1. (8 points) Identify all of the following statements that are true about the basics of services.

   A. When implementing a web service, we can promote its interoperability by directly parsing a URI string used for making a request, instead of relying on modeling languages such as RDF to give long descriptions of requests

      Solution: False: becoming reliant on parsing a URI makes it harder to maintain loose coupling of the service with its consumers; programmers should work at higher levels of abstraction

   B. Autonomy refers to the fact that entities (e.g., business partners) control their own resources and actions

      Solution: True:

   C. Modeling context provides a basis for intelligent decision making while reducing the need for explicit user input

      Solution: True:

   D. Skype demonstrates an open architecture

      Solution: False: No public protocol that another implementation could use

2. (10 points) Identify all of the following statements that are true about conceptual modeling, RDF, and OWL

   A. The main benefit of RDF and RDF Schema is to help define custom vocabularies

      Solution: True:

   B. Making a property assertion about a resource automatically asserts the resource as being of the type of the domain of the property

      Solution: True:

   C. From assertions that a property has domains $D_1$ and $D_2$, respectively, we can conclude that the property’s domain is $D_1 \cup D_2$

      Solution: False:

   D. OWL is an example of an RDF Schema vocabulary with a customized interpretation of the terms

      Solution: True:

   E. A functional property may have zero range instances for a domain instance

      Solution: True:

   F. A functional property may have multiple range instances for the same domain instance

      Solution: False:
3. (20 points) You are developing a service for helping select Valentine's Day gifts. You need to create a value map from a scale of prices to a scale of user preference. For your target users, gifts that are more expensive are more desirable, but only up to a point (say, $80); gifts that are more expensive than $80 become less and less appropriate.

- User rankings (best to worst): A, B, C, D, F
- Prices (in $): 0–20, 20–40, 40–60, 60–80, 80–100, 100–200, 200–500, 500–\(\infty\)

Devise a pair of mappings from rankings to prices and from prices to rankings that are total in each direction and consistent inverses of each other, or show that such mappings are impossible. Use the style of the figures used in the slides. That is, show points and directed edges. Explain in a few words why your solution satisfies the stated requirement or why no solution can satisfy the stated requirement.

**Solution:**

```
A

B

C

D

F

0–20

20–40

40–60

60–80

80–100

100–200

200–500

500–\(\infty\)
```

4. Produce a model in two parts capturing a vocabulary that formalizes a fragment of a language for goal modeling:

1. An actor adopts zero or more goals.
2. A goal is adopted by exactly one actor.
3. A goal can have zero or more decompositions.
4. A decomposition of a goal can be an AND-decomposition or an OR-decomposition.
5. A decomposition of a goal involves one or more goals.
6. A goal can be atomic or composite.
7. A goal can be atomic or composite but not both.
8. A goal must be atomic or composite.
9. An atomic goal has no decomposition.
10. A composite goal has at least one decomposition.

If you like, you can use the following abbreviations: actor (A); goal (G); atomic goal (AG); composite goal (CG); decomposition (D); AND-decomposition (AD); OR-decomposition (OD).

You can express your solution as a graph with suitable text annotations and associated constraints. For example, if you wish to specify that Foo is a class, you can show an edge labeled type from Foo to Class. Similarly, you can show that something is a property. However, for some elements of your solution, you may need to write a textual description. The syntactic details are not important as long as you approximate one of the languages discussed in class and I can understand your answer.

Hints. I include some hints below that are based on common errors I have seen.

- Use RDF (and RDF Schema) and OWL constructs, as asked for. Don’t make up constructs on your own. Don’t write complex English expressions (strings) in class and property labels and expect that such expressions will be interpreted as you hope: they won’t, because arbitrary strings are not constructs of our formal languages.
- Differentiate classes from instances. In OWL DL in general and in this problem, nothing should be both a class and an instance.
- If you are using instances, you are doing something wrong.
- Understand and use functional and similar constructs properly.
- Understand and use OWL restrictions properly. A restriction is a way to create a class; it does not alter a property.
- Avoid redundant cardinality assertions.
- Identify in your solution where it handles each of the above 10 constraints. Some constraints may be handled through a combination of two or more assertions.
- Identify what can work with just RDF and what needs OWL.
- You can produce your answer entirely in OWL (as a text document) if you like.

(a) (20 points) Capture as many of the above assertions as possible using RDF and RDF Schema constructs.

**Solution:**
In the picture below, solid ellipses are classes and dashed ellipses are properties (for OWL, understand them as Object Properties).
The OWL-specific constructs are restriction, union, inverse of, disjoint with, and equivalent class.
(b) (20 points) Capture as many of the remaining assertions as possible using OWL constructs.

**Solution:** The OWL-specific constructs are restriction, union, inverse of, disjoint with, and equivalent class.