CS 169 Spring 2019 MT2 Practice

Name: _____ **SID**: _____

Name of person on your left: _____

Name of person on your right: _____

I certify that in accordance with the UC Berkeley honor code, all the work on this exam is my own, that I did not receive assistance from anyone in constructing the answers, and that I did not use any resources other than those specifically allowed or provided for by the exam. I understand that the penalty for academic dishonesty is a grade of zero for the exam with no possibility of make-up; a likely reduction in the course final grade in addition to that implied by the lower exam grade; and referral of the case for disciplinary action and possible un-enrollment from the course.
Signature:

Instructions

1. All questions that you can select more than one answer say "Select ALL that apply".
2. If not specified to "Select ALL that apply", choose only one answer.

1. If a Dog has many Collars, and a Collar is owned by exactly one Dog, and you have a `dogs` DB table and a `collars` DB table, what is the best way to model this relationship (adding the minimal number of columns as possible)?
    a. Add column `collar_id` to `dogs` table. And add `dog_id` to `collar` table.
    b. Add column `collar_id` to `dogs` table.
    c. **Add column `dog_id` to `collar` table.**
    d. Add a separate table `dog_collar`, with the columns `dog_id` and `collar_id`.

2. Select ALL that apply. If a Person has many Collars through Dogs, and you want to run the query `@person = Person.find_by_username('carinaboo') ; @person.collars`, indexing what foreign key column(s) would help speed up this query. In your answer, do not select any columns that will NOT help speed up this query. (Hint: Think carefully what tables would have the primary keys to model these relationships.)
   Person has many Dogs. Dog belongs to one Person.
   Dog has many Collars. Collar belongs to one Dog.
    a. `people` table, `dog_id` column
    b. **`dog` table, `person_id` column**
    c. `dog` table, `collar_id` column
    d. **`collar` table, `dog_id` column**

3. Which would you use if you want to test that `sortTweets` correctly sorts tweets by most shares. `sortTweets` calls `TwitterService.getTweets` which makes a network request to Twitter's API. However you already know what the typical response from their API is.
    a. expect.to receive(...)
    b. Mock
    c. Stub
    d. Fixture
    e. Factory

4. Which would you use if you want to test that `sortTweets` eventually makes a call to `TwitterService.getTweets` with a set of specific arguments. However, you don't actually care what the response back from the API is. Just that `getTweets` is called exactly once.
    a. expect.to receive(...)
    b. Mock
    c. Stub
    d. Fixture
    e. Factory

5. Which would you use if you want to test that a logged-in user's profile page has all the UI elements you expect on there. But to do that, you need a `User` with almost all of it's 20 attributes filled in (name, email, profile picture URL, bio, etc). What is the best way to create a test `User` for this test? Note most of your tests don't need test users.
    a. Mock
    b. Stub
    c. Fixture
    d. Factory

6. Which would you use if you need the user's timezone set for your app and all of your tests to run? The user's timezone is stored in a database table.
    a. Mock
    b. Stub
    c. Fixture
    d. Factory

7. Which would you use if you wanted to test that `show_list` shows a list of all the puppies available for adoption. The only attribute of `Puppy` (of the 30 attributes in total) that is called is `Puppy.name`. What is a good option for creating the simplest list of puppies to run our test with?
    a. Mock
    b. Stub
    c. Fixture
    d. Factory

8. Select all that apply: Which statements are TRUE regarding refactoring?
    a. Improving code structure is a primary goal
    b. Refactoring improves test coverage
    c. During refactoring, you should never have a failing test
    d. Refactoring always reduces the amount of code

9. Refactoring makes sense for all BUT which of the following:
    a. Making code more readable.
    b. Making code more modular.
    c. Making code more testable.
    d. Making code less buggy.

10. The Open/Closed Principle states that:
    a. Classes should be open for subclassing but closed for extension.
    b. Classes should be open for composition, but closed for source modification.
    c. Classes should be open for extension but closed for source modification.
    d. Classes should be open for testing, but closed in a production environment.

