CHAPTER

03 流程控制

3.1 選擇結構

學習程式語言首先要認識資料型別才能對資料賦予(宣告)變數，接著學習該程式語言所提供各種敘述的語法，其中控制程式流程的敘述是最基本的語法之一，熟悉這些敘述便可寫出簡單的程式出來。

一個程式不外乎由循序、選擇和重複結構三者所提供的敘述組合而成。循序結構敘述的特性是由上而下逐行地執行。選擇結構敘述的使用時機是當程式執行時，欲改變程式執行的流程時使用。重複結構敘述俗稱迴圈(Loop)，使用時機是當程式中某個敘述區段需要重複執行多次時使用。因此，欲設計出一個具有結構化的程式，除了本身要具有清晰的邏輯分析能力外，只要能熟練上面三種結構敘述的使用方法才能寫出符合要求的程式。

首先在本章介紹 C# 所提供的選擇敘述，其做法就是當程式執行時，透過條件式的判斷，若條件式結果為真(true)，則執行屬於條件式為真的敘述區段，若條件式的結果為假(false)，則執行屬於條件式為假的敘述區段。兩者執行完畢都會回到同一位置，繼續執行接在後面的敘述。所以，選擇結構是用來改變程式執行的流程。

3.1.1 if…else…敘述

設計程式時常會碰到「若 ... 則 …」或是「若 ... 則 ... 否則 ...」，此種情形便需要使用到 if ... 敘述(語法 1)或是 if ... else ... 敘述(語法 2)。 由下面語法 2 可知，若滿足 <條件式> 就執行 [敘述區段 1]，不滿足 <條件式> 就執行 [敘述區段 2]。

語法 1

```
if(條件式)
{
   [敘述區段]
}
```

語法 2

```
if (條件式) {
   [敘述區段 1]
}else {
   [敘述區段 2]
}
```

說明

1. 若[敘述區段]內只有一行敘述，大括號可省略，如下：
   若[敘述區段]內有兩行敘述以上(含)必須使用大括號頭尾括住。
2. **<條件式>** 可為關係運算式，或是如下面 [例 1] 由多個關係運算式組成，中間使用邏輯運算子來連接。
3. C# 的 bool 布林資料型別和其他資料型別無法進行轉換，但在 C++裡面 bool 布林資料型別的資料是可以轉成 int 整數資料型別，所以在 C++裡面 true 等於非零值、false 等於零。如下面 [例 3]在 C++中可正常編譯，但在 C# 中會產生編譯失敗。

【例 1】 由鍵盤輸入年齡，若年齡介於 10~19 歲之間，顯示 "你的年齡是 xx，是青少年"；若超出範圍顯示 "你的年齡是 xx，不是青少年"。

(檔名：ConsoleifElse1.sln)

```
int age ;
Console.Write("請輸入年齡 : ");
age = int.Parse (Console.ReadLine());
if (age>=10 && age <=19)
      Console.WriteLine($"你的年齡是 {age} , 是青少年");
else
      Console.WriteLine($"你的年齡是 {age} , 不是青少年");
Console.Read();
```

【例 2】 延續上例將條件式改採 OR 邏輯運算子編寫程式。(檔名: ConsoleifElse2.sln)

```
int age ;
Console.Write("請輸入年齡：");
age = int.Parse (Console.ReadLine());
if (age < 10 || age >19)
      Console.WriteLine($"你的年齡是 {age} , 不是青少年");
else
      Console.WriteLine($"你的年齡是 {age} , 是青少年");
Console.Read();
```

【例 3】注意下列敘述在 C++中可正常編譯，但在 C# 中會產生編譯失敗：

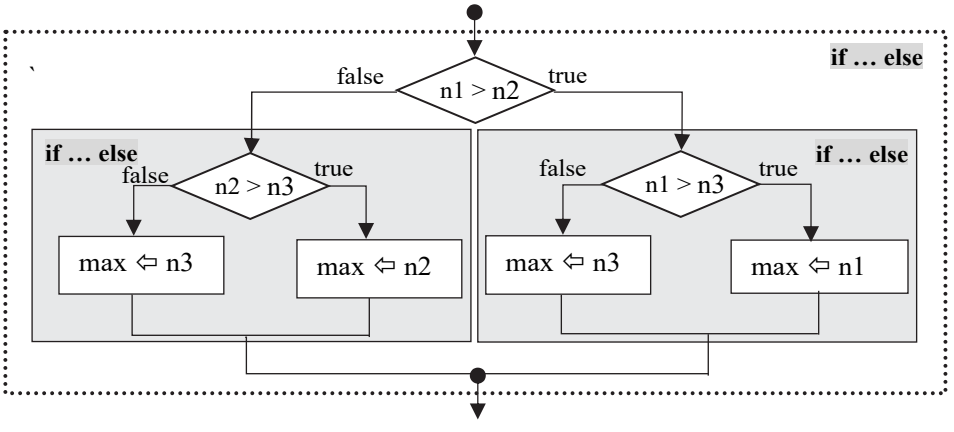
```
int k = 10;
if (k)
      Console.WriteLine("結果不等於零! ");
Console.Read();
```

