IS216 Term 2022/23 T1
Sample Mini Lab Test Set 1 [48 marks]

This lab test was an actual lab test used in 2021/22 T2 & is now released as a practice paper.

General Instructions:

- Time given: 2 hours. Format: open-book, open-Internet, and to be completed individually.
- Your graders will be using only **Google Chrome Web Browser** to test your web pages.
- No questions will be entertained by the IS216 teaching team (faculty/instructor/Teaching Assistants) during the test period. If necessary, make your own assumptions and complete test questions.
- You must use only standard HTML5, CSS, Bootstrap (Version 5.2), JavaScript, and Axios in your solutions unless the question specifies otherwise. Do NOT use any other third-party libraries (e.g. Angular, React, or others).
- Use meaningful names for HTML class/id and JavaScript variables/functions. You must indent your code (HTML/CSS/JavaScript) properly. Failure to do so will attract a penalty of up to **20%** of your score for the corresponding question.
- You **MUST** include your name and SMU email ID in the comments of all your submitted source files. Failure to do so will attract a penalty of up to **20%** of your score for the corresponding question.
  For example, if your registered name is “LEE Oppa” and email ID is “lee.oppa.2020”, include the following comment at the beginning of each source file you write.

HTML files	**CSS, JavaScript files**

`<!--`
`    Name:  LEE Oppa`
`    Email: lee.oppa.2020`
`-->`
	`/*`
	`    Name:  LEE Oppa`
	`    Email: lee.oppa.2020`
	`*/`

- You may wish to comment out the parts in your code which cause errors. Commented code will not be marked.

Submission Instructions:

- Zip up all files in Q1/Q2 folders into <YOUR_SMU_ID>.zip

- For example, lee.oppa.2020.zip
- Verify by unzipping this zip file – check the content inside. Incorrect submission file name WILL attract a penalty of up to 20% of your score for the entire test.
- Only **zip** format is accepted. **.7z**, **rar** or other compression formats are NOT accepted.
- Until the correct **zip** format is submitted again by the student, it will be assumed that the student has NOT made the submission and late submission policy will apply.

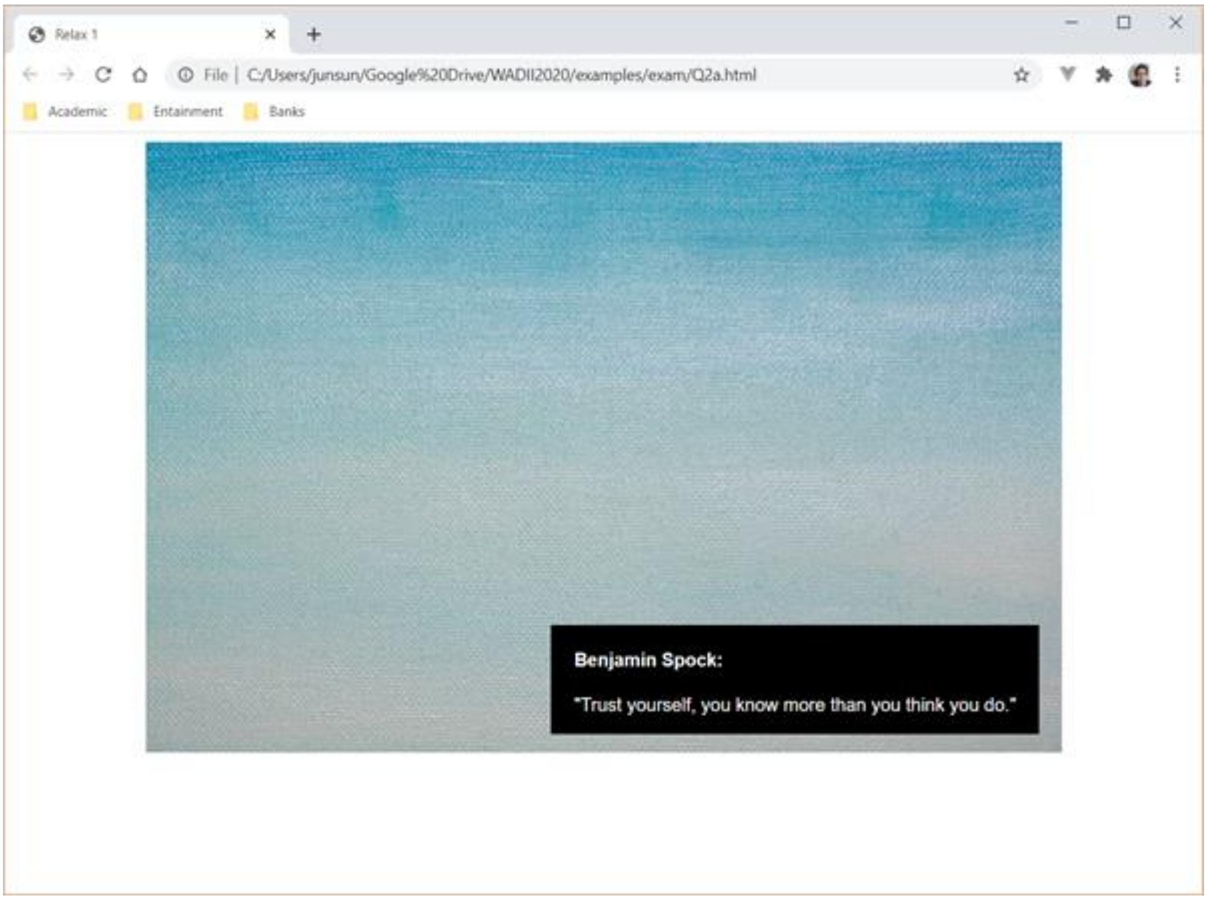
There are 5 questions for this test:

