Pass Task 2

This document contains a number of tasks for you to attempt.

Screen Capture Process

Please take screen captures of your HTML pages, JavaScript code and paste into a Word document.

Windows comes with an application called Snipping Tool which will perform this task.

Mac OSX has some built in keyboard shortcuts (Command-Shift-4). The image is saved on the desktop. https://www.itg.ias.edu/content/keyboard-shortcuts-capture-screen-shot-mac-os-x.

There are also products such as Jing (Win and OSX).

Try to avoid taking screen shots of the entire screen if you are only interested in part of that screen.

Submission Process

Name your Word document as 2P_surname_studentID.

Paste the required screen captures from the tasks below into the appropriate places within these files.

Convert the Word document into PDF.

Upload the PDF file into Pass Task 1 submission link on Canvas.

In some cases you will be asked to submit your .html and .js

Task 1. – Wordpress Task (based on Week 4 materials)

Work through the ICT10013_WordPress_intro.docx document (available from Week 4 subject section).

Ensure that your name and ID appear on the Contact page and the dog page. Modify these pages if necessary.

Add a Rabbit page to your site.
Include 2 different widgets on this page that take your fancy

Add any other pet page to your site. **Cats are not** allowed.
Include 2 different widgets on this page that take your fancy

Ensure that your menu now includes the Rabbit and another Pet page links.

Paste your screen captures of your Dog page and About page into 2P_surname_studentID document.

Task 2. Javascript task (based on week 4 materials)

Create a HTML file named **w4c.html** based on template.html.

- Change the title to Grade from mark, description "Determine subject grade based on the final mark" and update other meta tags as applicable.
- Add a script element which specified a file named w4c.js within the **src** attribute.
- Inside the <form> section:
  use <label> and <input /> to create a webpage allowing the user to enter student ID and student mark; use <button> to create a button
  with id="send".
- After the closing form tag, the output paragraphs should state :
  Student ID _____
  Subject mark _____
  Subject grade _____

The rule is specified in the table below (assume that a mark is an integer, no decimal part allowed):

Mark	**Grade**
Below 50	F (Fail)
From 50 and under 60	P (Pass)
From 60 and under 70	C (Credit)
From 70 and under 80	D (Distinction)
Between 80 and 100 inclusive	HD

Create a **JavaScript** file named **w4c.js** based on **template.js.**

First you need to create a function **markToGrade()** which accepts one parameter mark and returns corresponding grade, except when the mark is not between 0 and 100 returned value is X. Create a local variable grade.

You will need a set of nested if statements. Although there is more than one way of writing nested if statements here, let's start with validating mark. First check whether the mark is below 0 or above 100 and assign X to grade. Else means that the mark is valid (i.e. it is between 0 and 100 inclusive). So you can start checking if the mark is below 50 and assign the grade accordingly.

```
if (… - condition for invalid mark)
        grade = "X";
else //mark is between 0 and 100
        if (mark < 50)
                grade = "F";
        else //means mark >=50 so need to check the mark is <60
        if (mark < …
```

Note: tutors will not accept badly structured nested if statements, where the CPU has to test conditions.

Within init():

- Create a local variable btnSend and store the reference to the HTML button with id "send".
- Assign call to the function processMark to the onclick event of the btnSend variable.

Create a function processMark(). This function does not need any parameters.

Within processMark():

- Create a local variable **studentID** and assign the value entered into the first textbox to this variable.
- Create a local variable **subjMark** and assign the value entered into the second textbox to this variable.
- Since subjMark is a string, convert it to a number.
- Create a local variable **subjGrade**.
- Call the function **markToGrade()** and assign the returned value to subjGrade.
- Update the first 2 lines of the output on the webpage. Student ID <whatever id the user entered>
  Subject mark <whatever value the user entered>

- Use if statement to check whether subjGrade is X. In this case the output on the 3d line should show the error message:
  Subject grade – replace this paragraph with the error message that entered mark is invalid.

  Otherwise the 3d line of the output should be
  Subject grade <whatever is the corresponding grade>

Test w3b.html. Note you have to test every if branch, i.e. marks resulting in every possible grade, including borderline values, such as 0, 50, 60, 70, 80 and 100.

Upload your HTML and Javescript files on Canvas.

Task 3. Javascript task (based on Week 5 materials)

Create a HTML file named **w5P.html.**
Make sure that **your student name and ID** are in the meta tags.

Create a JavaScript file named **w5P.js**. Make sure the script element specifies **w5P.js** within the **src** attribute.