下面敘述是 C# 正確的寫法，將上面 if(k)選擇敘述改成 if(k != 0) 編譯時才不會發生錯誤。

```
int k = 10;
if (k != 0)
      Console.WriteLine("結果不等於零! ");
Console.Read();
```

3.1.2 巢狀選擇敘述

若 if ... else 敘述裡面還有 if ... else 敘述就構成巢狀選擇結構，其使用時機是資料需比較兩次(含)以上時使用。譬如：下面流程圖是三個同性質的數值做比較採巢狀 if 來找出三數中的最大值。



範例：ConsoleifElse3.sln

試將上面流程圖先透過鍵盤輸入三個不同的數值後，再使用巢狀選擇敘述找出三數中的最大值，請依下圖方式顯示在螢幕上。

執行結果

程式碼 FileName:Program.cs

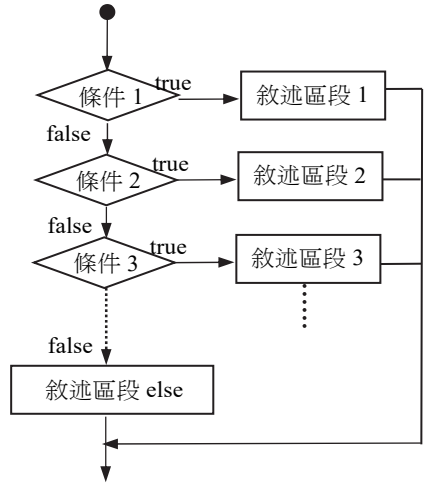
```
01 namespace ConsoleifElse3
02 {
03   class Program
04   {
05     static void Main(string[] args)
06     {
07        int n1, n2, n3, max;
08        Console.Write("1. 請輸入第一個數值 : ");
09        n1 = int.Parse(Console.ReadLine());
10        Console.Write("2. 請輸入第二個數值 : ");
11        n2 = int.Parse(Console.ReadLine());
12        Console.Write("3. 請輸入第三個數值 : ");
13        n3 = int.Parse(Console.ReadLine());
14        if (n1 > n2)
15           if (n1 > n3)
16              max = n1;
17           else
18              max = n3;
19        else
20           if (n2 > n3)
21              max = n2;
22           else
23              max = n3;
24        Console.WriteLine($"\n=== {n1},{n2},{n3} 三數中最大值為 : {max} ");
25        Console.Read();
26     }
27   }
28 }
```

3.1.3 if…else if…else…敘述

設計程式時，若碰到「若... 則... 否則... 再否則...」，此種情形便需要使用到 if ... else if ... else if ... else ... 敘述來完成。如下面語法，若 <條件 1> 的結果為 true，則執行 [敘述區段 1]，接著便結束整個 if 敘述；若 <條件 1> 的結果為 false，則檢查 <條件 2> 的結果，若為 true 則執行 [敘述區段 2]，接著便結束整個 if 敘述。若所有的 <條件> 都不滿足時，才執行接在 else 後面的 [敘述區段 else]。

語法

```
if (條件 1)
{
      [敘述區段 1]
}
else if (條件 2)
{
      [敘述區段 2]
}
   ……
else if (條件 N)
{
      [敘述區段 N]
}
else
{
      [敘述區段 else]
}
```



範例：ConsoleTest.sln

使用 if … else if … else 多項選擇敘述製作選擇題測驗程式。題目如下：

[題目]： 試問 Visual Studio 可開發下列哪種應用程式？

1.視窗程式　2.Web 程式　3.裝置應用程式　4. 以上皆是

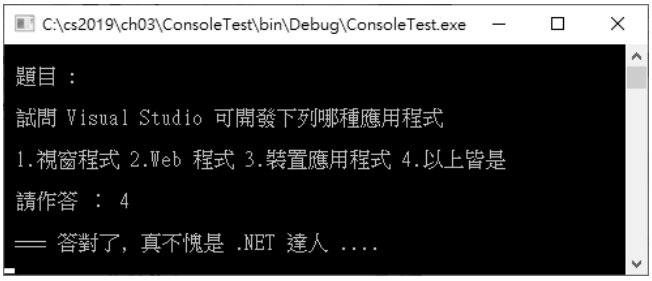
依使用者作答情形給予下列回應訊息：

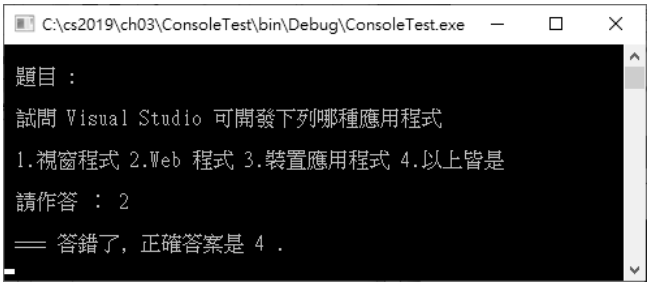
① 答案輸入 1 或 2 或 3，顯示 "答錯了, 正確答案是 4 ."