11. Select ALL that apply. Which is TRUE about template and strategy design pattern?
    a. Both are patterns we can use to help our service stick with the Open/Closed Principle.
    b. Both are patterns we can use to help our service stick with the Liskov Substitution Principle.
    c. Template design pattern is good to use when there is a defined set of methods you can define in a superclass that will always be called for a task. This means any new subclasses you want to add will just have to override the implementation of that set of methods.

d. **Strategy design pattern is good to use when then general task is the same and can be encapsulated into a method, e.g. `output_report()`, but the actual implementation can be very different. In this case, we'd typically structure it so a class delegates to another class, say `class B`, for that task, and `class B` is open for subclassing.**

12. Singletons are most useful for:
    a. **Not having to create many instances of the same class, e.g. instance of a null object.**
    b. Keeping a class's API cleaner.
    c. Keeping track of global state.
    d. Decreasing global variables clutter, because we can use singletons to replacing any global variables e.g. `$variable` we would have created.

13. Where is the best place to store a runtime feature flag on/off value.
    a. In a cookie
    b. In a server config file (e.g. .yaml configuration file)
    c. **In a database table**

14. With continuous integration, what is the best way to hide a large incomplete feature from users? This feature is being worked on by 10 engineers and will span multiple iterations. We do not want to leak the feature to user's browsers until ready for launch.
    a. Put the entire feature into one commit so it can be pushed all at once.
    b. **Use a server-side feature flag to disable the new feature behavior.**
    c. Add a URL query parameter on production to decide whether the feature is enabled.
    d. This is not feasible with continuous integration.

15. Unit tests should:
    a. Test that the UI behaves correctly when a user interacts with it (e.g clicks a button).
    b. **Test that the output of a method is correct for a given input.**
    c. Test that an entire module performs the correct behavior for a given input.
    d. Test that your feature correctly implements the customer's user stories.

16. Select ALL that apply. Examples of a proxy object include:
    a. **A `DirectionsService` that serves cached offline results if the user has downloaded this city's map area, else calls the Maps server.**
    b. **A `MailService` that sends out the email if online, else queues the email for sending.**
    c. **`Movies.reviews`, which returns an object that acts like a Collection of DB data objects, but doesn't actually do the SQL query until you need to.**
    d. The subclass `CanadianGoose`, that subclasses `Goose`, and overrides the superclass' `draw` method to output a different PNG image.

17. Which is FALSE about the observer pattern:
    a. The observer doesn't need to be any special type of class.
    b. **There can only be one observer.**

c. The subject doesn't need to be any special type of class.
d. A subject can add itself as an observer.

18. The Demeter Principle states that:
    a. A class should depend on the interfaces of other classes, not the implementations.
    b. A class can only call methods on itself and its instance variables, but not on results returned by them.
    c. A class should not leak its implementation by exposing private methods.
    d. A class may only have one delegate.

19. What is NOT a way you can improve response time.
    a. If your site currently includes many Javascript files separately, minify them into just one Javascript file.
    b. Cache any HTML pages and partial HTML that can be cached.
    c. If there are a lot of image thumbnails shown on your site, reduce the size and quality so each image is less bytes, but to the user, the images still look fine.
    d. All of the above can help improve response time.

20. When doing a database query like `@reviews = Review.where(rating: 5)`, which we later call `@reviews.each do |review| ; review.moviegoers.first` , we should:
    a. Lazy load the `moviegoers` table data, deferring it until the result is needed, so initial latency is decreased.
    b. Always eager load associated tables, e.g. `.includes(:moviegoers)` and `.includes(:movies)`, to decrease latency if the result is needed later. Including other tables at the time of the first SQL query doesn't increase initial latency.
    c. Combine subqueries into the initial query via eager loading if we'll need those subqueries eventually, e.g. add `.includes(:moviegoers)` in this case.

21. Select all that apply: Suppose we want to add a Theaters model to Rotten Potatoes, with the simplifying assumption that each Theater is showing only one movie at any given time, but a given Movie could be showing at many Theaters. Besides adding a theaters table to the database, which steps are necessary so that movie.theaters will return a list of all the theaters at which a movie is showing?
    a. The theaters table will need a foreign key movie_id.
    b. The movies table will need a foreign key theater_id.
    c. We must add has_many :theaters to the Movie model.
    d. We must add belongs_to :movie to the Theater model.
    e. We must add has_many :movies to the Theater model.
    f. We must add belongs_to :theater to the Movie model.

