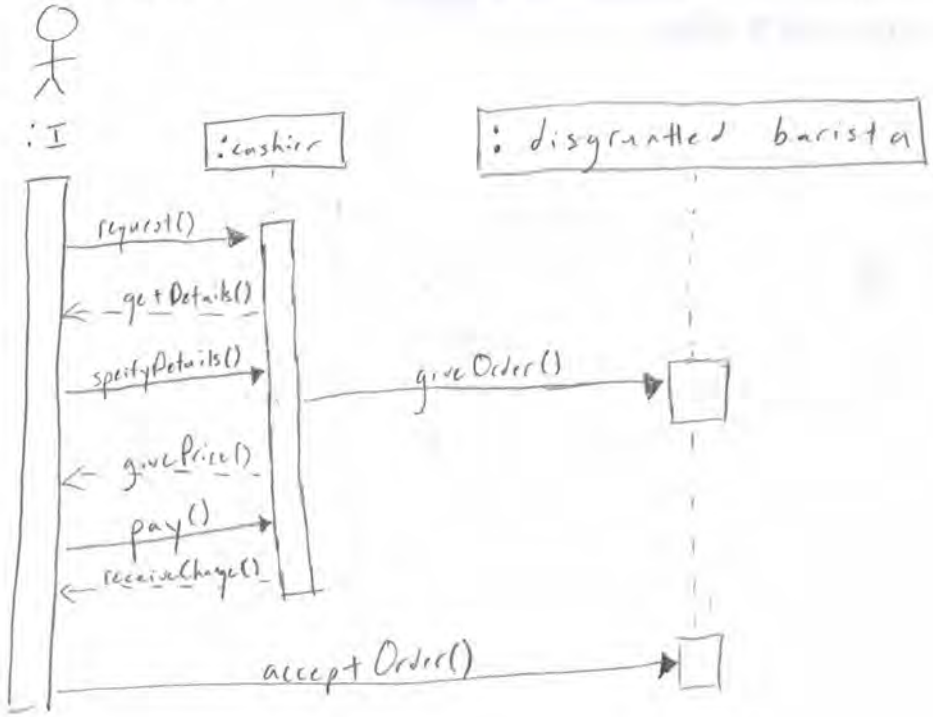
CCID:_____

UML Sequence Diagrams: [3 marks]

Convert this use case into a **sequence diagram,** remember to include all the actors, the components, the lifelines and use good names for the methods.

Use Case: Buying an iced coffee

1. **I** approach the **cashier** at the coffee bar and request an iced coffee.
2. The **cashier** asks if I want sweet syrup, cream, or milk in my iced coffee.
3. **I** specify my preferences for syrup, cream, milk etc.
4. The **cashier** yells the order to a **disgruntled barista (coffee attendant)**, who makes the iced coffee.
5. The **cashier** tells **me** the price and requests payment.
6. **I** pay and the **Cashier** accepts payment and deals with change if necessary.
7. I walk to end of the coffee bar and accept the iced coffee from the **disgruntled barista** (who I know is judging me for ordering iced coffee).



CMPUT 301 Winter 2012 Final

Name:_____

CCID:_____

Software Processes: [3 marks]

[2 marks] Explain what a **software development process** is.

engineering solution for creating software to have more effective time management, reduce software defects, increase quality and reuse existing successful solutions

[1 mark] Provide an example of **2** different **software development processes** and how they differ from each other.

Waterfall model: goes from one phase to another never going back to the same phase

iterative design: goes through many phases one cycle at a time

CMPUT 301 Winter 2012 Final;

