CS312 Spring 2019 – Midterm

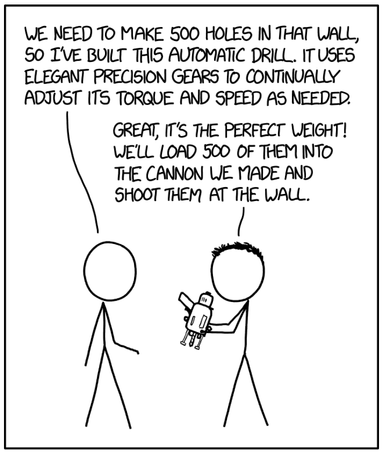
Name: ______________________________

Date: ____________ **Start time:** ____________ **End time:** ____________

Honor Code:

Signature: ______________________________

This exam is closed book, closed notes, closed computer, etc. You may only use a single double-sided letter-sized sheet of notes. **You have 2 hours in a single sitting.** 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.


WE NEED TO MAKE 500 HOLES IN THAT WALL, SO I'VE BUILT THIS AUTOMATIC DRILL. IT USES ELEGANT PRECISION GEARS TO CONTINUALLY ADJUST ITS TORQUE AND SPEED AS NEEDED.
GREAT, IT'S THE PERFECT WEIGHT! WE'LL LOAD 500 OF THEM INTO THE CANNON WE MADE AND SHOOT THEM AT THE WALL.
HOW SOFTWARE DEVELOPMENT WORKS

Question	Points	Score
JavaScript	11	
React	18	
Agile Practices	19	
Client Server	14	
Data Modeling and Databases	12	
Total:	74	

Question 1: JavaScript [11 points]

(a) Describe the type and length of the return value for each of these methods invoked on an `Array` of length `n`, e.g. `slice` returns an `Array` of length $\leq$ `n`.

i. `map` ____

ii. `reduce` ____

iii. `filter` ____

iv. `forEach` ____

(b) Choose ONE answer. Helen is creating a JavaScript (JS) application to keep track of her favorite books and plays. She implemented separates JS classes for `Book` and `Play`, because even though both have a `title` and `author`, they are otherwise very different: for example, a `Play` has a `numberOfActs` and other properties that don't apply to a `Book`. Helen needs to sort a combined array of books and plays by their title. What design should she use to solve this problem in JS?

- ◯ `Book` and `Play` must be subclasses a common class, e.g. `Manuscript` that has the `title` property.
- ◯ As long as `Book` and `Play` both have an attribute named `title`, Helen can use the existing `Array.sort` method on the combined array.
- ◯ Helen must create a custom sort function that can handle the combined array by checking the type of each item as part of the comparison function.
- ◯ Helen must separate the collection into two collections (one containing only Books and the other only Plays), sort each collection, then merge the two sorted collections.

(c) Choose ONE answer. Which of the following Jest features helps us implement tests that satisfy the "I" in F.I.R.S.T.?

- ◯ The "watcher" that automatically runs the tests when a file changes
- ◯ The many different matchers, e.g. `toBeFalsy`.
- ◯ Multiple `expect` calls in a single test
- ◯ The `beforeEach` and `afterEach` functions

(d) Choose ONE answer. Assume `articlesToSections` returns a sorted array of Simplepedia section labels for an array of articles. The following test

```
expect(articlesToSections(samplesArticles)).toEqual(['A','B']);
```

is an example which of the following kinds of testing?

- ◯ Unit testing
- ◯ Integration testing
- ◯ End-to-end testing

(e) `waitSeconds(sec)` returns a promise that resolves after `sec` seconds have elapsed. Write example valid output from running this code with `node`. Make sure to include the correct label, e.g. "Delay 1", with each statement.

```
waitSeconds = (sec) => {
  return new Promise(resolve => setTimeout(resolve, sec * 1000));
};
elapsedSeconds = (start) => Math.floor((Date.now() - start) / 1000);

let current = Date.now();
waitSeconds(2)
  .then(() => {
    console.log(`Delay 1: ${elapsedSeconds(current)}s`);
    return waitSeconds(3);
  })
  .then(() => {
    console.log(`Delay 2: ${elapsedSeconds(current)}s`);
    return 4;
  })
  .then(() => {
    console.log(`Delay 3: ${elapsedSeconds(current)}s`);
  });
console.log(`Delay 4: ${elapsedSeconds(current)}s`);
```

Question 2: React [18 points]

