CS312 Fall 2023 – Midterm 1 Solution

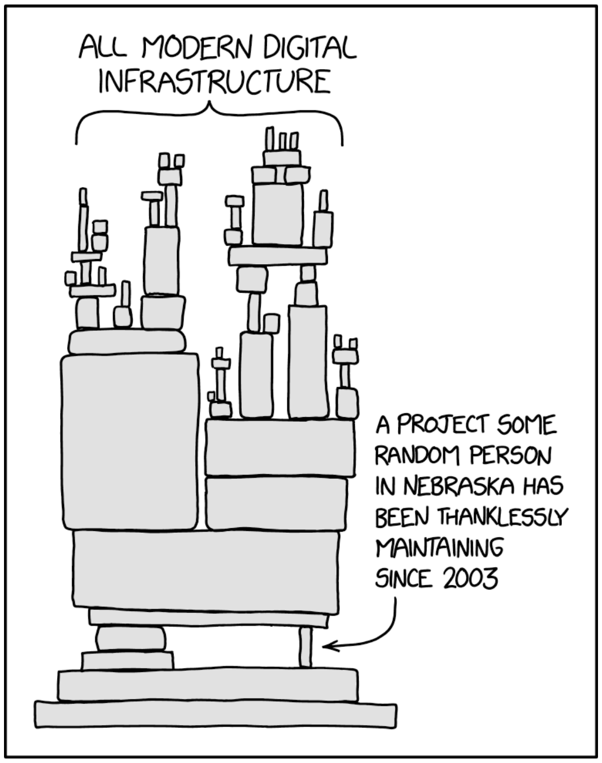
Name: ______________________________

Date: ________________ **Start time:** ______________ **End time:** ______________

Honor Code:

Signature: ________________________

This exam is open course web page, open Ed, open notes, open slides, open your assignment solutions and open calculator, but closed everything else (e.g., consulting with others and searching online are not permitted). **You have 2 hours in a single sitting to complete the exam.** Read the problem descriptions carefully and write your answers clearly and *legibly* in the space provided. Circle or otherwise indicate your answer if it might not be easily identified. You may use extra sheets of paper, stapled to your exam, if you need more room, as long as the problem number is clearly labeled and your name is on the paper. If you attached extra sheets indicate on your main exam paper to look for the extra sheets for that problem.


ALL MODERN DIGITAL INFRASTRUCTURE
A PROJECT SOME RANDOM PERSON IN NEBRASKA HAS BEEN THANKLESSLY MAINTAINING SINCE 2003

Learning Target	Assessment
1	
2	
3	
4	
5	
6	
7	
8	

Question 1. User stories

You are developing a web application for grading programming assignments (like Gradescope). When interviewing stakeholders, multiple respondents described wanting to indicate that a particular submission (of potentially many) is ready for review ahead of the due date. Write two I.N.V.E.S.T. user stories for this feature, one from the perspective of a student submitting their work, the other from the perspective of an instructor reviewing that submission. Your user stories will be evaluated on format and quality.

(a) Student:

Solution: As a student,
I want to tag a submission as ready for review
So that I get prompt feedback on my work.

(b) Instructor:

Solution: As an instructor,
I want to know that a particular submission is ready for review
So that I can promptly review final submissions and not review in-progress submissions.

Question 2. Javascript

Assume the following functions: `any(promises)` returns a promise that will resolve when the *first* of the promises `promises` has resolved, with the value of that first promise (the remaining promises are not canceled, but continue executing). `all(promises)` returns a promise that will resolve when *all* the promises in `promises` have resolved, with an array of the fulfilled values of `promises`. `wait(sec)` returns a promise that resolves after `sec` have elapsed.

```
function do(time) {
  return wait(time).then(() => {
    console.log(time)
  });
}

all([do(1), do(2), do(3)]).then(() => {
  console.log(4)
});
```

```
async function do(time) {
  await wait(time);
  console.log(time);
}

await any([do(1), do(2), do(3)]);
console.log(4);
```

Consider the two code snippets above. Write the expected output for left-side code below on the left. If the right-side code produces the same result indicate below, otherwise provide the expected output below on the right.

