Architecture

1. What is the difference between an HTTP PUT request and an HTTP POST request?

   An HTTP PUT request is idempotent, meaning that multiple PUT calls will not do anything different than a single PUT call. HTTP POST requests are not idempotent.

   You see a link in an interface whose markup is as follows:

   ```
   <a href="images/new?request_type=PUT" method="POST">create a new image</a>
   ```

2. Is the target URL relative or absolute?

   The target URL is relative.

3. What is the difference between an absolute and relative URL?

   A relative URL is a path to a resource on the local server starting from the location of the page you are currently accessing. An absolute URL is a path to a resource starting with the domain name or IP address (e.g. www.somesite.com/location/of/resource). With an absolute URL you can access resources from other sites/servers, and if you move a page around on your server then you don't have to change the URL, as you would if you were using a local URL.

4. If you clicked on this link, what kind of request would your browser generate? (which HTTP method?) Assume no JavaScript modifies the behavior of the link.

   I think it would generate an HTTP GET request, since it's a link, it asks to get a page from the server, and I don't believe "method" is a valid attribute for the anchor tag.

5. Is there a query string? What is it?

   There is a query string. That query string is `?request_type="PUT"`.

6. What is lacking from the link declaration that would otherwise enhance accessibility?

   I think that the link declaration is lacking a "rel" attribute that could enhance it's accessibility.

7. What are the roles of the database and Web browser in most Web applications? (One sentence for each.)

   The database stores any data that needs to persist, such as login information, order information, user settings (such as gmail settings). The Web browser is the user interface, the mediator between the logic of the application and the user.

   Given the following HTTP response header:

```
HTTP/1.1 200 OK
Date: Wed, 09 Mar 2011 16:43:33 GMT
Server: Apache
Connection: Keep-Alive
Keep-Alive: timeout=2, max=100
Etag: "110e412f-7df-49e0f6a106500"
Vary: Accept-Encoding
```

8. Would an HTTP response that begins like that usually contain a body? Why or why not?

   Yes, an HTTP response that begins with `HTTP/1.1 200 OK` would usually contain a body, because that first line says everything went fine with your request and it should contain a body of whatever information your HTTP request asked for.

Ruby

9. Write a Ruby class definition that meets the following criteria:

   - class is called `Troll`
   - class has publicly accessible attributes `ugliness`, `smelliness`, and `strength`
   - upon instantiation, an object of this class has a member variable, a String, called `grunt`, whose initial value is "UNGAH" (that's pronounced "oon-gah")
   - class has an instance method called `speak()` that prints the value of the instance variable grunt 42 times
   - class has an instance method called `reverse()` that prints the value of the instance variable grunt backwards
   - class has a static/class method called `propogate()` which returns a Troll instance whose grunt attribute is "eegah"

```
class Troll
  attr_accessor :ugliness, :smelliness, :strength

  def initialize
    @grunt = "UNGAH"
  end

  def speak()
    42.times { puts @grunt }
  end

  def reverse()
    @grunt.reverse
  end
```

```
def propogate()
  t = Troll.new()
  t.set_grunt("eegah")
  return t
end

protected
def set_grunt(grunt)
  @grunt = grunt
end

end
```

10. Imagine a `Troll` instance, `fred`, which, when the following method is called:
`fred.respond_to?("fight")`
returns true. What is missing from your class definition in order for this example to be accurate?

My class definition is missing a method called `fight()` in order for this example to be accurate.

11. Does the `respond_to?()` method illustrate object-oriented polymorphism? If so, in what manner?

Yes, the `respond_to?()` method illustrates object-oriented polymorphism. It illustrates polymorphism in the manner that multiple objects can have the same method. It essentially asks is this object able to do what I want it to do, and by extension, if you know a method of another object, can this object do what this other object to do. Also, the `respond_to` method itself is polymorphic since any object can call it.

12. According to Ruby conventions, what kind of value would you expect to receive from a method that ends in a question mark (?)?

I would expect a boolean value (true, false).

13. According to Ruby conventions, what is the difference between pairs of methods like `do_this()` and `do_this!()` (notice the bang) ?

The method without the bang will return a copy of the object that called `do_this()` with whatever `do_this()` did to the object. The method with the bang will modify the object that called `do_this!()`.

14. Briefly explain Ruby's type system. What is it (by name)? What does it mean?

    Ruby's type system is called *Duck typing*. The name stems from the saying "If it walks like a duck, and quacks like a duck, then it's a duck." Types are determined at run time in this system.

