Prepared By - Deepak Uniyal, Assistant Professor, CSE

Full Stack Web Development - Holi Assignment

Note - This assignment is for 6th semester section B and section C only. Please follow the below mentioned instructions for assignment completion. The instructions to submit the assignment will be provided soon.

1. All questions are mandatory.
2. Use Git and GitHub to complete the assignment.
3. Perform small commits as directed in lectures.
4. Don't push the whole code of a large problem using a single push command, that will be considered a use unfair means.
5. Copying code from Internet or others will be dealt seriously.

Problem Statement -1 **6 Marks**

Create a web application that takes student's enrolment number, name, age, course, branch, contact, and email from web portal using a PHP script and shows that on a web page. Extend the application to store the data entered through the web portal portal into database (CSE2020). Once you have saved data to the database, read the data through a PHP script and show it on the web portal. Further, extend the application to show the data on web page in form of a neat and clean table with dashed border for every row and column as shown below. Every row should have a edit and delete button to update or remove a record from the database. When clicking on edit button, it should open a new web page with all the data fields

Enrollment	Name	Age	Course	Branch	Contact	Email	Edit	Delete
123	ABC	21	B.Tech.	CSE	9897	cse@geu.ac.in	Edit	Delete

already filled for corresponding ID. After you click delete button, a confirmation should be taken to remove or cancel the record.

Problem Statement -2 **1 Mark**

Visit - https://jsonplaceholder.typicode.com/todos and parse JSON data to find out the ID of all users those have completed minimum and maximum number of tasks.

Problem Statement -3 **2 Marks**

Visit - https://jsonplaceholder.typicode.com/posts and do the following with JSON data -

a. After running program user should be able to enter an input keyword of his choice. In case user doesn't enter any keyword, program should exit safely.

b. Once keyword is entered, you should find out the count of input keyword in **title** and **body** of corresponding users. **For Example** - When user enters "**molestiae**" output for user with id 1 should be -

```
1 1 1
1 2 1
1 3 1
1 6 2
```

Explanation - First and second columns show the ID of user and post respectively, where third column shows the number of times a keyword appeared in that particular user and post ID.
You should generate the output for all the users that are in JSON file.

Problem Statement -4 **3 Marks**

This problem is taken from - Recruitment Test Part II for post of Software Engineer at Smarter Codes -

Steps to perform in this problem :

1. Parse the JSON file
2. Go inside 'Body > Recommendations'
3. Print name of Restaurant
4. Go inside 'Body > Recommendations'
5. Go inside menu
6. Loop for each menu (there can be multiple menus)
7. Check whether type=sectionheader. If it is - go inside its 'children
8. Check whether type=item. If it is, check whether selected=1. If both are true, print the name of item, and go inside its children.
9. Check whether selected=1 (no need to pay attention to the 'type' now that we are inside children of 'item'). If selected was 1 then print the name of child, and go further deep inside the children

Repeat the last step infinitely - until we have reached the deepest child
The output should look like this :

```
-------------------------------------------
Restaurant1
--> Item1
-----> Child1
----------> Child1.1
----------> Child1.2
------> Child2
------> Child3
----------> Child3.1
```

Take note of length of arrows in each line above. Your output should also contain these arrows to depict the indentation of each child.

Note -

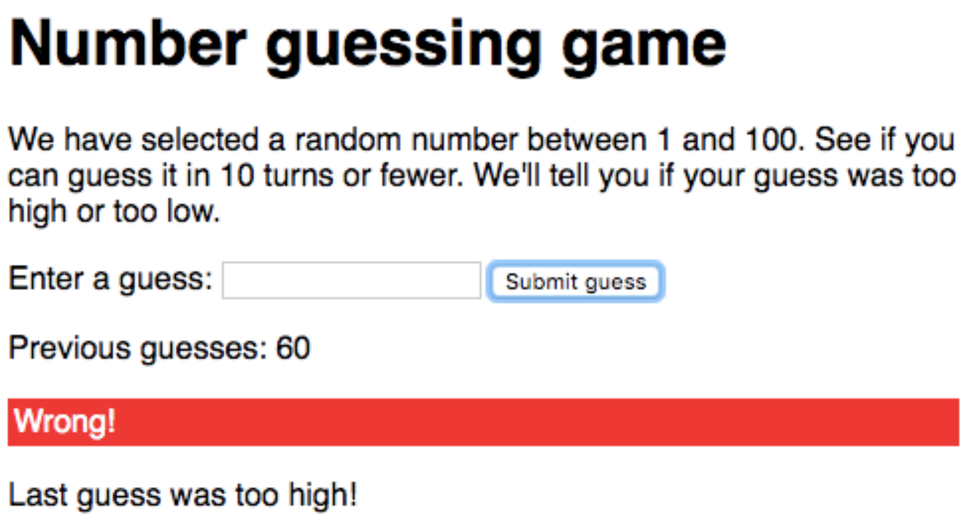
A. Attached is the JSON file, named parse.json which you have to parse.
B. This video will help you complete the assignment - Parse JSON File

