1. **data route** 裡 **sum function** 以及作業 **5 twoSum** 目前寫法的 **BigO** 各是多少呢？

sum function:

```
//recursion
    const sum = (n) => {
      if (n === 0) {
        return 0;
      } else {
        return n + sum(n - 1);
      }
    };
```

sum function採用recursion的寫法，最糟的狀況(也就是每次)會呼叫n次(從n到0)，而每次呼叫都是執行一次加法以及一個return，故每次呼叫皆為常數級別O(1)，而呼叫n次則為O(n)。

故**sum function** 為**O(n)-algorithm(linear complexity)**

Another Sum function:

```
const sum = (n) => {
  return (n * (n - 1)) / 2;
};
```

another sum function採用公式解的寫法，每次呼叫僅需要常數時間。

故**another sum function** 為**O(1)-algorithm**

twoSum:

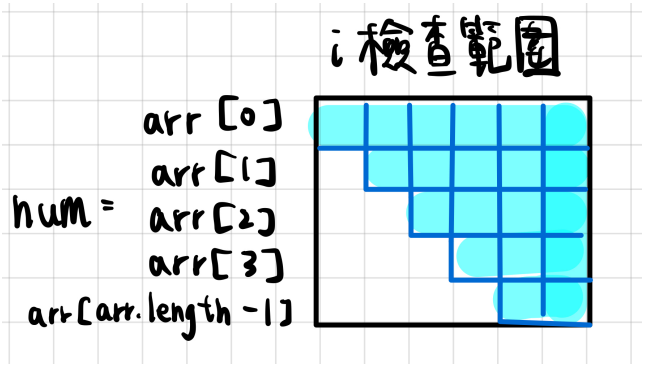
```
function twoSum(nums, target) {
  let answer = [];
  nums.forEach((num, index, arr) => {
    for (let i = index + 1; i < arr.length; i++) {
      if (num + arr[i] === target) {
        answer.push(index, i);
        break;
      }
    }
```

```
  });
  return answer;
}
```

twoSun採用的方式為:

當num為arr[index]時, 測試arr[index+1]到arr[arr.length-1], 觀察是否有num+arr[i] = target的情況發生。而每次呼叫都是執行一次加法, 故每次呼叫皆為常數級別O(1)。

此funciton最糟糕的狀況為測試到num = arr[arr.length-2]與arr[arr.length-1](arr的最後兩項)相加才等於target的狀況。此時呼叫次數為:



呼叫次數+1=arr.length*arr.length*½(三角形面積)

$time\Theta(\frac{n^2}{2}) = time\Theta(n^2)$

故**twoSun** 為**O(n^2)-algorithm**

2. cookie 是什麼, 為什麼需要這個機制？使用這機制有什麼要注意的地方嗎？

cookie是將用戶瀏覽以及偏好紀錄保留下來的一種工具, 因為http為不會記住state的protocal, 故用戶每次進入相同網站皆不會保留上次紀錄。如若要讓用戶有能承接上次進入此網站的紀錄(ex:購物車、瀏覽商品紀錄...等等)的體驗, 就可以使用cookie紀錄用戶的一些資料, 並且存在browser(瀏覽器, 也就是client客戶端)。如此一來在進入相同網站時, browser便可以將cookie交給server, 使得server可以調用cookie中的資訊, 為客戶提供具連續性的體驗。

使用cookie需要注意:

1. cookie可在用戶端的browser內被自動修改, 故應該在傳送cookie給用戶端以前對其進行簽名, 稱為signed cookie。

2. cookie儲存的資料量有限, 最多不能超過4095bytes。

3. cookie可能會過期。

為解決以上問題，我們可以在server端使用session作為代替cookie的工具。

3. 除了 **node.js** 外，有其他的 **javascript runtime** 嗎？

除了Node.js以外，還有其他的JavaScript Runtime!

1. Web Browser:瀏覽器可說是最常被使用的JavaScript Runtime。現代的Web Browser皆包含JavaScript引擎，可以讓JavaScript在網頁上運行。

2. Deno: Deno是一個基於V8引擎的JavaScript和TypeScript Runtime，由Node.js的創始人之一Ryan Dahl開發，它提供了更安全和更現代的開發環境。Deno在單個可執行檔中扮演執行環境和套件管理系統的角色，不需要將其分開。代表其不需依賴於npm，而是使用URL來引入modules。

3. Bun: Bun應該是近期最熱門、討論度最高的JavaScript Runtime。它的特點就是速度快，Bun 做到能處理 Deno 兩倍多的請求量 (也就是 Node.js 四倍多的請求量)。且Bun 可以兼容 CommonJS 與 ESM，意即可以同時用 import 與 require。並且Bun與Node.js兼容性做得很好，使開發者可以無痛轉移到Bun。

reference:
https://geekflare.com/best-javascript-runtime-environments/
https://www.explainthis.io/zh-hant/swe/what-is-bun