Abdulaziz Ismail
Lab 3
Review questions

Q1)
Explain what is meant by the stream abstraction. What is the relationship between streams and the observer pattern? What are streams useful for modelling and when might you use them in Rich Web development?

- A Stream is an abstraction of a sequence of data, which is distributed in time. This abstraction is used to model asynchronous data sources.
- Streams implement the observer pattern where data is realised using the subscribe operation.
- Streams are useful for modelling asynchronous data source, potential use for them is when making an API request.

Q2)
Assume that you are building an interface to an API in your Rich Web App.
Describe in detail how you could use the RxJS library to handle asynchronous network responses to API requests. In your opinion, what are the benefits to using a streams library for networking over, say, promises? And what do you think are the downsides?

Steps to handling asynchronous network responses to API requests:

- Using the Just operator we will create a stream based on the string URL for the end point we are going to fetch data from, this will be our request stream.
- The request stream is flat mapped into our response stream, essentially the request stream will create a stream of streams and we want to create one stream so we use the flat map operator to merge the streams.
- Request url from the original stream will be passed into the flat map operation.
- We can then use the built in fetch operation which will return a promise.
- From the promise we can create another stream, using the observable fromPromise operation.
- Once we have our responses, we can subscribe to those responses and for each response that arrives we can render it into the DOM however we like.

code example

```
const request$ = Observable.just('https://api.github.com/users');

const response$ = request$
    .flatMap(requestUrl =>
        Observable.fromPromise(fetch(requestUrl))
    );

response$.subscribe(response => {
    // render `response` to the DOM however you wish
});
```

An advantage of streams is that they can help dealing with the larger data synchronisation problem we face in UI design and implementation.
In my opinion a disadvantage to streams is readability, it is very hard to read and make sense of what is happening at first glance.