◯ Both snippets produce the same output

Solution:

```
1
2
3
4
```

Solution:

```
1
4
2
3
```

Question 3. Testing

You are developing a React component for setting reminders for some time in the future from now, e.g., "Remind me to check the potatoes in 30 minutes". A reminder object has a `message` string and a `time` string storing the specific date and time for the reminder as an ISO-formatted string (e.g., `"2023-01-01T12:30:00Z"`). You have implemented a `ReminderCreator` component that contains a text input for the message and numeric inputs for the number of days, hours, minutes and seconds to schedule the reminder in the future. When the user clicks a "Create" button, the component invokes a callback with the reminder object. Using the skeleton below, implement **pseudo-code** for a F.I.R.S.T. unit test to verify that when the user clicks the "Create" button, the callback is invoked with the correct reminder object. You do **not** need to provide executable Javascript, instead describe the steps of your test as pseudo-code. For example, one of the steps in your pseudo-code might be:

Assert mock function was not called

You may or may not need all of the functions below. You only need to include pseudo-code in bodies of the functions relevant to your answer.

```
describe("Reminder creation", () => {
  beforeEach(() => {
```

Solution:

```
Mock Date.now() to return a fixed value of Jan. 1. 2023 12:00PM.
```

```
  });
  afterEach(() => {
```

Solution:

```
Clear mock
```

```
  });
  test("Creating reminder invokes callback", () => {
```

Solution: An integration test would need to involve both the `Editor` component and the parent component that performs the `fetch`.

```
Create a mock function for the callback
Render the ReminderCreator component with the mock callback as a prop
Find and fill the message input with "Check the potatoes"
Find and fill the minutes input with 30 and the days, hours and seconds inputs with 0
Find "Create" button and simulate click.
Assert mock callback was invoked with object
  containing message "Check the potatoes" and time of Jan. 1. 2023, 12:30PM.
```

A common issue was not specifying what the object passed to the callback should look like (i.e., what was a "correct" argument). The test will need to verify the argument contains the expected message and time. Since the input is relative time (e.g., minutes from now) and the component translates that to absolute time, the test should assert the absolute time is correct (by mocking the current time, less robustly, attempting to calculate the expected time).

```
  });
});
```

Question 4. Scenarios

An “accordion” is a common UI element with multiple panels that when clicked, “open” to show additional detail, and when clicked again “close” to show just the title. In your particular `Accordion` component, all panels are closed by default, and only one panel can be open at a time, i.e., clicking on a second panel’s title should collapse the first panel. Write a Gherkin-style test scenario for the “exclusivity” behavior of this component. You do not need to provide the implementation details of the tests, just describe the scenario for the test.

Solution:

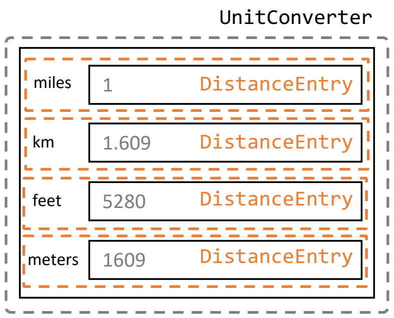
We want to make sure that the first panel opens exclusively, and then clicking another panel opens the new panels exclusively (i.e., the first panel closes).

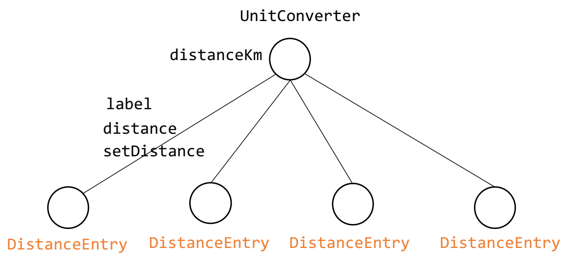
```
Given an accordion with Panel1 and Panel2,
Then Panel1 and Panel2 should be closed,
When I click on the title for Panel1,
Then Panel1 should be open and Panel2 should be closed,
When I click on the title for Panel2,
Then Panel1 should be closed and Panel2 should be open.
```

Question 5. React

You are implementing the unit converter below with React. Entering a distance in any of the text inputs should automatically update the others with the correctly converted distance. Outline and label the wireframe (below, left) with a possible set of components. Label the tree (below, right) with components to show the hierarchy. Further label the tree nodes with state implemented in that component and label the tree edges with props passed to each component (similar to the figure in programming assignment 2). The top-level component `UnitConverter` is labeled for you. Any implementation reflecting good React practices will be accepted. You may not need all the nodes in the tree; cross out any unused nodes. Your component, state and prop names should be sufficiently descriptive that their role is clear.