1. [CSS, bootstrap] Inspirational Quote (10 marks)
2. [CSS, bootstrap] Correcting Errors (10 marks)
3. [JS, CSS] Noughts & Crosses (10 marks)
4. [JS, DOM] Menu (10 marks)
5. [Async request, JS] Weather (8 marks)

Q1: CSS/Bootstrap: Inspirational Quote (10 Points)

Assume that you would like to build a Web page showing an inspirational quote displayed over a nice image.

PART A: In this first design, you would like to apply your CSS knowledge to have the page as shown below.


Relax 1
File | C:/Users/junsun/Google%20Drive/WADII2020/examples/exam/Q2a.html
Academic
Entainment
Banks

Benjamin Spock:

"Trust yourself, you know more than you think you do."

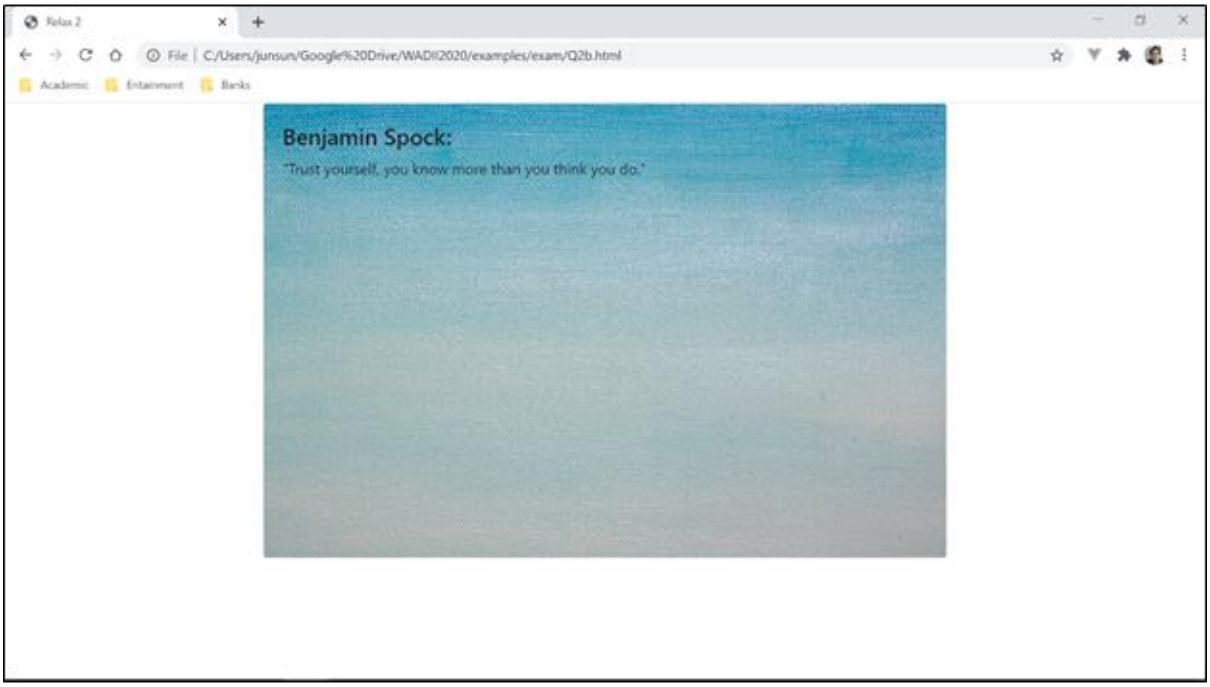
Complete Q1a.html so that the following requirements are satisfied (Do NOT change the HTML code)