Create a function isNumberMobile(), which takes the phone number as a parameter, extracts the first two characters and checks that they are "04". If yes, return true, otherwise return false.

Within acceptInput()

- Add one more variable **isMobile**
- Inside if(isPhoneValid), i.e. if we have the valid phone number call isNumberMobile() and store the result in the variable **isMobile**

- Use if(isMobile) to add text “Phone number is mobile” to the text “Phone number valid” and output it in the paragraph with the id “msg”, otherwise concatenate the text “Phone number is probably landline” with the text “Phone number valid” and output it in the paragraph with the id “msg”. If the phone number is invalid your program should not check if it’s mobile but should simply display “Phone number invalid”

Test w5P.html. Make sure you test:

- all digits but less than 10 characters
- exactly 10 characters but not only digits
- less than 10 characters and not only digits
- all digits, exactly 10 characters and the first 2 are “04”
- all digits, exactly 10 characters and the first 2 are not “04”

Screen Capture the HTML page displayed in your browser and paste into 2P_surname_studentID document.

Upload your HTML and Javescript files on Canvas.

Task 4. HTML task (based on week 5 materials)

- Create a HTML file named **w5b.html** based on template.htm
- Change the title to Testing flexbox, description “Containers, flexboxes” and update other meta tags as applicable.
- Add the following to the .html file <head> section:
  <link rel="stylesheet" href="**w5b.css**">

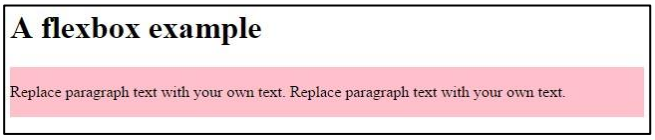
- Add the following to the .html file <header> section:
  <h1> A flexbox example</h1>

- Add the following to the .html file <article> section:
  <section class="flex-container">
   Replace paragraph text with your own text. Replace paragraph text with your own text.
   </section>

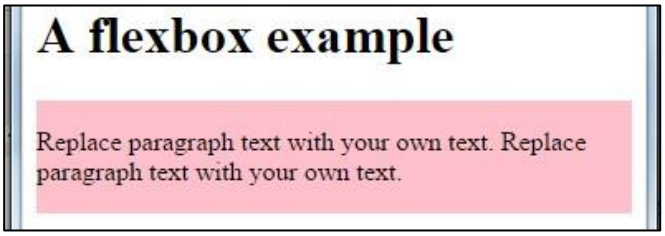
- Create a CSS file named **w5b.css** in the same folder.
- Add the following to the .css file: .flex-container { display: flex;

```
    display: -webkit-flex;  /* for safari browser */
background-color: pink;
}
```

- View the html file in your Web Browser.



- Resize the web page.
  The shape and size of the flexbox will alter.



We are about to nest flexboxes (i.e. place flexboxes within another flexbox).
Nested flexboxes are a very powerful way to layout web pages.

We will refer to the outer flexbox as the **flex container.** This will be the *parent*.
The inner flexboxes are referred to as **flex items.** They are considered to be *children*.

- Modify the file named **w5b.html**

- Add this code to the html file:
  Please take note of the **class names**, as getting these correct is crucial.

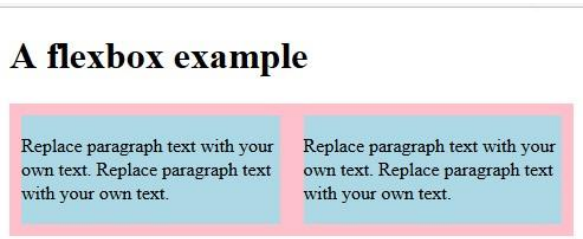
```
<header><h1> A flexbox example</h1></header>
<article class="flex-container">
  <section class="flex-item">
    <p>Replace paragraph text with your own text. Replace paragraph text with your own text.
</p>
  </section>
  <section class="flex-item">
    <p>Replace paragraph text with your own text. Replace paragraph text with your own text.
</p>   </section>
</article>
```

- Add this code to the **w5b.css** file: .flex-container { display: flex;
  display: -webkit-flex; /* for safari browser */
  background-color: pink;
  }

```
.flex-item {
 /* apply styles as needed */
background-color: lightblue;
margin: 10px;
```

```
}
```

- Load the web page in a browser



As you can see, the flex-items with the blue background are displayed within the parent flexcontainer with the pink background.

Note, default display is horizontal, i.e. row.

Screen Capture the HTML page displayed in your browser and paste into 2P_surname_studentID document.
Upload the HTML and CSS files on Canvas.