Assignment Answers:

1) Cookie:
Cookie is a samall amount of data that can be stored in the form of text (usually upto 4kb) on the client's machine.
It is in the form of key=value pair. This cookie is sent with every server request and is usyally read on the Server side and processed.
Cookies can have exipry and domain set along with the value.
Cookies are prone to XSRF attacks and can be secured to a certain extent using HTTPOnly, Secure flags and Same-site flag.
One disadvantage of cookie is you need to store the session information on Server side.

Local Storage:
Local storage is used to store data on client side however, unlike cookie, it is not sent with every request.
Local storage usually is at least 5MB or more and hence can be used to store larger data and is only read on client side.
Since the data is completely on client side and readable without security, only non-confidential data should be stored here as local storage is also susceptible to XSS attacks.
Local storage data does not expire unless it is explicitly deleted.

Session Storage:
Session storage is similar to local storage but data only stays for one session.
Data is deleted upon closing of tab.

2) By wrapping entire code in a javascript function, you can control the visiblity of the members.
This way you can assess which functions should not be accessed from outside the main function block.

3) console.log(0.1+0.2) will output 0.30000000000000004
console.log(0.1+0.2==0.3) will output false.
This happens because javascript does not define different numeric datatypes and stores all numbers as double precision floating point numbers (64 bit).
Format for storage:
Number stored in 0-51 bits
Exponent 52-62 bits
Sign 63 bit
When represented the numbers in Sign Exponent and Mantissa formation i.e. IEEE754 and converting it back to decimal yields the mentioned value.

4)

```
function sum(a,b){
   if(b===undefined){
      return function(c){ return a + c}
   }
   return a + b;
}
console.log("Normal way: "+ sum(2,3));
console.log("Different way: " + sum(2)(3));
```

OUTPUT:

```
Normal way: 5
Different way: 5
```

This is achieved by returning a function in case of missing argument, catching the second argument and using it.

5) Output of the code is :

```
undefined
2
```

Explanation: This happens because first time it does not get the context of "this" and hence the length is undefined.
Second time, the argument object is passed and since we are passing 2 values, the length printed is 2.