Question 1. Multiple Choice [20 points total; 2 points each]

Circle the best answer for each multiple choice question. Only one answer may be selected for each multiple choice question.

Part I

In software development, requirements are about:

(a) ***what the software system should do***

(b) how the software system should work

(c) both about what the software system should do and how the system should work

(d) none of the above

Part II

When the estimate for a user story is accurate but the estimate is very large:

(a) there is no impact on the overall project, the story just takes longer to implement.

(b) ***the overall sprint is impacted as the user story might not be delivered***

(c) the overall project is doomed to failure because the developers cannot estimate

(d) by definition a user story cannot be too large

Part III

When the REST principle is being used to describe the interactions between a client and a server:

(a) the server maintains the state of the client

(b) the client can send up to five kinds of `http` requests: create, get, put, post and delete

(c) ***the client maintains state***

(d) both the client and the server maintain state

Part IV

When using an agile development process, at a Daily Scrum meeting:

(a) ***the meeting is guided by three questions: 1) what did you do yesterday, 2) what will you do today and 3) what obstacles are in your way?***

(b) the manager should attend to give feedback about the performance of team members

(c) task durations are never changed

(d) always includes a demo of the current state of the system under development

Part V

When using an agile development process, the product backlog:

(a) is a list of the products a company needs to build

(b) ***is a prioritized list of backlog items specific to the product under development***

(c) is a list of defect associated with the current product

(d) is a list of engineering tasks negotiated by the team and the product owner about the current development project

Part VI

To satisfy the INVEST properties, a user story must:

(a) have an associated automated test

(b) have an associated manual test

(c) ***have either an associated automated test or an associated manual test***

(d) does not need an associated test

Part VII

A suitable user-instruction for testing a prototype of your comic strip project is:

(a) Go to this URL, enter a filename with the cell, press the add button.

(b) Use this application to ensure your internet connection works and to add a cell to this comic strip.

(c) ***Use this application to add a cell to this comic strip.***

(d) None of the above.

Part VIII

Recall that in the dependency inverstion principle:

1. High-level modules should not depend on low-level modules. Both should depend on abstractions.
2. Abstraction should not depend on details. Details should depend on abstractions.

The dependency inversion principle in object-oriented design:

(a) is always being used if we define and use interfaces

(b) ***is intended to limit the scope of changes in an application***

(c) Both a) and b)

(d) None of the above.

Part IX

Imagine you have been asked to implement a Piazza-like system. One of your team members defines the following user story:

> *As a professor, I would like to be able to post announcements so that they appear on my students' feed.*

This user story:

(a) meets the "V" goal of the INVEST property: to be valuable to users or customers

(b) meets the "T" goal of the INVEST property: to be testable

(c) ***both a and b***

(d) None of the above.

Part X

Imagine that you have been asked to implement a Piazza-like system. One of your team members defines the following user story:

As a developer, I need to implement an ability to filter posts for viewing by a student by different criteria.

(a) meets the "V" goal of the INVEST property: to be valuable to users or customers

(b) meets the "T" goal of the INVEST property: to be testable

(c) both a and b

(d) ***None of the above.***

Question 2. RESTful Systems [~~8~~ points total]

A shopping list application is being developed in which a group of users share a common list. The following statements describe the behaviour of the application:

- Each user can put individual items onto the shared list.
- Each user can retrieve all items currently on the list.
- Each user can check if a particular item is currently on the list.
- The only means of changing the shared list is for an individual user to add or delete a single item.

You have been asked to design a RESTFul interface to this appiication. Complete the following table with the responses to each requests in your design.

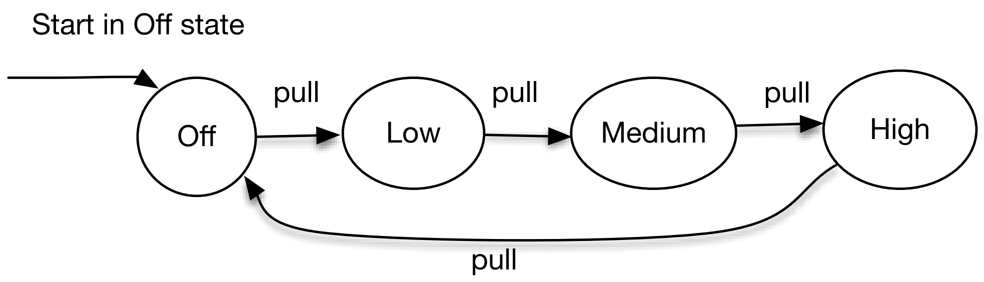
Resource	GET	PUT	POST	DEL
items	returns the list of items	not implemented	~~not implemented~~ Add a single item (more common on collection)	not implemented
items/item	returns the item	not implemented	~~add a single item~~ not implemented	delete the item

Question 3. Design: State Pattern [Total: 15 points]

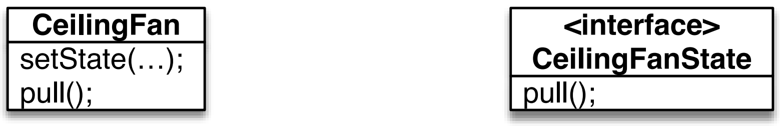
You have been asked to design and implement software to control a ceiling fan. The fan has one pull cord. The state machine shown below indicates that the fan starts in the `Off` state. When the pull chain is pulled, the fan starts rotating on low speed (the `Low` state). When the pull chain is pulled again, the fan starts rotating on medium speed (the `Medium` state). When the pull chain is pulled again, the fan starts rotating on high speed (the `High` state). When the pull chain is pulled again, the fan stops rotating.