The following are the specific requirements.

- The image should be displayed with a width of 800px;
- The image should be shown in the center of the window.
- The block of text is shown at the bottom-right corner of the image.
- The text is in white over a black background.
- The right edge of the text block is 20px from the right edge of the image and the bottom edge of the text block is 20px from the bottom edge of the image.

- There are paddings of 20px on the left and right of the text block.

PART B: In this second design, you would like to apply Bootstrap to make the page more responsive using the help of Bootstrap classes such as “row”, “col-*” and “Card” (Bootstrap component) . The design is shown below.


Relax 2
File | C:/Users/junsun/Google%20Drive/WADII2020/examples/exam/Q2b.html
Academic
Entainment
Banks

Benjamin Spock:

"Trust yourself, you know more than you think you do."

The image and text will be presented in the form of a *card* in bootstrap.

Furthermore, the card should be shown in the center, occupying 5/6 of the width of the browser window. The remaining width will be distributed equally on the left and right. Note that the size of the image should change accordingly when the browser is resized.

Complete Q1b.html to achieve the effect using only **Bootstrap** classes.

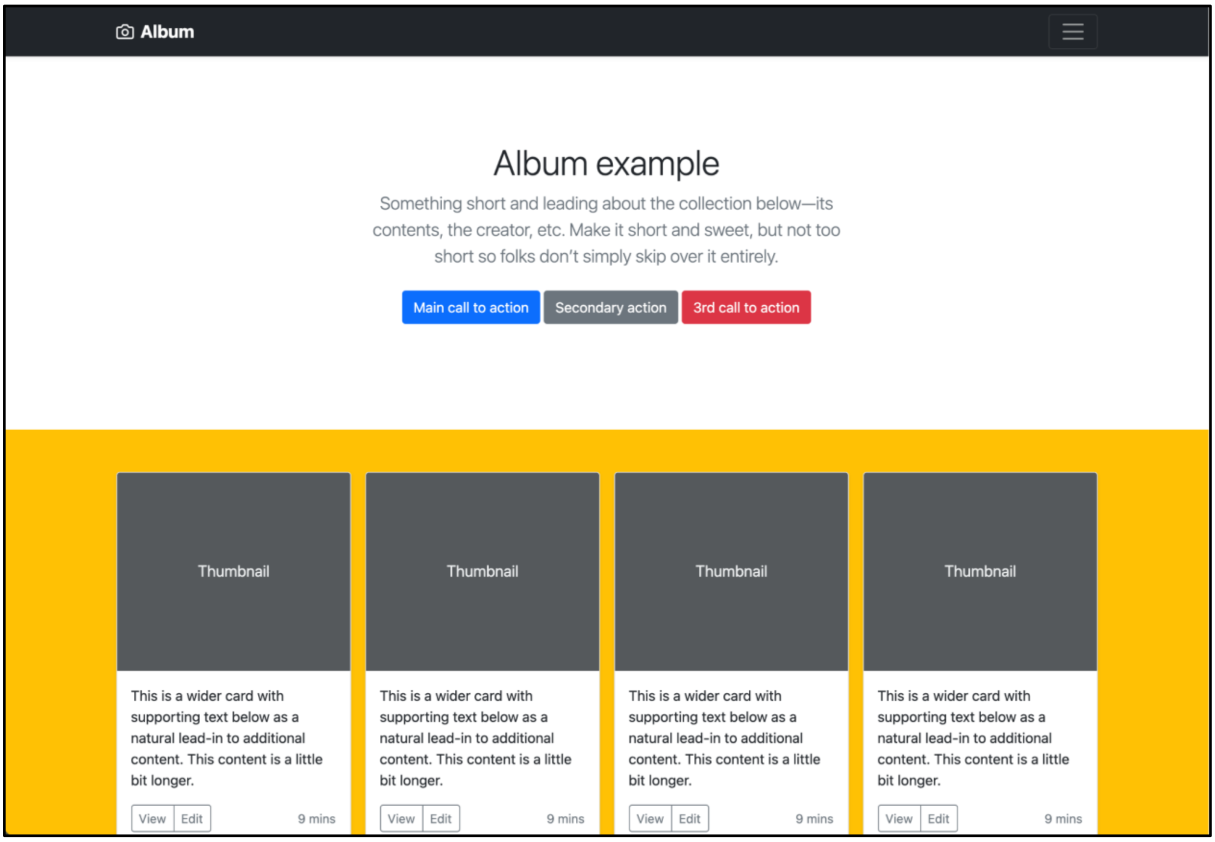
Q2: Bootstrap/CSS: Correcting Errors (10 Points)