(a) There are several correctness problems with this React to-do list component (i.e. not just poor style). Identify three (3) such problems. Each problem should be distinct, not just a variation on the same problem. You do *not* need to provide a fix for the problems you identify.

```
import React, { Component } from 'react';
class ToDo extends Component {
  constructor(props) {
    super(props);
    this.state = { truncate: false, newItem: '' };
  }
  render() {
    if (this.props.items.length > this.props.numToShow) {
      this.setState({ truncate: true });
    }
    const showItems = this.state.truncate ?
      this.props.items.slice(0, this.props.numToShow) :
      this.props.items;
    return (<div>
      <ul>{showItems.map(item => <li key={item}>{item}</li>)}</ul>
      <p onClick={() => {
        this.setState(prevState => ({ truncate: !prevState.truncate }));
      }}>...</p>
      <input value={this.state.newItem} onChange={(evt) => {
        this.state.newItem = evt.target.value;
      }}/>
      <button onClick={() => {
        this.props.items.push(this.state.newItem);
      }}>Add</button>
    </div>);
  }
}
```

(b) The `ToDo` component is designed for quick input, with a text input and an Add button at the bottom of the component. To support data persistence, the application sends new to-do items to a server via the `fetch` function. Imagine that a teammate has already implemented a suitable mock for `fetch`. Ignore the current (broken) implementation, and think about performing integration testing based on how the user would experience adding a new item.

i. Using pseudo-code or detailed text, explicitly describe each step of a test of adding a new item to the to-do list.

ii. The `ToDo` component has a feature that restricts the number of visible elements. Propose two (2) additional tests for your test suite that cover this behavior. You do *not* need to provide implementation details of these tests, just summarize the behavior that needs to be tested.

(c) As part of your to-do list application you are developing a separate `DatePicker` component that provides a calendar view to help your users pick "due dates" for their to-dos. `DatePicker` will be rendered next to the text input field. Briefly describe what state you will need and where that state will be maintained.

Question 3: Agile Practices [19 points]

(a) The requirements specification in Plan and Document corresponds to what Agile practice(s)?

(b) Choose ONE answer. You have an idea for a new feature for your project. According to our CS312 development process, what is your next step?

- ◯ Implement a spike to see if it will work
- ◯ Add it to the project backlog
- ◯ Draw a "lo-fi" storyboard
- ◯ Write a user story
- ◯ Pitch it at the next stand-up

(c) In the Scrum approach, fill in who has responsibility for each of the following tasks:

i. Prioritizing features ____________________

ii. Facilitating the sprint planning meeting ____________________

iii. Updating the sprint backlog during the sprint ____________________

iv. Determining who implements which backlog items ____________________

(d) Briefly describe three (3) motivations for creating a pull request (PR) as part of our project workflow.

(e) Colors in CSS are typically specified in hex, e.g. `#0D395F` for Panther blue.

i. Write a user story in Connextra format for a new feature for the color picker that displays the current color in hex.

ii. Write a Gherkin-style scenario for this feature that incorporates user interaction.

Question 4: Client Server [14 points]

(a) You are developing a web application with a contact information form. One of your teammates notices that the client-side code validates the inputs (makes sure fields are present, checks that email addresses and phone numbers are in valid formats) before sending it to the server, which then performs essentially the same validations. Your teammate suggests removing either the server- or client-side validation steps to DRY out the codebase and make it more maintainable. Make the counter argument that advocates leaving both server- and client-side validation in place by providing advantages for each step that can't be duplicated by the other.

(b) Select ALL that apply. Given the HTTP request `GET http://www.example.com:8000/search?q=cs312`, which elements of that request does Express use to determine which handler to execute?

- ◯ GET
- ◯ www.example.com
- ◯ :8000
- ◯ /search
- ◯ q=cs312

(c) For each of the following actions in a web application provide a appropriate RESTful HTTP verb and URL, e.g. `GET /api/articles/3`.

i. Read the data for a single to-do list item

ii. Create a new movie

iii. Update a single to-do list item

iv. List all of a user's movie ratings

Question 5: Data Modeling and Databases [12 points]

(a) Select ALL that apply. Which of the following statements are true regarding the use of migrations to manage the database schema of a SaaS app?

- ◯ Migrations allow "versioning" the database schema analogously to how code is versioned.
- ◯ Migrations can be expressed in a way that is independent of minor syntactic differences between different underlying databases.
- ◯ Migrations may change the schema, but they never result in destroying or discarding data.

(b) You are writing a simple blogging application in a which users can write posts and comment on posts that they or others wrote. As with Simplepedia, a post should have a `title`, `body` and `edited` time. A comment should have a `body` and `edited` time.

i. Draw the CRC cards for the nouns in this application focusing just on posting and commenting.

ii. What are the association(s), if any, between the various nouns?

Page purposely left blank.