一、DB課程形式

從db 的pass function query出的學生修過課程之課程形式為如右圖

{ cos\_code: 'IOE5066',

cos\_cname: '雲端計算系統與實務',

cos\_code\_old: null,

cos\_cname\_old: null,

cos\_ename: 'Cloud Computing Systems and Applications',

pass\_fail: '通過',

cos\_type: '大學部修研究所課程',

score: '95',

score\_level: 'A+',

cos\_typeext: '',

type: null,

brief: '',

brief\_new: '',

cos\_credit: '3',

year: '106',

semester: '1',

offset\_type: null,

tname: '陳添福' }

cos\_code為課號

cos\_cname為課程中文名稱

cos\_ename為課程英文名稱

pass\_fail表示學生為通過不通過

cos\_type會顯示該課為必修、選修、、大學部修研究所課程、外語、體育或通識等

type適用於資工系的課，會根據該學生所屬組別，顯示其為核心、副核心、資工/網多/資電組核心等課程屬性，詳見後面範例

score為成績

score\_level為成績(有些課的成績為等級制)

cos\_typeext為補充課程屬性，如是否為英文授課，詳見後面範例

brief為用來表示通識向度

brief\_new 為用來表示新制通識向度

cos\_credit為課程學分

year為該課開課年度

semester為該課開課學期（上or 下）

offset\_type 為表示此課是否為該學生的抵免或免修課，若都不是則為null

tname為授課老師

以下為不同課程的課程範例

(每個課程根據要表達的重要屬性不同會省略其他屬性)

{(C)

cos\_cname: '影像處理概論',

cos\_type: '選修',

type: '核心',

},

{(B)

cos\_cname: '人工智慧概論',

cos\_type: '選修',

type: '資工核心',

}

{ (A)

cos\_cname: '俄羅斯歷史與文化',

cos\_type: '通識',

brief: '歷史/進階(96 )',

brief\_new: '校基本素養(106)',

}

要特別說明的為

{ (F)

cos\_cname: '微處理機系統實驗',

cos\_type: '必修',

}

{ (E)

cos\_cname: '無線網路概論',

cos\_type: '選修',

type: '副核心',

}

{ (D)

cos\_cname: '體育－健康體適能',

cos\_type: '體育',

cos\_typeext: '體育',

brief: '體育必修(87 )',

}

1. 由於該生為網多組學生，因此(B)對他來說為type資工組核心，而(C)影像處理為網多組核心，因此type顯示核心
2. (E)無線網路為網多組副核心，因此type顯示為副核心
3. (F)為三組必修，因此cos\_type顯示為必修

二、db的table

由於有些課程有英文授課，如計算機組織，因此課名會有‘計算機組織‘及’計算機組織(英文授課)’，為了讓課程名稱統一，即都不會顯示‘英文授課’，DB 建了一個資工系的課號對照課名的表格，像是，由於不同年的同一門的課號可能會不一樣，因此會有多個課號對到同一個課名，例如：

|  |  |
| --- | --- |
| 計算機組織 | DCP1234, DCP5678, DCP1123 |
| 資料結構 | DCP2222, DCP3333, DCP5555 |

在table.js裡 db的group function為會根據每個學生顯示資工系的必修、核心及副核心的課，以及這些課相對應得屬性，我們則會依據這些屬性建立表格

|  |  |
| --- | --- |
| Compulse 必修 |  |
| Core 核心 |  |
| Vice 副核心 |  |
| Others 他組合心 |  |
| Elective 選修 | （其實不需要） |
| Total 全部資工系的課 | （其實不需要） |

我們會在processCourse function裡依據登入學生的program將這些課分類到相對應的地方

三、課程分類方法

目前畢業預審將課程分為11類：

共同必修、核心課程、副核心與他組核心、專業選修、外語、通識、其他選修、體育、藝文賞析、服務學習及抵免研究所課程

* 我們的資料結構為

一個大陣列裡面包有9個小陣列，也就是那9種課程屬性，每一個小陣列之資料結構為

title表示是哪一種課程屬性

credit表示在該課程屬性中學生共修了幾學分

require表示畢業標準規定該課程屬性須修幾學分

course即為該學生修了哪些課

{

title: '共同必修',

credit: 0,

require: 0,

course: []

}

而每一堂課的資料結構分兩種形式

左邊為未排序部分的課程形式，右邊由於排序課程時會需要用到較多特性，每個課程有較多的屬性。cn/en表示中英文課名，score表示分數，若為-1表示尚無成績，reason會標示是否為資工系的課(用外系的課抵免資工系之課要標示之用)，type是表示此課為必修、選修或通識等; originalCredit表示該課原始的學分; realCredit為該課真正的學分(這樣分是因為，在排序課程時，有些課會被分割，像是普物4學分多的1學分可以放到其他選修); complete則表示是否通過這門課; code為課號; grade為等級制表示法的課程表示成績之用; semester為表示哪學期的課(上學期為1下學期為2暑修為3); year為修課年度

{

cn:'',

en:'',

score: -1,

realCredit:0,

reason: 'CS',

complete:true,

grade:'0',

year: '',

semester: ''

};

{

cn:'',

en:'',

score: -1,

reason: 'CS',

realCredit:0,

type:'',

originalCredit:0,

complete:'0',

code:'',

grade:'0',

year: '',

semester: ''

};