Your teammate started to sketch a UML class diagram for this sytem in which the State design pattern is used. You have been asked to complete the application of the State Design Pattern for this system.

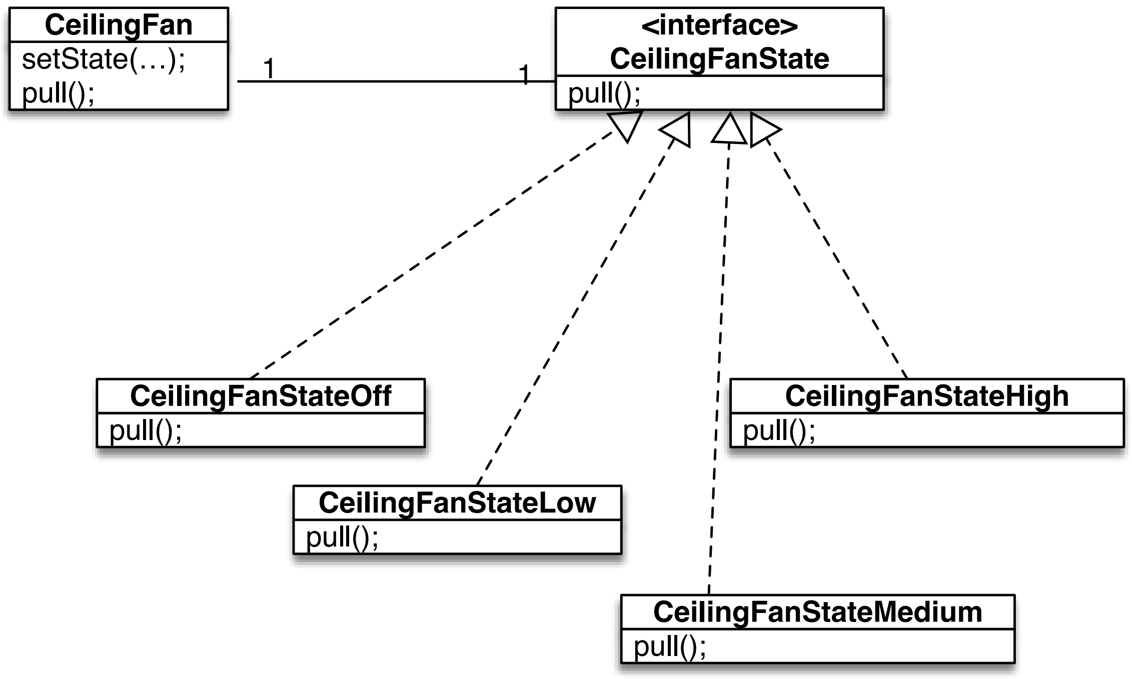
Note: Refer to the UML guide provided at the back of the exam for the structure diagram of the State design pattern.



Initial class diagram



a) Sketch any classes and interfaces (not methods) needed to complete the application of the State design pattern to this problem. Include any needed extends, implements, association or composition relationships between the classes and interfaces to the class diagram. [5 points]



b) Using Java or Typescript code, write the code for the class representing the `High` state. Your code needs to be close to correct Java or Typescript code but does not need tto be compilable. (You should only need 5 lines of code. If you are unable to write the code, express what you would write in code in English. You will be docked 1 mark if you write in English rather than providing Java or Typescript code.) [3 points]

```
class CeilingFanStateHigh implements CeilingFanState {
   private CeilingFan _fan;

   constructor(fan: CeilingFan) {
     this._fan = fan;
   }

   pull() {
     this._fan.setState(new CeilingFanStateOff(_fan));
   }
}
```

c) Using Java or Typescript code, write the code for the `CelingFan` class. Your code needs to be close to correct Java or Typescript code but does not need to be compilable. (You should only need 12 lines of code. If you are unable to write the code, express what you would write in code in English. You will be docked 2 marks if you write in English rather than providing Java or Typescript code.) [7 points]

```
class CeilingFan {

   private CeilingFanState _state;

   constructor() {
     this._state = new CeilingFanStateOff(this);
   }

   setState(state: CeilingFanState) {
     this._state = state;
   }

   pull() {
     this._state.pull();
   }

}
```