CS312 Spring 2023 – Midterm 2 Solution

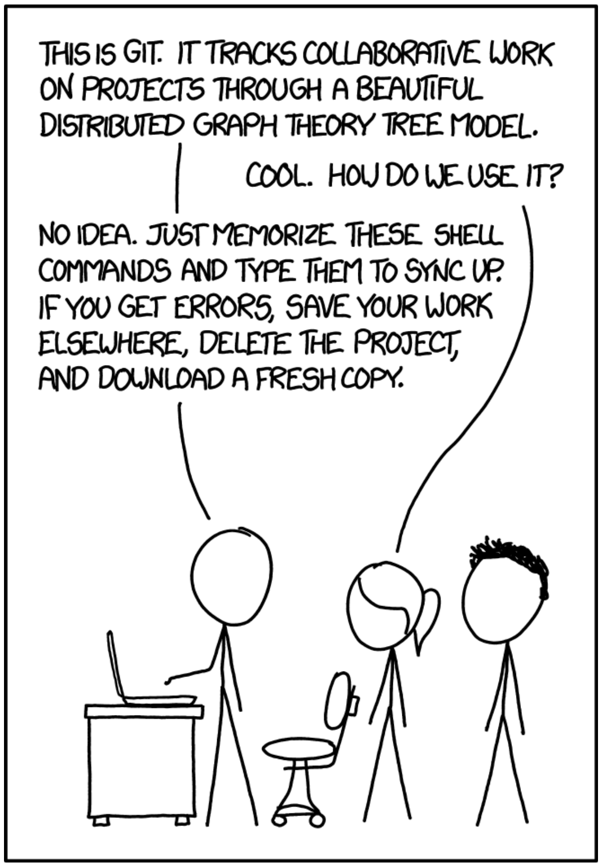
Name: ______________________________

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 3 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.


THIS IS GIT. IT TRACKS COLLABORATIVE WORK ON PROJECTS THROUGH A BEAUTIFUL DISTRIBUTED GRAPH THEORY TREE MODEL.
COOL. HOW DO WE USE IT?
NO IDEA. JUST MEMORIZE THESE SHELL COMMANDS AND TYPE THEM TO SYNC UP. IF YOU GET ERRORS, SAVE YOUR WORK ELSEWHERE, DELETE THE PROJECT, AND DOWNLOAD A FRESH COPY.

Learning Target	Assessment
1	
2	
3	
4	
5	
6	
7	
8	

Question 1. User stories

You are developing a web application for managing the queue for in-person office hours appointments. When interviewing stakeholders, multiple respondents described e-mail notifications for individuals as they approached the front of the queue. Write two I.N.V.E.S.T. user stories for this feature, one from the perspective of a student in the queue for office hours, the other from the perspective of an instructor holding office hours. Your user stories will be evaluated on format and quality.

(a) Student:

Solution:
As a student,
I want receive a notification when I am approaching the front of the queue
So that I don't need to wait outside the office but instead can walk over when my turn approaches.

(b) Instructor:

Solution:
As an instructor,
I want waiting students to automatically receive notifications as they approach the front of the queue
So that students don't need to wait outside the office, but are ready for their turn.

Question 2. Javascript

Assume `wait(sec)` returns a promise that resolves after `sec` seconds have elapsed:

```
function first() {
  return wait(3).then(() => console.log(1));
}

function second() {
  return wait(2).then(() => console.log(2));
}

function third() {
  return wait(1).then(() => console.log(3));
}
```

(a) What will the following code print?

```
first();
second();
third();
```

Solution:

```
3
2
1
```

(b) Using only the three functions above and either the Promise API, e.g., `then` methods, or `await`, write code that prints 1, 2, 3 in that order. You must use the 3 functions above without modification. No additional `console.log` statements or other code that isn't `then` with a callback or `await` is allowed.

Solution:

Solutions using Promise API and `await`:

```
first()
  .then(() => second())
  .then(() => third());
```

```
await first();
await second();
await third();
```

Question 3. Testing

Imagine you are developing a reminder application, where each reminder object has a `description` string and `dueDate` date. You have implemented a component, `ReminderCreator`, containing input elements and a "Create" button to create new reminders. When the user clicks "Create", the component invokes a callback with the reminder object. If the user doesn't provide an item description or enters a date in the past, the "Create" button is disable and the form provides error feedback. Using the skeleton below, implement **pseudo-code** for F.I.R.S.T. integration tests to verify that creating a valid reminder invokes the callback, and that a reminder with a past date disables the "Create" button and results in error feedback. We measure error feedback as the date input element having the attribute `"aria-invalid"` set to `"true"`. 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 answer might be:

Assert date input element has attribute "aria-invalid" with value "true"

```
describe("Reminder creation", () => {
  test("Creating valid item invokes callback", () => {
```

Solution:

```
Create mock function for callback
Render ReminderCreator passing mock as the callback
Find and fill the item text with "Test item" and set the date to Date.now() + 1 day
Find the "Create" button and simulate a click
Assert mock callback called with object containing "Test item" and a due date of Date.now() + 1 da
```

```
  });
  test("Past date shows error feedback", () => {
```

Solution:

```
Create mock function for callback
Render ReminderCreator passing mock as the callback
Find and fill the item text with "Test item" and set the date to Date.now() - 1 day
Find the "Create" button and assert it is disabled
Find and assert date element has attribute "aria-invalid" with value "true"
Assert mock callback was not called
```

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

Question 4. Scenarios

Imagine you are writing a React component to display a list of products in an e-commerce store. Your component has a dropdown to display the items sorted in ascending order of cost (low to high) or descending order of cost (high to low). Write a Gherkin-style test scenario that covers this behavior. 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 to the test toggles between the two orders, and back again.