15. What type of Ruby object does the following expression yield?
    ```
    %w( master rails and then try another framework you'll never go back)
    ```
    This will return an array of strings.

16. Given an array of strings, `@happy_places`, would these two snippets of code do the same thing?

    ```
    @happy_places.each do |happy_place|
      puts happy_place
    end
    ```

    and

    ```
    @happy_places.each { |hp| puts hp }
    ```

    Yes. They are two different ways of doing blocks of code, and the second snippet uses a shorter variable name for each variable in `@happy_places`.

17. Given a function that needs to return a value to it's caller, does the function need an explicit `return` statement? If so, explain why? If not, then what can you always expect a Ruby function to return?

    A function does not need an explicit `return` statement. Unless you specify an explicit `return` statement, Ruby functions always return the value of the last expression in the function, and since everything in Ruby is an expression, this would be the value of the last line in the function.

Rails

18. Name four ActiveRecord callbacks that you can bind methods to.

    Four ActiveRecord callbacks that you can bind methods to are `validate`, `before_save`, `before_create`, and `after_create`.

19. The Rails convention maps HTTP methods to certain controller methods, and those methods usually involve specific CRUD operations on models. Given the following CRUD database methods:
    `create`, `read`, `update`, and `delete`
    and the following HTTP methods:
    `GET`, `PUT`, `POST`, `DELETE`
    and the following controller actions:
    `index`, `new`, `create`, `edit`, `update`, `destroy`
    Complete the following table:

HTTP method	controller action	CRUD operation
GET	index	read
None, unless you save it. Then it would be PUT.	new	create
PUT	create	create
None, unless you save it. Then it would be POST	edit	update
POST	update	update
DELETE	destroy	delete

20. Rails "simulates" PUT and DELETE requests. Why?

    Rails "simulates" PUT and DELETE requests, because some servers don't support those requests.

21. What is the difference between the two Rails environments: 'production' and 'development'?

    The development environment is where you create the application. You are running a virtual server, only when you need it, and you are most likely using SQLite as a query language. The server updates automatically upon most file saves, and provides you with errors, and where those errors are, when something goes wrong. The production environment is where you are releasing the application to the internet/intranet, where anyone who can see your server can access it. You will be running some production server like Apache, and most likely you will be using MySQL as a query language.

22. Usually, Rails's controllers incorporate plural nouns, such as `ProtestsController` and `RevolutionsController`. In what case should a controller have a singular name like `GeocodingController`?

    You should use a singular name for the controller when it's for something the user won't be looking up by id. For example a logged in user will have his/her own wishlist, so the user won't access it by id, they will just access wishlist.

23. What is a Rails "helper method" and when should they be defined and used by you, the developer?

Rails helper methods are functions that will help you remove logic from the view. Rails has some built in helper methods such as `number_to_currency()`, and allows the developer to define helper methods. As mentioned above, helper methods should be used to remove logic from the view. An example of this in action is when I needed to sort articles by date_created, so I made a helper method to do this for me, so I could display the sorted array without having to have any sorting logic in my view.

Assume you have a `Flower` AR class that `has_and_belongs_to_many :bees`, and a `Bee` class that `has_and_belongs_to_many :flowers`.

24. What must exist in the database schema in order for AR to infer the proper foreign key/relationship?

    There must exist a relationship table that will store in each row the key of a flower and the key of a bee, showing that that flower and that bee are related.

    Assume that a Bee `:belongs_to` a Hive and a Hive `has_many :bees`. Also assume a GET request is sent to the `FlowersController#show` action, which contains a finder method call `@flower = Flower.find(params[:id])`. Assume the view `app/views/flowers/show.html.haml` displays the name of the `Flower` and each `Flower`'s bee's name and Hive name like so:

    ```
    -@flower.bees.each do |b|
      %h1 = b.name
      %p = b.hive.name
    ```

    If you were tailing the log of your application during the rendering of the response, you would notice tons of database queries.

25. Are all of those queries ok? If so, explain why. If not, explain how you would reduce the number of database queries (without hand-rolling your own SQL query).

    All of those queries are valid. As for okay, probably not, since that puts a lot of strain on the server, but I can't think of another way to do it, so I guess yeah it's okay. I'm going to say it's okay, because there isn't another way to do it. This won't get rid of having to do all those queries, but once you've performed all the queries you can cache the result, so you won't have to do all the queries over again.