PART A: Q2a.html is an HTML file adapted from Bootstrap Album.

Edit Q2a.html as little as possible in order to achieve the following objectives. All changes are to be made in the same HTML file.

Here is what you need to achieve (see screenshot below):

1. Insert a red color third button labelled “3rd call to action” beside the two original buttons.
2. Edit the background color of the thumbnail section to orange.
3. Currently, 1, 2 or 3 thumbnails appear in each row depending on the browser’s width. Edit the HTML file such that when the browser width is wide enough, the user should see 4 cards in each row.



Album example

Something short and leading about the collection below—its contents, the creator, etc. Make it short and sweet, but not too short so folks don’t simply skip over it entirely.

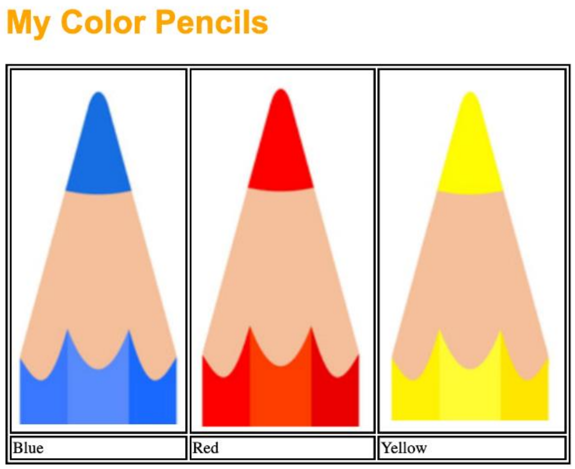
This is a wider card with supporting text below as a natural lead-in to additional content. This content is a little bit longer.

This is a wider card with supporting text below as a natural lead-in to additional content. This content is a little bit longer.

This is a wider card with supporting text below as a natural lead-in to additional content. This content is a little bit longer.

This is a wider card with supporting text below as a natural lead-in to additional content. This content is a little bit longer.

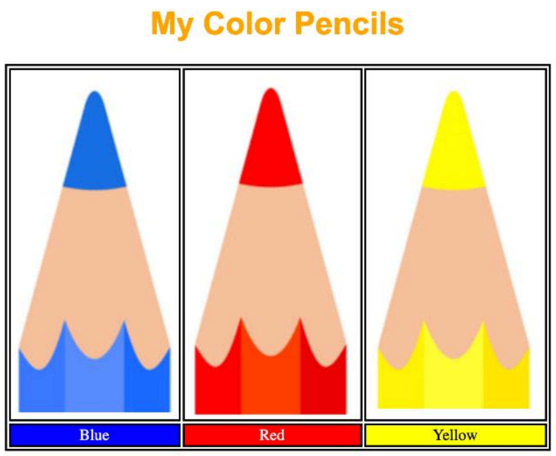
PART B: Open Q2b.html in a browser. This is how it looks:



My Color Pencils

Blue	Red	Yellow

Edit only the `<STYLE>` element in the HTML file so that it looks like this:



My Color Pencils

Blue	Red	Yellow

The heading "My Color Pencils" and table are to be centered in the browser window. Pay attention to the text colors as well as the background color of the second row in the table. “My Color Pencils” is supposed to be in orange. Do not link to any external file and only the `STYLE` element can be edited.