Problem Statement -5 — 3 Marks

Guess The Number Game using JavaScript

See Instructions for Guess The Number Game (use **guess.html** file as a framework for the game and see the screenshots given below for reference.**)**

1. Generate a random number between 1 and 100.
2. Keep a record of guessCount. Start it on 1.
3. Provide the player with a way to guess what the number is.
4. Once a user submits guess record it somewhere so the user can see their previous guesses.
5. Next, check whether it is the correct number.
6. If it is correct:
    1. Show congratulations message.
    2. **Stop the player from being able to enter more guesses.**
    3. After that display control allowing the player to restart the game.
7. If it is wrong and the player has some guess counts left:
    1. Display the wrong message.
    2. Allow them to enter another guess.
    3. Increment the guessCount by 1.
8. If it is wrong and the player has no guess counts left:
    1. Tell the player that game is over.
    2. Stop the player from being able to enter more guesses.
    3. After that display control allowing the player to restart the game.
9. Once the game restarts, make sure the game logic and UI are completely reset, then go back to step 1.



Number guessing game

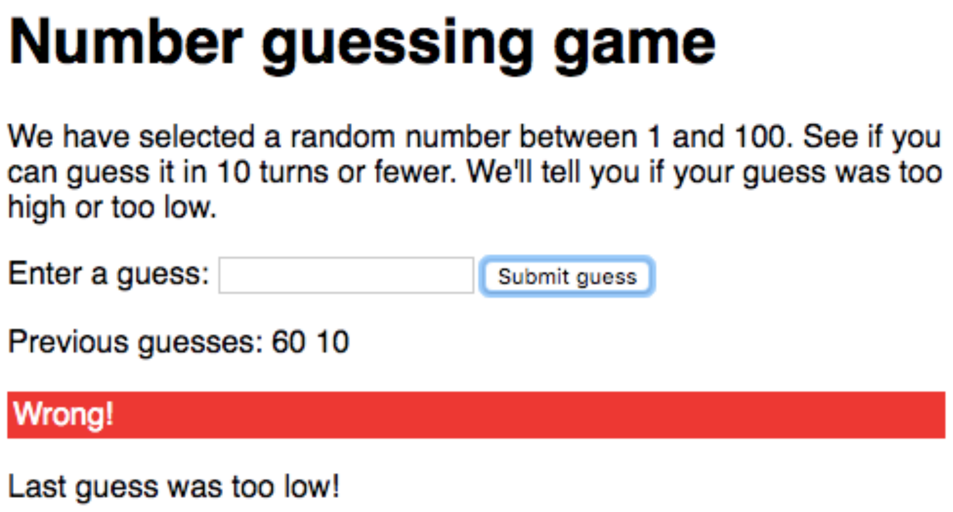
We have selected a random number between 1 and 100. See if you can guess it in 10 turns or fewer. We'll tell you if your guess was too high or too low.

Enter a guess:

Previous guesses: 60

Wrong!

Last guess was too high!



Number guessing game

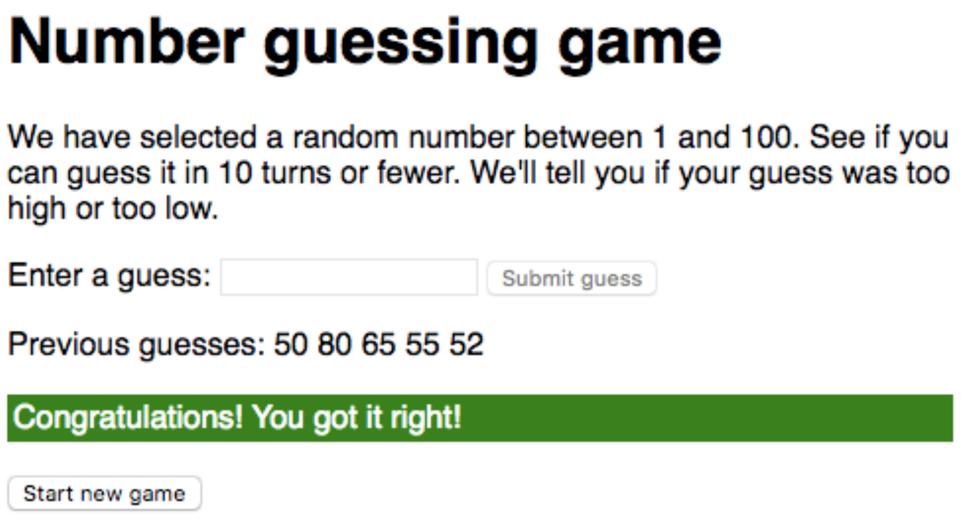
We have selected a random number between 1 and 100. See if you can guess it in 10 turns or fewer. We'll tell you if your guess was too high or too low.

Enter a guess:

Previous guesses: 60 10

Wrong!

Last guess was too low!



Number guessing game

We have selected a random number between 1 and 100. See if you can guess it in 10 turns or fewer. We'll tell you if your guess was too high or too low.

Enter a guess:

Previous guesses: 50 80 65 55 52

Congratulations! You got it right!

Please Take Care of You and Your Family

Avoid Travel and Crowded Place as much as Possible

Wash your Hands Frequently