CS 343: Application Development
Practice Exam 2

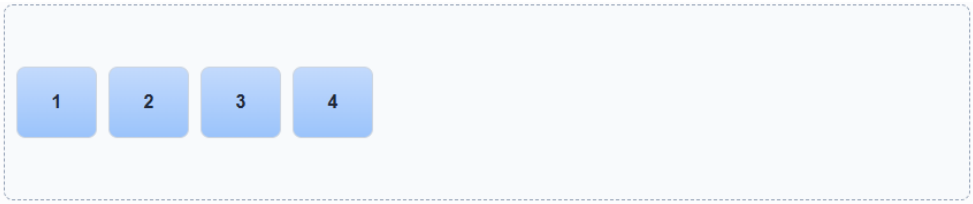
James Madison University
Spring 2026

Name: _____ JACard #: _____

Question 1. CSS Layout
(20 pts)

Given each image, complete the CSS to implement the layout. Assume each item has a fixed size and the parent container has the container class. You do not need to do any styling (color, spacing, etc.).

a. (5 pts)

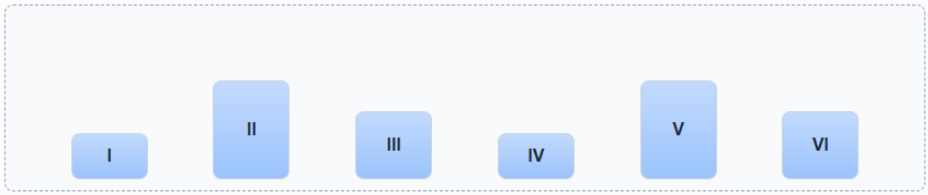



```
.container {
    display: flex;
    flex-direction: row;

    justify-content: flex-start;
    align-items: center;


}
```

b. (5 pts)




```
.container {
    display: flex;
    flex-direction: row;

    justify-content: space-evenly;
    align-items: flex-end;


}
```

c. (10 pts)

Given the image of the desired layout and the HTML, write the CSS rules to implement the layout. You do not need to do any styling (color, spacing, etc.).


```
<div class="container">
  <div id="a">A</div>
  <div id="b">B</div>
  <div id="c">C</div>
  <div id="d">D</div>
</div>
```

```
.container {
    display: grid;
    grid-template-columns: 1fr 1fr 1fr;  // OR: 33.3% 33.3% 33.3%
}

#a {
    grid-column-start: 1;
    grid-column-end: 3;
    // OR: grid-column: 1 / 3;  // OR: 1 / span 2;
}

#b {
    grid-column: 3;
    grid-row-start: 1;
    grid-row-end: 3;
    // OR: grid-row: 1 / 3;  // OR: 1 / span 2;
}
```

Question 2. Array Methods
(20 pts)

For the following problems, you may only use combinations of *.map()*, *.filter()*, *.reduce()* to solve the problem. You **may not** use any loops (*for*, *while*, or even *.forEach()*).

Full credit will only be given for using the appropriate array method(s) for each problem *and* using them appropriately.

a. Write a function *onlyEven* that takes an array of numbers and returns a new array containing only even values.

```
function onlyEven(nums) {

    return nums.filter(n => n % 2 === 0);




}
```

b. Write a function *addOne* that takes an array of numbers and returns a new array where each number is increased by 1.

```
function addOne(nums) {

    return nums.map(n => n + 1);




}
```

c. Write a function *nameLengths* that takes an array of strings and returns the **sum** of all their lengths.

```
function nameLengths(names) {

     return names.map(name => name.length)
                 .reduce((acc, x) => acc + x);




}
```

d. Given an array of objects representing vehicles, like the following:

```
[
    { make: "Toyota", model: "Camry", year: 2025 },
    { make: "Honda", model: "Civic", year: 2018 },
    ...
]
```

Write a function *onlyNew* that takes an array of such objects and returns a new array containing only the **models** of vehicles that were made in 2019 or later.

```
function onlyNew(vehicles) {

    return vehicles.filter(vehicle => vehicle.year >= 2019)
                   .map(vehicle => vehicle.model);




}
```

Question 3. Fetch and External APIs
(30 pts)

Assume you have an HTML list of track IDs (representing different songs from Spotify):

```
<body>
    ...
    <ul class="trackList">
        <li>3n3Ppam7vgaVa1iaRUc9Lp</li>
        <li>7qiZfU4dY1lWllzX7mPBI</li>
        <li>0VjIjW4GlUZAMYd2vXMi3b</li>
        ...
    </ul>
    ....
</body>
```

Given the API specification in Attachment 1, implement an asynchronous function, **`saveSongs()`**, which must get all the track IDs and make an API request to save tracks for the current user.

```
async function saveSongs() {
    // Get all track IDs from the list items
    const ids = [];

    const trackEl = document.querySelectorAll(".trackList li");
    trackEl.forEach(x => ids.push(x.textContent));

    // Make API request to save tracks
    const data = {
        ids: ids
    };

    const response = await fetch("https://api.spotify.com/v1/me/tracks", {
        method: "PUT",
        headers: {
            "Content-Type": "application/json"
        },
        body: JSON.stringify(data)
    });
}
```

Question 4. Local Storage and Query Parameters

3.1 (15 pts) Implement the **`get_data()`** function in `stocks.js`.

```
async function get_data(key) {
  let data = localStorage.getItem(key);
  if (data) {
    data = JSON.parse(data);
  } else {
    let response = await fetch(API_URL);
    data = await response.json();
    localStorage.setItem(key, JSON.stringify(data));
  }
  return data;
}
```

3.2 (15 pts) Implement the **`onPageLoad()`** function in `stocks.js`.

```
function onPageLoad() {
    // Get the query string from the current URL
    const params = new URLSearchParams(window.location.search);

    // Display values in corresponding HTML elements
    document.getElementById('symbol').textContent = params.get('symbol');
    document.getElementById('shares').textContent = params.get('shares');
    document.getElementById('price').textContent = params.get('price');
}
```