Name:_____

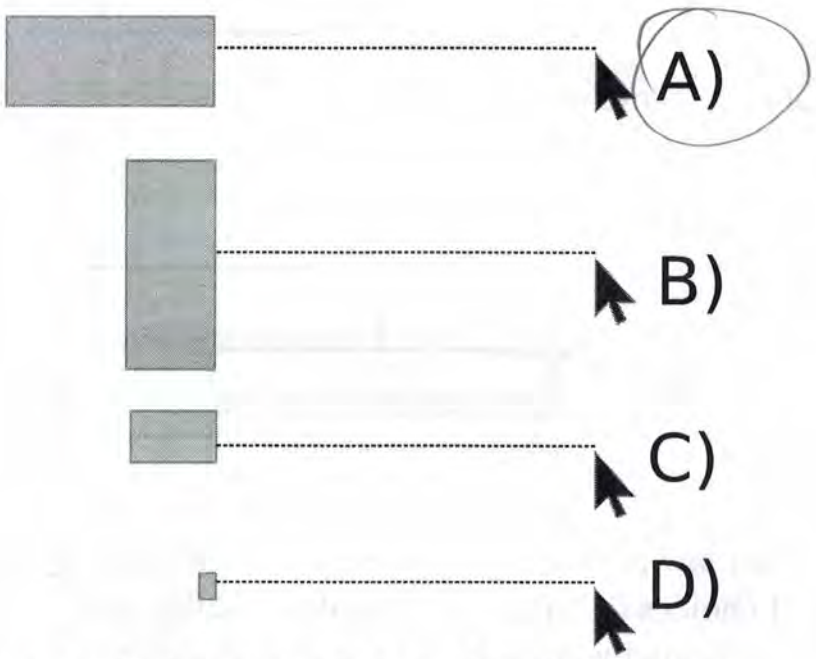
CCID:_____

Human Error: [3 Marks]

[1 mark] What is the **name** of the law that defines the speed of clicking on a target?

Fitt's Law

[1 mark] Which target is the fastest to click? Note: in all cases the cursor only has to move in one direction (horizontal) to the target. Also I have drawn the cursor relevant to the target.



[1 mark] Why does it take longer to click on the other targets?

they are not as wide

CMPUT 301 Winter 2012 Final;

Name:_____

CCID:_____

User Interfaces: [3 Marks]

[1 mark] Why must we be very careful about the colors we use in user interfaces (e.g. What's wrong with red and green)?

users may be colorblind and unable to see the distinct colors

[1 mark] Give 2 examples of interface metaphors.

desktop

menus

[1 mark] What law estimates the average time to make a simple decision from a set of choices (if subdivision applies)? Alternatively if you forget the name, what is the mathematical relationship between *n* choices and *t* time?

Hick's Law

$\ln 2(n+1)$

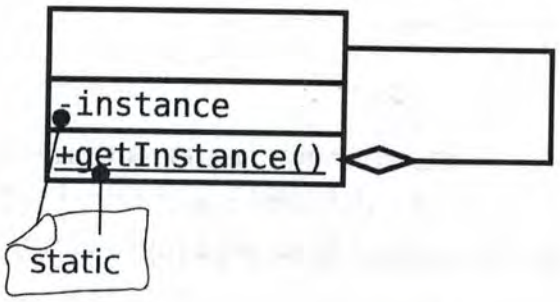
CMPUT 301 Winter 2012 Final;

Name:_____

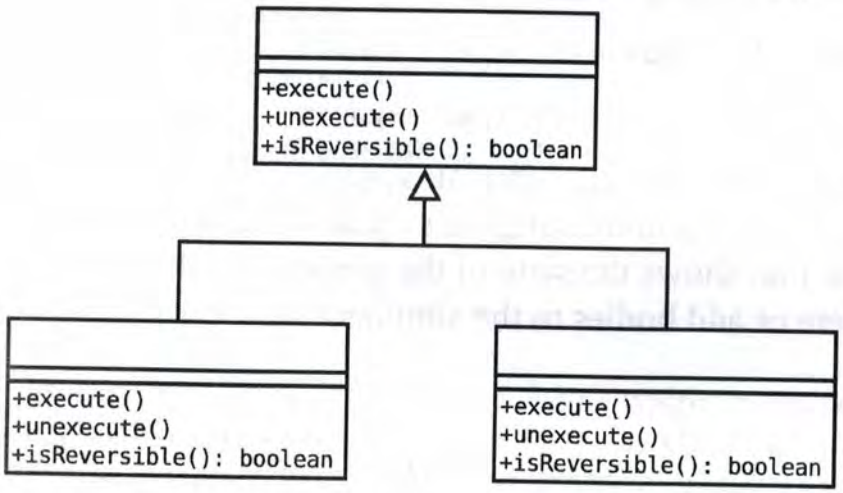
CCID:_____

Design Patterns: [3 Marks]

