WAD Trial Lab Test 1 (1.5 hours) - Set C [25 marks]

General Instructions:

- You can refer to any offline resources already on your laptop, but you must disable all networking and Bluetooth connections during the test. You must not communicate with anyone via any means during the test.

- Just before the test, you will be given instructions by the invigilator as to how to obtain resource files required for the lab test and how to submit your solutions.

- No questions will be entertained during the test. If necessary, make your own assumptions.

- You are allowed to use only standard PHP classes and functions in your solutions – do not use any third party libraries.

- Use meaningful names for classes, methods, functions and variables, as well as indent your code correctly. Use 4 spaces for indentation. Otherwise, you may attract penalty of up to **20%** of your score for the corresponding question.

- You **MUST** include your name as author 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 "PARK Bogum" and email ID is park.bogum.2020, include the following comment at the beginning of each source file you write.

```
<!--
        Name:  PARK Bogum
        Email: park.bogum.2020
-->
```

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

- **Resources:** Click here
- **Solutions:** Click here

DO NOT TURN OVER UNTIL YOU ARE TOLD TO DO SO

Question 1 (Difficulty Level: */)** **[15 marks]**

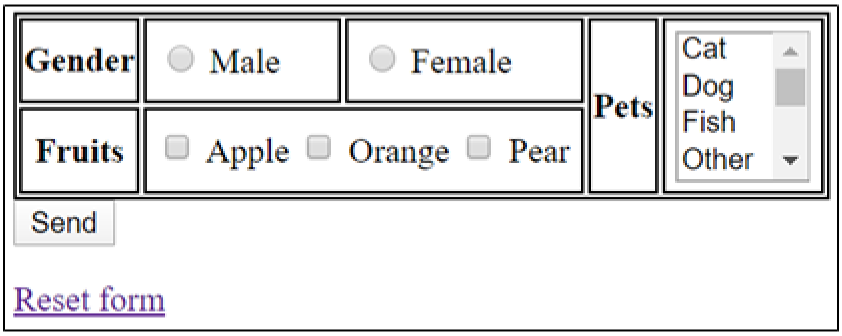
Given:

- `q1`
  - `one.php`
  - `fruits`
    - `apple.jpg`
    - `orange.jpg`
    - `pear.jpg`
    - `none.jpg`

<u>Part A (5 marks)</u> - Difficulty Level (*)

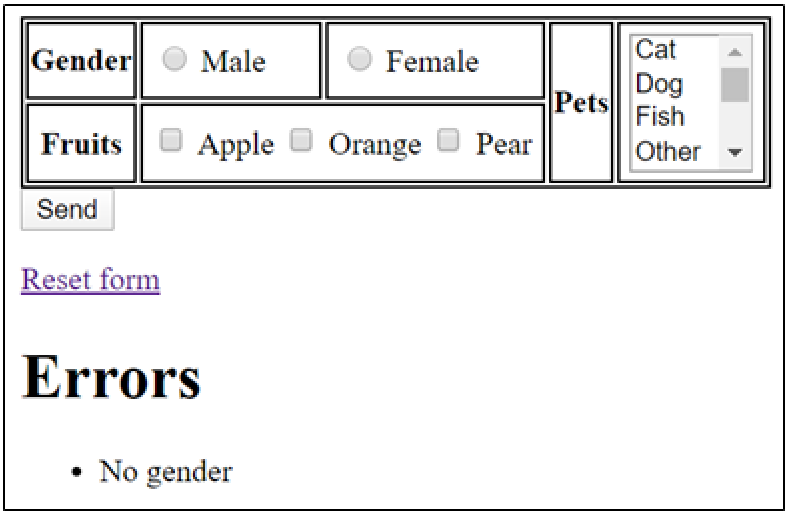
Update `one.php` so that it will display the following form.

