**Bezier Curve**

**Draw four arbitrary points on a canvas.**

1.建立起畫布canvas

在html建立畫布，JavaScript也要引入html的元件

2.當畫布建立好，要能觸發繪製的功能

在html建立繪圖按鈕，JavaScript也要引入html的元件，同時也建立清除按鈕。

3.需要分辨視窗，引入CSS

畫布採用淺灰色，讓他好識別

4.思考要如何畫4個點

(1).引用jQuery的library做處理<jQuery可以支援動畫、滑鼠操控、網頁的行為>

(2).寫法是，先偵測滑鼠座標，偵測滑座標後，按下去做確認點，並且生成一個圓。

<同時，我給予了點text，並將偵測的座標放到畫布上>

**Draw a Bezier Curve according to the given four points.**

1.先了解貝茲曲線方程式

(1).為建構二次貝茲曲線，可以中介點Q0和Q1作為由0至1的t：

可以使用三個點 P0、P1、P2，在 P0 與 P1 間直線的四分之一處找個點 Q0，在 P1 與 P2 間直線的四分之一處找個點 Q1，這時 Q0 與 Q1 會構成一條直線，這時也在 Q0 與 Q1 之間直線的四分之一處找個點 B0。

接著類似的作法，只是將四分之一變成二分之一、四分之三，分別找出 B1、B2 好了，接著將 P0、B0、B1、B2、P2 連起來有點像是曲線了。

(2).整理一下，貝茲曲線方程式為:

|  |
| --- |
| B(t) = (1 - t) \* (1 - t) \* P0 + 2 \* t \* (1 - t) \* P1 + t \* t \* P2, t ∈ [0, 1] |

(3).因此，我的做法是，定義一個t的參數（貝茲函數的參數式0<=t<=1），然後，我的t每次都加上0.01（固定值）

2.貝茲曲線的寫法

(1).定義一個遞迴

|  |
| --- |
| function factorial(num) { //遞迴階層  if (num <= 1) {  return 1;  } else {  return num \* factorial(num - 1);  }  } |

(2).導入曲線方程

|  |
| --- |
| x += factorial(n) / factorial(index) / factorial(n - index) \* item.x \* Math.pow(( 1 - t ), n - index) \* Math.pow(t, index)  y += factorial(n) / factorial(index) / factorial(n - index) \* item.y \* Math.pow(( 1 - t ), n - index) \* Math.pow(t, index) |

(3).讀取x , y的座標，並置入到此方程式中

4.最後，顯示出要畫的曲線

<這邊引用jQuery做了一個動畫>

**Allow a user to move any one of the four points to update the corresponding Bezier Curve accordingly.**

1.偵測畫布上的點

2.如果這個點是貝茲曲線上的點，則就選擇

3.畫布更新：

(1).當選了點移動後，畫布必須重新設定

(2).新的點必須放入貝茲方程並重新計算，並產生曲線

(3).每次產生的曲線，根據移動，還有位移的改變，其曲線與連接的線做重新更新。

**Must use a recursive approach as described in the class, that is, you can't call existing library functions to draw the Bezier curve directly.**

1.本程式針對動畫與點的操作皆使用jQuery

2.本程式的遞迴設計為

(1).定義一個遞迴

|  |
| --- |
| function factorial(num) { //遞迴階層  if (num <= 1) {  return 1;  } else {  return num \* factorial(num - 1);  }  } |

(2).導入曲線方程

|  |
| --- |
| x += factorial(n) / factorial(index) / factorial(n - index) \* item.x \* Math.pow(( 1 - t ), n - index) \* Math.pow(t, index)  y += factorial(n) / factorial(index) / factorial(n - index) \* item.y \* Math.pow(( 1 - t ), n - index) \* Math.pow(t, index) |