***Rich Web Applications Winter exam***

**Question 1**

**(A)**

Both web apps and applications are built with the same web technology so there is no hard-and-fast rule to differentiate between the two of them, more commonly and for our purpose we will consider the difference between their intended purpose an relative level of complexity.

Websites

The main purpose of websites is to convey information and the interaction between the server for them is usually minimal. A good example f a website can be a picture gallery or marketing campaign site, the content would mostly be static and constructed in advance. The main technologies would be HTML and CSS for building the content.

Web Apps

Web apps are more akin to desktop application, typically designed to fulfil some data processing function involving a server. The content is mostly dynamic generated in reaction to user input, and there tends to be more java script used to process user input and fetch data from the server. A good example of this would be a social messaging application.

Therefore, websites and Web applications are very different as they serve and functional in a different and more complicated fashion than a website.

**(B)**

It is a programming interface for html and xml documents, and is a standard for accessing the documents. It allows a language to manipulate the document structure, style and content of a website. Therefore it is a browsers internal representation of a the page content. DOM elements can be created using html and css.

**Diagram

Description automatically generated**

**(C)**

//href replacement to tudublin.ie

var hrefreplace = document.queryselector(‘a[href=’ www.dit.ie ‘ ] ‘);

if(hrefreplace = ‘ www.dit.ie ‘ ){

    hrefreplace.setattribute(‘ www.tudublin.ie ‘);

)

//setting colours for paragraphs

document.getElementbyID(‘p1’).style.color=’green’

document.getElementbyID(‘p2’).style.color=’blue’

document.getElementbyID(‘p3’).style.color=’red’

//paragraph align

document.queryselector(‘p’).style.textAlign=’center’

**(D)**

let discountAll = basket.map(item => item.price - (item.price \* 0.2))

let price = discount.filter(price => price < 12).reduce((price, total) =>total + price, 0)

console.log(price)

Functional programming helps separate the functions that operate on data from the data on which they operate, as a result u can have a chained pipeline of functions that transform the data and pass the results to the next stage. This helps transform complex transformations into something readable and maintainable. A key concept of functional programming is the immutability of data flow as well.

This is very important in this example as it can allow us to perform each of these operations stage by stage, effectively forming a pipeline

**Question 2**

**(A)**

Json is a string representation of a JS object or array, and it does not accommodate comments. The main advantage of json is that it can be very easily processed by JS and is also simpler and more flexible than xml requiring less code to parse making it more light weight.

Xml is a markup language and designed specifical to store data, by defining markup elements and generating customized markup language.

    const response = await fetch("https://api.github.com/users/"+myInput)

    const post = await response.json()

    let name  =  post.name

    let author = post.author

    let year=    post.year

    console.log(post)

**(B)**

*1. Puffins have black and white plumage.*

*2. Puffins have large beaks.*

*3. Puffins are very intelligent*

These three will be the first to be printed in the console as they are without call-backs and are just simple console.log with nothing attached to them.

Next the first the first call back item will now be placed in the queue and then it will be printed as it is first to enter the queue.

*4. Puffins have short wings.*

*Then the same happens to the second setTimeout as it was last to enter the queue it will now be printed as well*

*5. Puffins are good swimmers.*

Functions in the program are placed in stacks within the memory and this will hold information about what progams run. Each program is placed at the top of a stack and waits for its turn to come using the FIFO method which is first in first out.

This is generally a good method if ther isn’t any functions that will take a long time to complete however if they do then large delays can be caused to the functions later on

**Question 3**

(A)

An observable button clicks by providing a event handler which the button calls when it is clicked. In this case, the button is an observable, which notifies a number of observers in the form of event handlers.

In reactive programming, an observable is just a stream of events that you can subscribe to - i.e. observe. It is like a event line in which process will follow a predetermined path. This is possible by observing streams and handling the events which you are interested in.

**(B)**

Web Streams are an abstraction used to model asynchronous data sources. Streams are powerful technique when processing the data when you are not sure of the size or when it will be arriving to the application. This is in important for the web as you cannot always be sure of for example download size or speed, compared to data on disk, where you can see the size of files before moving the data.

There is many setting where this is important, videos, large programs and even log files.

Whatsapp for example allows the user to start watching videos before finishing the download by loading the beginning of the video. Games can be played at 80% download as well.

**(c)**

const display = document.getElementById("counter");

const increaseButton = document.getElementById("increase-counter");

let counter = {value: 0};

const incr = acc => ({value: acc.value +1});

const buttonObservable = Observable.merge(

    Observable.fromEvent(increaseButton, 'click').mapTo(incr),

)

.scan((acc, update) => update(acc), counter)

.subscribe(counter => {display.innerHTML = counter.value;});