1. The field labels (e.g. Gender, Fruits, Pets) are in table header cells.
2. For Gender and Fruits, a field is selected when the user clicks on its corresponding text.
3. You MUST use the given arrays `$gender_list`, `$pet_list` and `$fruit_list` to generate the respective form fields.
4. The form is submitted back to itself. The submitted form values must NOT be visible in the web browser's address bar.
5. Clicking the 'Reset form' link brings the user back to `one.php`.



<u>Part B (10 marks)</u> - Difficulty Level (**)

1. Upon receiving the form's values, one.php checks that the user has selected a gender.
2. If there is any error, display error messages (see image below) in an unordered list



3. If there is no error, `one.php` does the following:
   a. Repopulates the form fields with the submitted value.
   b. If gender is male, display salutation `'Dear Sir'`. Otherwise, display `'Dear Miss'`.
   c. Display the list of pets in an ordered list. If there is no pet selected, display '`No pets`'.
   d. For the submitted list of fruits, display its corresponding images in folder 'fruits'. If there is no fruit selected, display the image `'none.jpg'`.

Example 1:

Example 2:

Example 3:

Question 2 (Difficulty Level: *//**)** **[10 marks]**

Given:

- q2
  - functions.php
  - getMonthName.php
  - index.php
  - luckStringToDict.php

<u>Part A (1 mark)</u> - Difficulty Level (*)

Edit functions.php to complete function getMonthName($month_num)

1. Take in 1 parameter $month_num which is an integer that represents the required month. You can assume that its value is an integer between 1 to 12 inclusive.
2. Return string 3-characters name of the month.
   1. E.g. $month_num 1 is 'JAN', and so on
   2. You are to make use of the given indexed array $names to get the name of the required month.

If done correctly, open getMonthName.php in browser and the expected output is as follows:

```
getMonthName(1)

C:\wamp64\www\idea\solution\q2\getMonthName.php:7:string 'JAN' (length=3)

getMonthName(5)

C:\wamp64\www\idea\solution\q2\getMonthName.php:11:string 'MAY' (length=3)
```

<u>Part B (4 marks)</u> - Difficulty Level (**)

Edit functions.php to complete function luckStringToDict($luck_str)

1. Take in 1 parameter $luck_str which is a string depicting the monthly luck.
   a. The string has the format
      '<luck>[optional:x<number of months>], ..., <luck>'

      i. <luck> can be either GOOD, NORMAL or BAD
      ii. If it is followed by x<number of months>, this <luck> will persist for the specified number of months. Otherwise, this <luck> is for one month.
      iii. The string always ends with <luck> which means this is luck until the end of year.
   b. **Example 1:** 'GOODx2,BAD,GOODx3,NORMAL' means
      i. GOOD for 2 months (Jan, Feb),
      ii. BAD for 1 month (Mar),
      iii. GOOD for 3 months (Apr, May, Jun) and
      iv. NORMAL till the end of year (Jul to Dec).
   c. **Example 2:** 'NORMALx5,BAD' means
      i. NORMAL for 5 months (Jan to May),
      ii. BAD till the end of year (Jun to Dec).
   d. **Example 3:** 'NORMAL' means NORMAL till the end of year (Jan to Dec),
2. Returns an associative array whose

a. Key is either 'GOOD', 'BAD', or 'NORMAL'
b. Value is indexed array of 3-characters month names; e.g. ['JAN', 'APR']