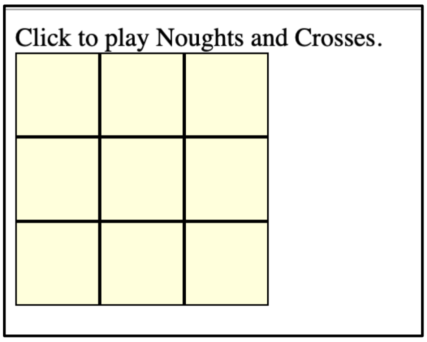
Q3: JS/CSS: Noughts & Crosses (10 Points)

Q3.html is a fully functional HTML file from getcodingkids.com.

Do not break the current functionality of the game and edit Q3.html as little as possible in order to achieve the following objectives. All changes are to be made in the same HTML file, and the HTML file should not link to any external resources.

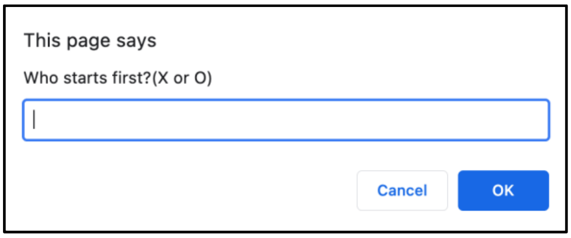
Here is what you need to achieve:

1. The whole grid of 9 cells has a background color of "lightyellow" when the page is loaded.



Click to play Noughts and Crosses.

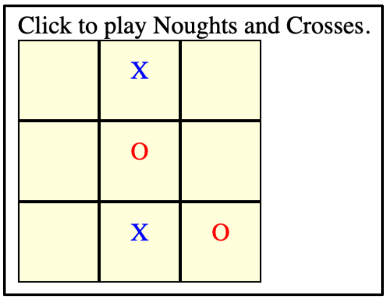
2. The Xs and Os should be horizontally aligned center when they appear within each cell. They don't have to be vertically aligned though.
3. For now, the game always starts with O as the first player. Change it so that when the page loads, it prompts for "Who starts first? (X or O)" and use the response to determine the first player. If the user enters an invalid response to the prompt, there is no need to show an error message, and the game should just default to O as the starting player.



This page says

Who starts first?(X or O)

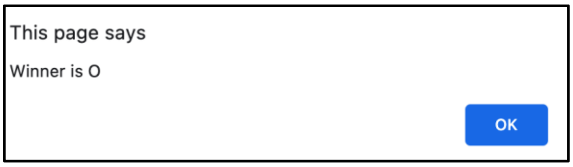
4. When they appear in the grid, all Xs should be blue and Os should be red. If you are unable to do this, make both X and O red in color.



Click to play Noughts and Crosses.