Solution:





Each child has the same props, but is passed values specific to that unit, e.g.

```
<DistanceEntry
  label="miles"
  distance={distanceKm * 0.621371}
  setDistance={(dist) => setDistanceKm(dist / 0.621371)}
/>
```

or a conversion factor prop applied internal to the component. Since all inputs are modifying/displaying the same underlying distance there should be only one piece of state (one "source of truth") maintained in the nearest common ancestor, `UnitConverter`.

Question 6. REST

For each of the following pages in a NextJS-based web application, provide an appropriate RESTful front-end (browser) URL for that page and, where relevant, an appropriate RESTful server API endpoint (HTTP verb and URL) that component would interact with. An example is provided below.

Page	Page URL	API HTTP verb and URL
Add a new article to Simplepedia	**/edit**	**POST /api/articles**
View a specific product on an e-commerce site	**/products/1**	**GET /products/1**
Create a new review for a product on an e-commerce site	**/products/1/reviews/new**	**POST /products/1/reviews**
Show only “verified” reviews for a product	**/products/1/reviews?verified**	**GET /products/1/reviews?verified**

Question 7. Data modeling

Assume you are developing a web application for managing physical room reservations by specific people at an institution with multiple rooms and multiple people who might reserve rooms. You will be using a relational database to store the data for this application.

(a) Identify the *minimum* set of models you would define in your server backend to implement the following user story:

As a person hosting an event, I want to be able to view the rooms with sufficient capacity that are available (i.e., not already booked) at a specific date and time without logging in, so that I can quickly identify possible locations for my event.

Solution: The minimum models would be `Room` and `Reservation`. The `Room` model would maintain information about all available rooms and their capacity (i.e., capacity would be an attribute of `Room`) and `Reservation` would record existing bookings for rooms. Since the user doesn't need to want to be logged in, and we don't need to know who made the existing reservations, a `User` model is not need for this user story. Answers with or without an `Institution` model were accepted.

(b) Choose ONE answer. You are developing your application to support many different institutions at the same time. To do so you implement an `Institution` model. Which of the following associations between `Institution` and `Room` would be appropriate for this application?

- ○ A one-to-one association between `Institution` and `Room`.
- √ **A one-to-many association between `Institution` and `Room`, i.e., an `Institution` has many `Rooms`.**
- ○ A one-to-many association between `Room` and `Institution`, i.e., a `Room` has many `Institutions`.
- ○ A many-to-many association between `Institution` and `Room`.

Solution: An institution can have many rooms, but a room can only belong to one institution.

(c) In a normalized schema designed for a relational database (RDBMS), what are attributes that would be needed to support a user reserving a room for specific window indicated by a start and end time. Assume a reservation is owned and manged by a single user. You do not need to provide SQL, just the attributes, their types, the primary key, and any foreign key constraints.

Solution: `Reservation` has a one-to-many relationship to `Room` and `User`. Alternately we could think about `Reservation` as a join model linking `Room` and `User`. Thus, `Reservation` will need integer foreign keys for `Room` and `User`. In addition it will need the starting and ending timestamps, and since the same user would reserve the same room at different times, we will use a separate integer primary key (instead of creating a composite primary key).

The intent was to define all three models, although answers that described the necessary attributes of the `Reservation` model were accepted.

Question 8. Development processes

For each of the following, indicate whether the action would be consistent with the best practices for software development as described in class or not consistent. Briefly explain your answer.

(a) Activate branch protection on GitHub, so a Pull Request is required to merge changes into `main` (i.e., a direct push to `main` is not allowed).

[✓ **Consistent**] ○ Not consistent

> **Solution:** Our best practices are to complete all merges to `main` via pull request (to allow for review) and so this action is consistent with and would help enforce those best practices.

(b) Designate one team member as responsible for executing all deployments to the production environment (e.g., csci312.dev).

○ Consistent [✓ **Not consistent**]

> **Solution:** Our best practices are to work as a cross-functional team. This action would create a ”silo” within the team that could be a bottleneck. Further, it would run counter to our ”DevOps” approach, by effectively creating a separate ”operations” team.

(c) Designate some user stories in the sprint backlog, as “multi-sprint”, i.e., stories that are expected take more than one sprint to complete.

○ Consistent [✓ **Not consistent**]

> **Solution:** The sprint backlog is defined as the set of user stories and other tasks to be completed in the next sprint. If user story is expected to take multiple sprints it should be broken down into smaller stories.