If done correctly, open `luckStringToDict.php` in browser and the expected output is as follows:

```
luckStringToDict( 'NORMALx2, GOODx2, BADx2, NORMALx3, BAD' )
C:\wamp64\www\idea\solution\q2\luckStringToDict.php:21:
array (size=3)
  'GOOD' =>
    array (size=2)
      0 => string 'MAR' (length=3)
      1 => string 'APR' (length=3)
  'BAD' =>
    array (size=5)
      0 => string 'MAY' (length=3)
      1 => string 'JUN' (length=3)
      2 => string 'OCT' (length=3)
      3 => string 'NOV' (length=3)
      4 => string 'DEC' (length=3)
  'NORMAL' =>
    array (size=5)
      0 => string 'JAN' (length=3)
      1 => string 'FEB' (length=3)
      2 => string 'JUL' (length=3)
      3 => string 'AUG' (length=3)
      4 => string 'SEP' (length=3)
```

```
luckStringToDict( 'NORMALx2, GOODx2, BADx2, NORMALx3, BAD' )
C:\wamp64\www\idea\solution\q2\luckStringToDict.php:26:
array (size=3)
  'GOOD' =>
    array (size=10)
      0 => string 'MAR' (length=3)
      1 => string 'APR' (length=3)
      2 => string 'MAY' (length=3)
      3 => string 'JUN' (length=3)
      4 => string 'JUL' (length=3)
      5 => string 'AUG' (length=3)
      6 => string 'SEP' (length=3)
      7 => string 'OCT' (length=3)
      8 => string 'NOV' (length=3)
      9 => string 'DEC' (length=3)
  'BAD' =>
    array (size=0)
      empty
  'NORMAL' =>
    array (size=2)
      0 => string 'JAN' (length=3)
      1 => string 'FEB' (length=3)
```

Part C (5 marks) - Difficulty Level (**)

Add code to `index.php` such that the page works as described below.

1. When the page is loaded for the first time, it has a form with a text field and the form submits back to itself using HTTP GET.

   Horoscope [ ] Submit

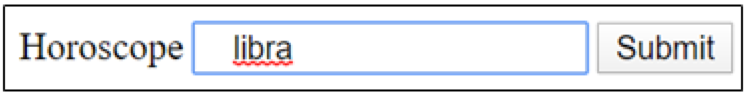
2. Upon receiving the form submission, the page checks if the submitted value is a valid horoscope by doing the following
   a. Remove any preceding and trailing white spaces from the submitted horoscope value
   b. Check if the submitted horoscope value is a key in the given associative array `$horoscopeDict`. Your code should do case-insensitive check against the array's keys.

User's input	Valid horoscope?
`‘          libra    ’`	Yes
`‘COW’`	No
`‘’`	No

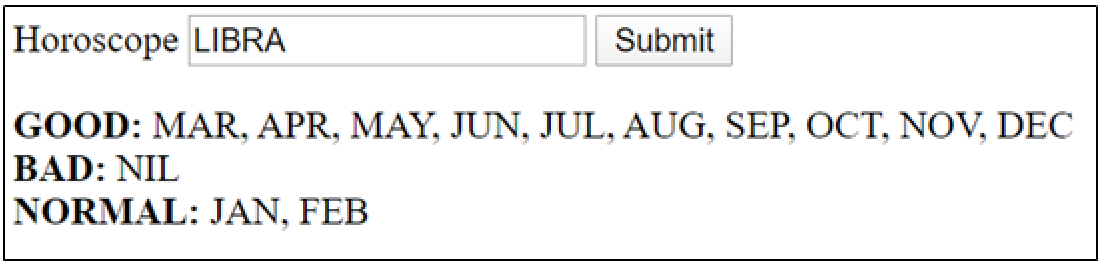
   WARNING: Do NOT CHANGE the given associative array `$horoscopeDict`!

3. If the submitted value is a valid horoscope,
   a. Repopulate the text field with the user's input in UPPERCASE with the preceding and trailing white spaces removed.
   b. Display the monthly luck for the specified horoscope as shown below. **You MUST use the function `luckStringToDict()` for this.**
   c. If there are no months for a particular luck, display NIL.

Example 1: User enters `'      libra     '`.



The expected output. Note there is no month with bad luck.



4. If the submitted value is NOT a valid horoscope,
   a. Repopulate the text field with user's input in **UPPERCASE** and
   b. displays 'Unknown horoscope' as an unordered list.

Example 2: User enters `'cow'`.

The expected output.