	X	
	O	
	X	O

5. When someone wins, the alert message should show "Winner is X" or "Winner is O", instead of just "Winner!".

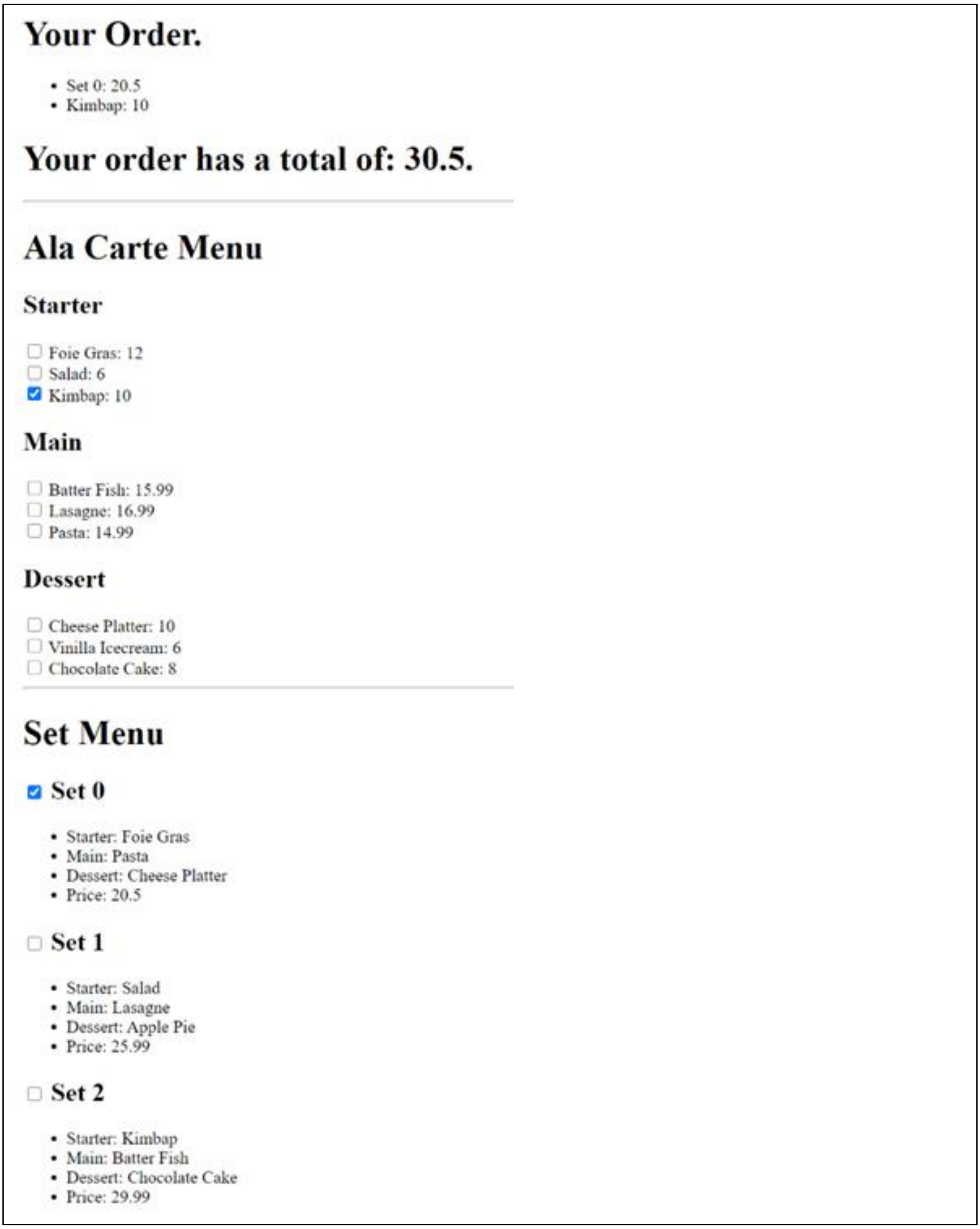


This page says

Winner is O

Q4: JS/DOM: Menu (10 Points)

You are to build a menu ordering page as shown in the following two figures. It shows an Ala Carte menu as well as a set menu. At the top of the page, when nothing is selected, it shows "`You haven't ordered anything`"; otherwise, it shows the ordered items together with the price of each ordered item as well as the total price, as shown in the Figures.



Your Order.

- Set 0: 20.5
- Kimbap: 10

Your order has a total of: 30.5.

