A screenshot of a computer code

Description automatically generated

Result:

Syntax Error

Reason:

’CS’ this is the wrong way to write or initialize a string. The correct way would be ‘CS’ or “CS”. As the interpreter doesn’t understand ’CS’ format it will not run and give a syntax error.

A screenshot of a computer code

Description automatically generated

Result:

Number of Limbs: 6

Retractable Limbs: false

Explanation:

Experiment626 is a prototype of the class Alien. A new prototype of Alien is initially created and Alien has numberOfLimbs and retractableLimbs local variables. As experiment626 is a prototype of Alien it can access the variables and thus we update the variables and new then we can access the function print defined inside Alien to console log the message displayed in the result.

A rectangular object with a hole in it

Description automatically generated

div {

    border: 1px solid black;

}

.outer {

    width: 500px;

height: 150px;

background-color: green;

/\* WRITE YOUR CODE HERE \*/

    display: flex;

    align-items: center;

    justify-content: space-between;

    padding: 10px;

    margin: 20px;

    position: relative;

}

.inner {

    height: 100%;

width: 100%;

background-color: lightgreen;

/\* WRITE YOUR CODE HERE \*/

    border: 2px solid black;

    display: flex;

    justify-content: space-between;

    align-items: center;

}

.inner .value {

    margin: 0 25px;

font-size: 48px;

font-weight: bold;

/\* WRITE YOUR CODE HERE \*/

    display: flex;

    align-items: center;

}

.left {

    /\* WRITE YOUR CODE HERE \*/

    display: flex;

    flex-direction: column;

    justify-content: space-between;

    align-items: center;

    position: absolute;

    left: 10px;

    top: 50%;

    transform: translateY(-50%);

}

.right {

    /\* WRITE YOUR CODE HERE \*/

    display: flex;

    flex-direction: column;

    justify-content: space-between;

    align-items: center;

    position: absolute;

    right: 10px;

    top: 50%;

    transform: translateY(-50%);

}

.portrait {

    height: 100%;

    width: 100px;

    border-radius: 100px;

    background-color: white;

/\* WRITE YOUR CODE HERE \*/

    display: flex;

    justify-content: center;

    align-items: center;

    margin: 0 10px;

    position: absolute;

    height: 88%;

    left: 50%;

    top: 50%;

    transform: translate(-50%, -50%);

}

A green bill with a white oval in the middle

Description automatically generated

A screenshot of a computer

Description automatically generated

import React, { useState } from 'react';

function TodoList() {

  const [tasks, setTasks] = useState([]);

  const [newTask, setNewTask] = useState('');

  const handleInputChange = (event) => {

    setNewTask(event.target.value);

  }

  const handleSubmit = (event) => {

    event.preventDefault();

    if (newTask.trim() !== '') {

      setTasks([...tasks, newTask]);

      setNewTask('');

    }

  }

  return (

    <div>

      <form onSubmit={handleSubmit}>

        <input

          type="text"

          value={newTask}

          onChange={handleInputChange}

          placeholder="Add a new task"

        />

        <button type="submit">Submit</button>

      </form>

      <List tasks={tasks} />

    </div>

  );

}

function List({ tasks }) {

  return (

    <ul>

      {tasks.map((task, index) => (

        <li key={index}>{task}</li>

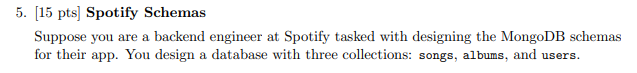
      ))}

    </ul>

  );

}

export { TodoList, List };



Part A

A screenshot of a computer program

Description automatically generated

Solution:

var db = connect("localhost:27017/myDatabase");

var album = db.albums.findOne({ name: "Thriller", artist: "Michael Jackson" });

var album\_songs = db.songs.find({ \_id: { $in: album.songs } }, { name: 1, \_id: 0 }).toArray();

Part B:



Ans: Yes I agree with my boss as with two way linking each song will be linked to the album so if a user wants to listen to the album they don’t have to look up the album and instead just any song from that album will work. Moreover, this can be helpful because if we change something about a song, it will automatically update in all the albums it's in.

Part C:

A close up of text

Description automatically generated

Ans: It sounds like a smart plan and may help with suggesting better and more curated song suggestions. It will improve the app’s algorithm of recommending songs, however, it will be very slow and not to forget way too much storage will used for no good reason. Moreover, users might complain about changing their information every time their information changes. It would cause more problems for both users and developers as the users are on the app just to listen to their favorite songs and not update their information daily.

Question 6:

A computer screen shot of a program

Description automatically generated

Ans: The button will be red, never having been yellow. (option iii)

Part B:

A screenshot of a computer program

Description automatically generated

Solution:

//  assuming the existence of the functions

//  SetButtonColor,

//  GetButtonText, and

//  SetButtonText

function OnButtonClick() {

    fetch('/endpoint1').then(response => {

        if (response) {

            color = response.color;

            SetButtonColor(color, function() {

                GetButtonText(function(text) {

                    SetButtonText(text, function() {

                    });

                });

            });

        } else {

            throw new Error('Network response was not ok.');

        }

    }).catch(error => {

        console.log(error);

    });

}

Part C:

HTML:

<!DOCTYPE html>

<html lang="en">

<head>

  <meta charset="UTF-8">

  <meta name="viewport" content="width=device-width, initial-scale=1.0">

  <link rel="stylesheet" href="./Index.css">

</head>

<body>

  <div class="divs" id="divs">

    BOXES FOR RENDERING

  </div>

  <script src="./script.js"></script>

</body>

</html>

JS:

const Border = (col, res) => {

    const bord = (res) => {

      if (res) {

        return 'solid';

      }

      return 'dashed';

    };

    document.getElementById('divs').style.border = `3px ${bord} ${col}`;

  };

const GetEnd = (endP, col) => {

  return new Promise((resolve, reject) => {

    fetch(endP)

      .then(response => {

        if (response.ok) {

          resolve(col);

        } else {

          reject(col);

        }

      }).catch(error => reject(col));});

};

Promise.allSettled([

  GetEnd('/endpoint1', 'yellow'),GetEnd('/endpoint2', 'blue'),GetEnd('/endpoint3', 'red')

])

.then(results => {

  results.forEach(result => {

    if (result.status && result.status === 'fulfilled') {

      Border(result.value, true);

    } else {

      Border(result.reason, false);

    }

  });

});

CSS:

body {

    font-family: Arial, sans-serif;

    display: flex;

    justify-content: center;

    align-items: center;

    height: 100vh;

    margin: 0;

  }

  .container {

    padding: 20px;

    background-color: #f7f7f7;

  }