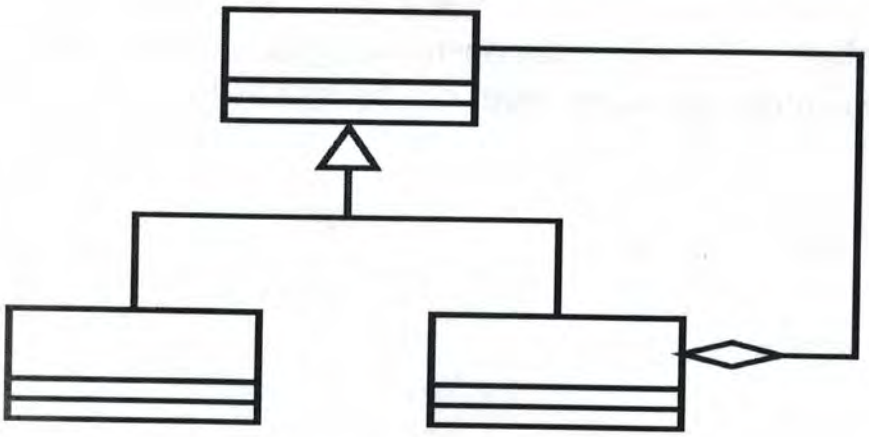
Identify and name each of these design patterns. If you make an assumption, explain it.



Singleton



Command



composite

CMPUT 301 Winter 2012 Final;

Name:_____

CCID:_____

Design Paterns: [3 Marks]

Read the following scenarios, then **name** and **explain** the design pattern most appropriate to address this problem.

A) You have an algorithm for recognizing different kinds of minerals from photos of river bed sand. This algorithm needs some specialized logic for each different mineral, but the general control flow and logic can be shared.

template pattern since it provides a framework for adding a subclass of each different mineral (distinctive part of algorithm)

B) You are building a gravity simulator that simulates planets. The 3D view is hard to control and hard to control and configure so you want to provide a 2D view and a textual view that shows the state of the planets. Furthermore you want to be able to delete or add bodies to the simulator as it is running.

decorator since you can add and remove objects dynamically

C) You're making a mass photo editor where the operations you take (such as change brightness, change contrast, equalize, crop) can be repeated across an entire directory of photos.

composite since photos can be treated uniformly

CMPUT 301 Winter 2012 Final;

Name:_____

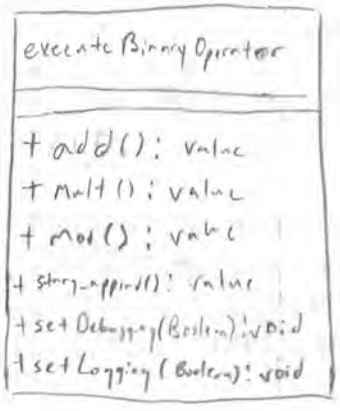
CCID:_____

Refactoring: [3 Marks]

```
// We are implementing a LISP interpreter and we want to
// be able to do arithmetic, we have already parsed the code
// but now we have to execute representations of it
// for instance if we want to execute the code(+ 10 12) which means
// 10 + 12, and it should return a value of 22.
class Interpreter {
    ...
    Value executeBinaryOperator(BinaryOperation operation, Value
first, Value second, boolean debugging, boolean logging) {
        ...
        switch( operator.getOperatorType() ) {
           case ADD:
               return first.toInteger() + second.toInteger();
           case MULT:
               return first.toInteger() * second.toInteger();
           case MOD:
               return first.toInteger() % second.toInteger();
           case STRING_APPEND:
               return first.toString() + second.toString();
        ...
        }
    }
}
```

[3 mark] **List** at least 2 bad smells one finds, and then at least 1 refactoring one could apply to this code snippet and then **draw** the **UML class diagram** of the relevant code after you applied these refactorings. State assumptions.

bad smells: switch statements, long parameter list



CMPUT 301 Winter 2012 Final;

Name:_____

CCID:_____

BONUS 3 Marks Testing: [3 Marks]

Provide 5 really good test cases for implementations of the following interface, but remember implementations can vary so sometimes naive mistakes are made. 5 good tests would consist of only 1 test per equivalence class.

```
public interface Statistics {
     // The mean or average of a set of values is the central
     // tendency of data commonly represented as the sum the values
     // divided by the number of values. e.g. sum(v)/length(v)
     public double average(double [] values);
}
```

Test Input for average	Output of average
[1.0]	1.0
[Double.NaN, Double.NaN]	NullPointerException
[1, 2, 3]	2.0
[]	0.0
[Double.MAX_VALUE, Double.MAX_VALUE]	ArithmeticException
[Double.MIN_VALUE, Double.MIN_VALUE]	Double.MIN_VALUE

Hints: Double.MAX_VALUE and Double.MIN_VALUE define the maximum and minimum double floating point values in Java. Also doubles can't represent all **Real** numbers. NaN in doubles means Not a Number (Double.NaN) as in undefined. Popular exceptions include `DivideByZeroException` and `NullPointerException`.