Ala Carte Menu

Starter

□ Foie Gras: 12
□ Salad: 6
☑ Kimbap: 10

Main

□ Batter Fish: 15.99
□ Lasagne: 16.99
□ Pasta: 14.99

Dessert

□ Cheese Platter: 10
□ Vinilla Icecream: 6
□ Chocolate Cake: 8

Set Menu

☑ Set 0

- Starter: Foie Gras
- Main: Pasta
- Dessert: Cheese Platter
- Price: 20.5

□ Set 1

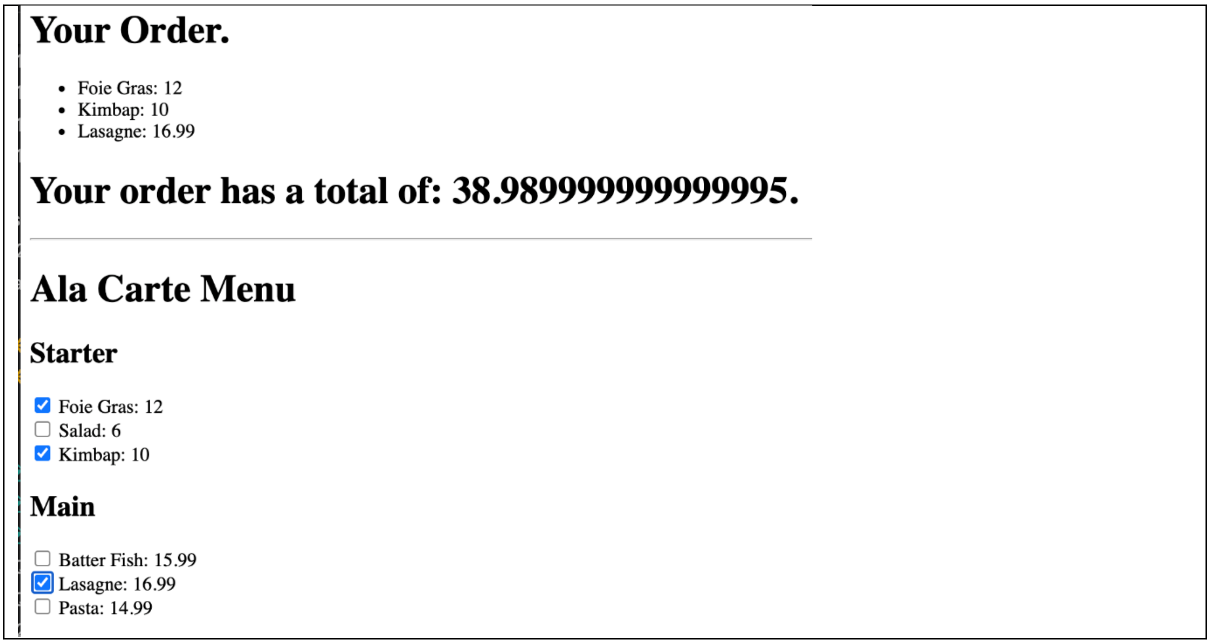
- Starter: Salad
- Main: Lasagne
- Dessert: Apple Pie
- Price: 25.99

□ Set 2

- Starter: Kimbap
- Main: Batter Fish
- Dessert: Chocolate Cake
- Price: 29.99

`Q4.html` shows the skeleton of the page. Complete the page at the position indicated by the **“TODO”** comments. Do not change any other given code.

Note that the prices of the items are a part of the page. You are to read the price from the page. Furthermore, the function `updt(order)` is invoked whenever the user selects or un-selects an item. Define the function so that the selected items are shown at the top of the page.



Your Order.

- Foie Gras: 12
- Kimbap: 10
- Lasagne: 16.99

Your order has a total of: 38.989999999999995.

Ala Carte Menu

Starter

☑ Foie Gras: 12
□ Salad: 6
☑ Kimbap: 10

Main

□ Batter Fish: 15.99
☑ Lasagne: 16.99
□ Pasta: 14.99

Note that to update the message at the top of the page, you are allowed to set `.innerHTML` for this exercise.

Q5: Async Request/JS: Weather (8 Points)

We would like to build a Web page which changes its appearance according to the time the user loads the page, i.e. the page will have a black background if it is loaded half an hour before sunrise or half an hour after sunset; and will have a particular background during dawn (half an hour before and after sunrise, both inclusive) and twilight (half an hour before and after sunset, both inclusive); and will have a white background otherwise. Four CSS classes have been defined in Q5.html. Your task is to associate the right CSS class with the tag `body` depending on when the page is loaded.

To know the exact sunrise and sunset time, we would like to use the Web service provided by `openweathermap.org`. The following shows a sample JSON response from the Web service

```
{
  "coord": {
      "lon": 103.85,
      "lat": 1.29
  },
  "weather": [
      {
      "id": 803,
      "main": "Clouds",
       "description": "broken clouds",
      "icon": "04d"
      }
  ],
  "base": "stations",
  "main": {
      "temp": 303.76,
    "feels_like": 307,
    "temp_min": 303.15,
    "temp_max": 304.15,
    "pressure": 1009,
    "humidity": 70
  },
  "visibility": 10000,
  "wind": {
      "speed": 4.1,
      "deg": 180
  },
  "clouds": {
      "all": 75
  },
  "dt": 1600402268,
  "sys": {
      "type": 1,
     "id": 9470,
    "country": "SG",
    "sunrise": 1600383320,
     "sunset": 1600426938
  },
  "timezone": 28800,
  "id": 1880252,
  "name": "Singapore",
  "cod": 200
}
```

The data contains the sunrise and sunset time, in the UNIX time format, i.e., the number of **seconds** since the UNIX Epoch (00:00:00 UTC on 1 January 1970). Write a function `handleData` which takes the JSON data, reads the sunrise and sunset time and updates the appearance of the page accordingly using the given CSS classes.

Hint: You can use `new Date().getTime()` to get the number of milliseconds since the Unix Epoch.