**分散式系統  
Lab: Remoting**

請務必填寫學號系級姓名，以免成績登錄錯誤。

學號系級姓名:

請依問題與提示在指定區域回答問題，並依規定時間內上傳至moodle。

操作一: SOAP-based Web Services開發 (平台: Node.js)

1. 建立一個新的資料夾「lab-remoting」，在此目錄下，新建一個soap目錄
2. 在此lab-remoting目錄中建立一個新的package.json檔案，內容如下:

{

"name": "2021-dslab-remoting",

"version": "1.0.0",

"dependencies": {

"soap": "^0.36.0",

"@grpc/grpc-js": "^1.2.2",

"@grpc/proto-loader": "\*"

}

}

1. 將Adder.wsdl、AddMu.wsdl、soapClient.js與soapServer.js拷貝到lab-remoting/soap目錄中。
2. 開啟並了解soapServer.js程式碼的功能與意義:
3. 請貼上含「讀入wsdl檔」功能的敘述(請貼上整個statement，也就是分號前的所有程式碼)

答:

1. 請貼上含「實作add並回傳x和y之和的實作」功能的敘述(請貼上整個statement)

答:

1. 在程式中，建立http server後，指派給一個變數，該變數的名稱為何? 這個http server傾聽的通訊埠號(port number)為何?

答:

1. soap.listen(…)中傳入了四個參數，包含WSDL、http server、服務的實作與一個此服務的掛載網址，請寫出此網址為何?

答:

1. 開啟並了解soapClient.js程式碼的功能與意義:
2. 引入soap函式庫後，程式呼叫了soap的createClient的方法，這個方法傳入二個參數，其中一個是SOAP Server的WSDL的位址。請問此位址為何?  
   答：
3. 由createClient方法所傳入的回呼函式中有二個參數，分別為err與client，由client我們可以直接呼叫client.add來呼叫SOAP Server上的加法函式。其中，args指的就是傳入遠端add呼叫的參數x與y，請問x與y的值各為何?  
   答：
4. 執行soapServer.js，在console中應出現server initialized
5. 執行soapClient.js，觀察console所印出的執行結果。
6. 修改soapClient.js中的args，試著藉由呼叫SOAP Server計算x=10, y=20的結果。將soapClient.js所印出在console中的SOAP訊息貼在下面。

答：

操作二: 寫作新的SOAP 乘法(multiply)服務

1. 請根據操作一中的觀察，修改soapServer.js，將引入的wsdl檔案由Adder.wsdl改為AddMul.wsdl。
2. 根據AddMul.wsdl中的註解，參考add服務的定義，定義乘法(multiply)服務的相關wsdl宣告。將修改後的AddMul.wsdl貼在下方

答：

1. 修改soapServer.js，在service中新增multiply服務與實作

提示:

const service = {

CalculatorImplService: {

CalculatorImplPort: {

add: function (args) {

return {result: args.x + args.y};

},

*multiply: function(args) {*

*….*

*}*

}

}

};

1. 修改soapServer.js，在修改存取網址為「AddMul」:

soap.listen(server, '/AddMul', service, xml, function () {

console.log('server initialized');

});

1. 執行soapServer.js，在console中應出現server initialized
2. 修改soapClient.js，將url改為<http://localhost:8192/AddMul?wsdl>

const url = 'http://localhost:8192/AddMul?wsdl';

1. 修改soapClient.js，將client.add改為client.multiply

client.multiply(args, function (err, result, rawResponse, soapHeader, rawRequest) {

if (err) console.log(err);

console.log(rawRequest);

console.log('');

console.log(rawResponse);

});

1. 執行soapClient.js，觀察console所印出的執行結果是否正確。
2. 修改soapClient.js中的args，試著藉由呼叫SOAP Server計算x=10, y=20的結果。將soapClient.js所印出在console中的SOAP訊息貼在下面。

答：

操作三: gRPC開發 (平台: Node.js)

1. 在「lab-remoting」目錄下，新建一個grpc目錄
2. 將拷貝到lab-remoting/grpc目錄中
3. 開啟並了解server.js程式碼的功能與意義:
4. rpc SayHello (HelloRequest) returns (HelloReply) {}中用到二個訊息 HelloRequest和HelloReply，

message HelloRequest {

string name = 1;

}

message HelloReply {

string message = 1;

}

請問裡面的name=1、message=1，是什麼意思?

答:

1. 找出程式從那裡讀入helloworld.proto定義檔?

(請整個敘述貼在下方)

答：

1. 觀察sayHello函式中如何處理傳入訊息(如何取得參數值name)之後回傳(本題不需作答)。
2. 觀察server.addService()中，sayHello函式是如何登錄到服務中
3. 依序執行server.js、client.js觀察執行結果。
4. 請修改helloworld.proto、server.js與client.js，加入一個新的遠端gRPC函式。
5. 功能，傳入2個值x、y，回傳results為x+y的結果
6. 名稱: Add (Helloworld.proto), add(server.js和client.js):
7. 訊息與參數: AddRequest，參數依序為int32 x與int32 y
8. 回傳訊息與參數: AddReply，參數為int32 result
9. 修改server.js模仿 function sayHello加入新的函式function add
10. 修改server.js，在server.addService中登錄add函式
11. 修改client.js，模仿client.sayHello新增client.add
12. 測試程式執行結果
13. 請將修改後的helloworld.proto、server.js與client.js貼下面。

helloworld.proto

答：

server.js

答：

client.js

答：