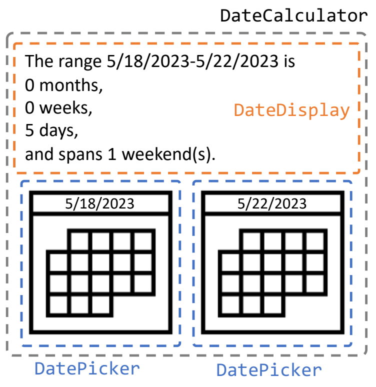
```
Given the is a list of two products, $10 t-shirt and $15 pants,
  and the display order is ascending,
Then the products should be displayed as $10 t-shirt, then $15 pants
When I select descending order
Then the products should be displayed as $15 pants, then $10 t-shirt
When I select descending order
Then the products should be displayed as $10 t-shirt, then $15 pants
```

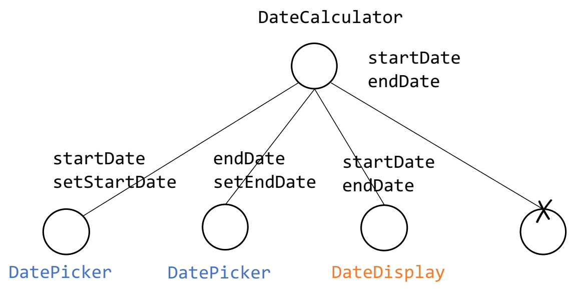
Question 5. React

You are implementing an application for displaying statistics about date ranges in React. Outline and label the wireframe (below, left) with a possible set of components. Label the tree (below, right) with components to show the hierarchy, *including* labeling the tree nodes with state implemented in that component and labeling the tree edges with props passed to each component (similar to the figure in programming assignment 2). The top-level component `DateCalculator` is labeled for you. Assume you have a Javascript library that can compute statistics for arbitrary date ranges and that your component library already provides a controlled calendar-based `DatePicker` component (like shown in the wireframe). 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:

Good design would hoist the date state to the `DateCalculator` component where it can be used by the display (and set by the pickers).





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 is below.

Page	Page URL	API HTTP verb and URL
Add a new article to Simplepedia	**/edit**	**POST /api/articles**
View a patron's loaned (checked-out) items in a library management application	**/patrons/1/loans**	**GET /api/patrons/1/loans**
Renew a patron's loaned item to extend the due date in a library management application	**/patrons/1/loans/1/renew**	**PUT /api/patrons/1/loans/1**
Search results for items in a library collection with "fire" in the title	**/items/search?title="fire"**	**GET /items/search?title="fire"**

Question 7. Data modeling

Assume you are developing an application for managing class rosters at an institution where students take multiple classes and faculty members teach multiple classes, but each class is only taught by one faculty member.

(a) Identify the models you would define in your server backend to implement the following user story: As a faculty member, I want to be able to view my current class roster, so that I know who is enrolled in my class

> **Solution:** At a minimum we would want `User` (with an attribute for student/faculty) and `Course`. Students would be linked with a Course via a join table. Answers with and without a Model for the join table, or with separate `Student` and `Faculty` models were accepted.

(b) Choose ONE answer. In a normalized schema designed for a relational database (RDBMS), how would you best store link between faculty members and the classes they teach?

- ○ An array of class IDs stored in a `Faculty` table row
- ○ An array Faculty IDs stored in the `Class` table row
- ○ A single `Class` ID stored in a `Faculty` table row
- [✓ **A single `Faculty` ID stored in the `Class` table row**]

> **Solution:** A class has one faculty member, but a faculty member has many classes. We put the foreign key in the "many" side of the relation, i.e., in `Class`.

(c) How would your answer to the part above change if a single class could be taught by multiple faculty members with different factions of responsibility (i.e., faculty member one is 75% responsible and faculty member two is 25% responsible)? Describe the association that models the relationship *and* note any attributes needed to support this feature. Your answer should use the association vocabulary from class.

> **Solution:** We would describe this as a many-to-many relationship, i.e., a `Class` has many `Faculty` and a `Faculty` teaches many `Class`es. We would implement this relationship with a

join table, e.g., `Teaches`, that include the class and faculty IDs, and also the relative responsibility as an attribute.

Question 8. Development processes

For each of the following series of commands, indicate whether it is consistent with our in-class development and deployment processes or not. If not, briefly explain why those actions would be problematic.

```
git checkout -b edit_article
...
git add .
git commit -m "New page ..."
git push origin edit_article
```

[✓ **Consistent with class practices**]

◯ *Not* consistent with class practices

```
git checkout main
...
git add .
git commit -m "New page ..."
git push origin main
```

◯ Consistent with class practices

[✓ ***Not* consistent with class practices**]

Solution: We don't directly push new features to main, instead we create a feature branch that is merged in via pull request.

```
git checkout -b edit_article
...
git add .
# --no-verify bypasses pre-commit hooks
git commit --no-verify -m "New page ..."
```

◯ Consistent with class practices

[✓ ***Not* consistent with class practices**]

Solution: We don't want to bypass formatting and linting checks in pre-commit hooks as that can let errors slip through or result in consistent formatting.

```
git checkout -b edit_article
...
git add .
git commit -m "New page ..."
git fetch
git merge origin/main
git push origin edit_article
```

[✓ **Consistent with class practices**]

◯ *Not* consistent with class practices

Page intentionally blank.