我們分類的順序為：

**分類學生的抵免課程(Othercourse.js) -> 分類學生修過課程中非資工系的課程以及非在必修、核心、副核心及他組核心的資工系的課(即非在table裡的課) (Othercourse.js) -> 分類學生修過課程中在table裡的課程(CScourse.js)**

* 分類學生的抵免課程

未排序的分類寫在/nctu-cs-ca/routes/user/course/original資料夾裡

抵免課程為該學生可能修別系的課要拿來抵免資工系的課，則會顯示資工的哪一堂課可被抵免。這邊的分類較為簡單，主要是依據cos\_type來分類。

* 分類非在table裡的課
* 主要也是依據cos\_type來分類。首先會先判斷該課是否在table內，若是則跳過。由於資工系開的課課號會是'DCP', 'IOC', 'IOE' 及 'ILE’ 開頭，因此當遇到這些課號開頭的課時，會先看是否為資工系開的服務學習，若是則須歸類為服務學習，另外也須判斷該課是否為導師時間，若是則須歸在必修(因為table裡並未存導師時間)，其他課則可歸類為專業選修。
* 若課號開頭為ART則歸類為藝文賞析
* 若cos\_type為外語，則歸類為外語
* 若brief為軍訓，則歸類為其他選修
* 若cos\_type為通識，則會再看其brief，紀錄該課程歸為哪一個向度
* 若課號開頭為PYY，則歸類為體育
* 若cos\_typeext為服務學習，則歸類為服務學習
* 若cos\_cname為導師時間，則也歸類為必修的抵免課程，以及歸類為其他選修
* 其他未符合上述條件者就歸類為其他選修
* 分類有在table裡的課程
* 有在table裡的課程即為學生修過的必修、核心、副核心及他組核心及副核心的課。由於table裡有將課程分為必修、核心、副核心及他組核心及副核心，判斷時會先看學生修了哪些必修 -> 核心 -> 副核心 -> 他組核心及副核心的課程。
* 這裡在判斷學生修了哪些課時，是從table裡的課號到學生pass的課中來找是否有修過這門課，例如

table裡的核心課程中有下列表格：

|  |  |  |
| --- | --- | --- |
| Core | 網路程式設計 | DCP1234, DCP2234 |
| 計算機圖學 | DCP5678, DCP7785 |

學生pass的課程如下

{ DCP1234, DCP3456, DCP0987, DCP5678…..}

則我們判斷學生是否修過核心的網路程式設計的課即為用Core中網程設的DCP1234到學生的pass裡看是否有DCP1234，如此才能達到大家修的課名會顯示統一名稱的目的。

四、排序課程

所謂排序課程就是除了我們會把該學生修的課分好類之外，我們還會幫他排成最接近畢業標準的方式。例如像是多的專業選修學分可以放至其他選修，因此在未排序的部分，我們只需將課程分類好，但在排序課程的部分，我們在將某門課歸類為專業選修前，會先看專業選修學分是否已滿，若已滿，則將它放在其他選修。在歸類其他類別的課程前，在排序部分，我們都會多了這道判斷的程序。不過每一屆的畢業標準以及每堂課可以放的位置的限制可能會不同，會依不同屆而異。

排序課程的部分我們是寫在/nctu-cs-ca/routes/user/course/revise資料夾裡面

五、抵免研究所課程

由於在做畢業預審時，我們可以決定要將哪些課當作研究所課的抵免，因此我們有個欄位為抵免研究所課程，學生可以自行將想抵免研究所的課拉至這邊。若課程被拿來抵免研究所的課，則不會將此課採計為畢業學分。

六、編輯課程

若學生對於我們幫他們排序的課程不滿意，學生可以按編輯課程，自行移動課程，決定自己要怎麼安排課程的位置，不過我們這邊會幫學生做判斷，提醒學生這樣的移動合不合法。在學生每一次的移動課程，我們會判斷這門課是否可以從這個位置拉到那個位置，例如韓文課不可拉到專業選修。這個判斷我們寫在 /nctu-cs-ca/routes/user/students/graduate/graduateChange資料夾裡。

七、討論紀錄

以下是一些我們記錄在hackMD上的後端討論紀錄

<https://hackmd.io/kkeEZs9wQFitUw69cYDZTg>

<https://hackmd.io/RTjIl4Z3QYO88SzHIilchA>

<https://hackmd.io/CADxUo-BT0SMiLedMwfpiw>

八、補充nodeJS get/post request處理

我們在做每一項資料處理時，都是運用一個個function來處理資料，在做畢業預審的分類及排序時，都是在/nctu-cs-ca/routes/user/course資料夾裡中寫的各種function裡做事。

在處理每一項get request時，會用middleware，有就是call上述那些function做事，如下例：

var getStudentId = require('../../course/getStudentId');

var query = require('../../course/query');

var StudentId = getStudentId.getStudentId.studentId;

var StudentProfile = query.query.queryProfile;

var queryFree = query.query.queryFree;

router.get('/students/graduate/revised', StudentId, StudentProfile, queryFree, queryGeneral, queryPass, queryChange, queryCourse, queryNow,queryRule, processOther, processCS, currentOther, currentCS, processRestore, processResult, function(req, res){

res.send(res.locals.courseResult);

});

這個get request經過了StudentId, StudentProfule, queryFree 等等這些middleware，來達成目標，最後將結果send出去

注意這些function的include方式，先include寫這些function的file，在指定要導入這個file裡的哪個function。