22. Assuming that a Movie has many Reviews, a Review belongs to a single Movie, and movie ID 5 exists, what table(s) will be updated as a result of the following code? (HINT: recall that build is like new in that it creates and populates a new instance of the owned object, but does not save anything to the database.)

```
m = Movie.find(5)
```

```
m.reviews.build(:potatoes => 5)
m.save!
```

a. Only the movies table
b. **Only the reviews table**
c. Both the movies table and the reviews table
d. You cannot tell from the code given

23. We're creating a new app that allows students to schedule appointments with faculty at specific times and days. Which BEST describes how to model appointments between faculty and students:
    a. **Faculty has-many appointments; Student has-many appointments**
    b. Faculty has-many Appointments, through Students
    c. Faculty belongs-to Appointment; Student belongs-to Appointment
    d. Faculty has-and-belongs-to-many Students; Student has-and-belongs-to-many Faculty

24. Select ALL that apply. A `DogTrainer` has many `Customers`. A `Customer` has many `Dogs`. Which are violations of the Demeter Principle.
    a. **`dog_trainer.customers.first.dogs`**
    b. **`dog_trainer.customers.first.dogs.where(color: 'brown')`**
    c. `dog_trainer.customers`

25. Adnan needs to deploy a new version of his app and a new database schema that goes with it. Compared to the current schema, the new schema adds some new columns and deletes some existing columns. Order the following steps correctly for using a feature flag to deploy this change:
    i. Apply the destructive migration
    ii. Apply the nondestructive migration
    iii. Deploy the new code including the feature flag conditionals
    iv. Deploy the new code without the feature flag conditionals
    v. Flip the feature flag
    a. i, iii, v, iv, ii
    b. **ii, iii, v, iv, i**
    c. ii, iii, i, v, iv
    d. iii, ii, v, i, iv
    e. iii, ii, v, iv, i

26. Code smells (e.g. that SOFA helps indicate) can be harmful because they indicate:
    a. There's a serious bug in the code.
    b. The code may be doing unnecessary work, causing slowdown.
    c. The code is not validating user input correctly, and can produce unpredictable results.
    d. **There's an issue with the design of the code, which can slow down development and testability, and can cause bugs later on.**
    e. Working with such code may be harmful to your health. You should ask your manager for air filters and nice-scented Febreze for the office.

27. If you have many classes that make CRUD operation calls to the database, but you need to be able to swap out between multiple databases, e.g MySQL, PostgreSQL, Oracle, what is the best design pattern to use?
    a. **Adapter**
    b. Observer
    c. Composition and delegation
    d. Composite

28. Suppose you work on Google Maps and you sync a user's labeled places data every few minutes. If there is new data, you want to update the basemap, user's labeled places list view, and any place details views open. What is the best pattern to use?
    a. Delegate. Have the sync class keep pointers (be composed of) the basemap, labeled places list view, and the current place details view, and delegate to those to update.
    b. **Observer. Have all the different views be observers of the successful sync change.**

29. Sometimes, an array of a type can be treated like the type itself, if they share a large amount of the same functionality (e.g. `Ticket` and a `Subscription` of several tickets). In this case:
    a. The single type can subclass the collection superclass.
    b. **The collection can subclass the single type superclass.**
    c. We can create an adapter from the collection to the single type.
    d. They should always remain different types, because otherwise it would break the LSP.

30. Select all that apply: Which are TRUE regarding the use of points, velocity and Pivotal Tracker to measure progress in BDD and Agile?
    a. **Velocity is useful for gauging the progress of one team across iterations.**
    b. When a story is completed, the developer who checked in the code and test(s) for that story is responsible for marking the story as Accepted.
    c. Velocity is useful for comparing the progress of two teams across iterations.
    d. **An experienced team can use velocity to predict when certain features will be ready to ship.**

31. Select all that apply: Which are TRUE regarding daily scrums?
    a. Scrum members are responsible for prioritizing their own user stories.
    b. **The Scrum Master is responsible for dealing with obstacles that may impede team progress.**
    c. **The Scrum practice encourages team members to switch among different roles frequently (Scrum Master, Product Owner, etc.).**
    d. Scrum implies Agile development practices such as TDD.