② 答案輸入 4，顯示 " === 答對了, 真不愧是 .NET 達人 ...."

③ 當答案非 1~4 時，顯示 " === 無此選項 .... "

執行結果





程式碼 FileName:Program.cs

```
01 namespace ConsoleTest
02 {
03   class Program
04   {
05     static void Main(string[] args)
06     {
07       Console.WriteLine("\n 題目 : \n");
08       Console.WriteLine(" 試問 Visual Studio 可開發下列哪種應用程式\n");
09       Console.WriteLine(" 1.視窗程式 2.Web 程式 3.裝置應用程式 4.以上皆是\n");
10       Console.Write(" 請作答 : ");
11       // 宣告 ans 字串變數用來存放使用者由鍵盤輸入的答案
12       string ans = Console.ReadLine();
13       // 判斷 ans 是否為 1, 2, 3 其中之一
14       if (ans == "1" || ans == "2" || ans == "3")
15         Console.WriteLine("\n === 答錯了, 正確答案是 4 .");
16       else if (ans == "4")    // 判斷 ans 是否等於 4
17         Console.WriteLine("\n === 答對了, 真不愧是 .NET 達人 ....");
18       else                    // 當 ans 不等於 1 ,2 ,3, 4 時執行下列敘述
```

25.4 使用 Open AI 製作聊天機器人

25.4.1 ChatGPT 聊天機器人初體驗

如下是使用 OpenAI 製作 ChatGPT 聊天機器人的步驟，完整實作可參閱 ChatGPT01 範例。

Step 01 前往 OpenAI 申請金鑰 (Key)，步驟可參考 25.3 節。

Step 02 在專案安裝 OpenAI 套件。

Step 03 使用 OpenAI API 物件進行對話操作，並取得聊天機器人的回應 (ChatResut)，寫法如下：

```
// 使用 OpenAI 金鑰建立 OpenAIAPI 物件
OpenAIAPI oenai 物件 = new OpenAIAPI("OpenAI API 金鑰");

// 透過 OpenAIAPI 物件執行對話操作並取得回應結果 CharResult 物件
ChatResult chatResult =
    await oenai 物件.Chat.CreateChatCompletionAsync(new ChatRequest()
    {
        Model = Model.ChatGPTTurbo,                  //使用模型
        Messages = new ChatMessage[]                 //指定聊天訊息
        {
            new ChatMessage(ChatMessageRole.User, prompt)   //聊天提示
        }
    });
```

Step 04 ChatResult 物件包含 OpenAI 模型回應的相關資訊，如生成的文本、回應的信度…等。ChatResult 物件的 Choices[0].Message.Content 屬性可取得生成的文本。

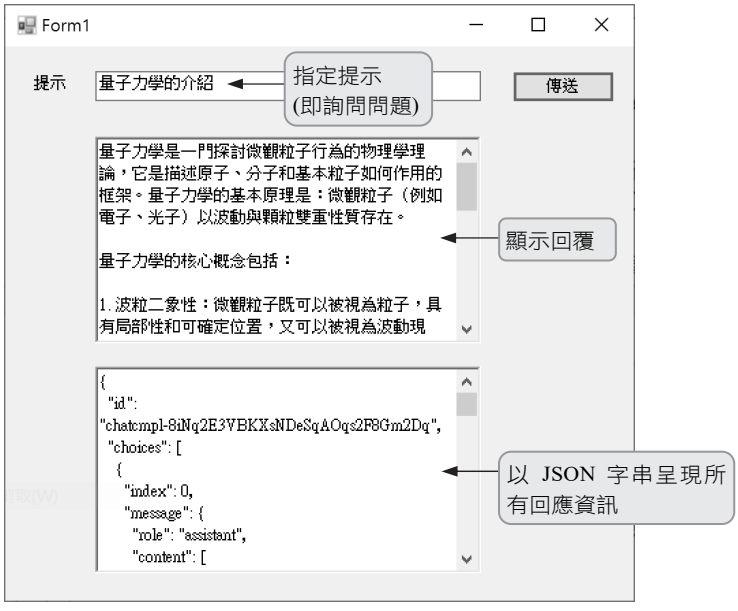
```
// 將回傳結果顯示在 richTextBox1 中
var reply = chatResult.Choices[0].Message.Content;
richTextBox1.Text = reply;
```

接著以 ChatGPT01 範例來練習製作聊天機器人。

範例：ChatGPT01.sln

使用 OpenAI 的 API 金鑰製作簡易的 ChatGPT 聊天機器人。當使用者指定提示(即詢問問題)後按下 傳送 鈕即將提示傳送給 OpenAI API。接著 OpenAI API 即會將回覆結果顯示於上方的多行文字方塊；而所有回應資訊會以 JSON 格式顯示在下方的多行文字方塊內。

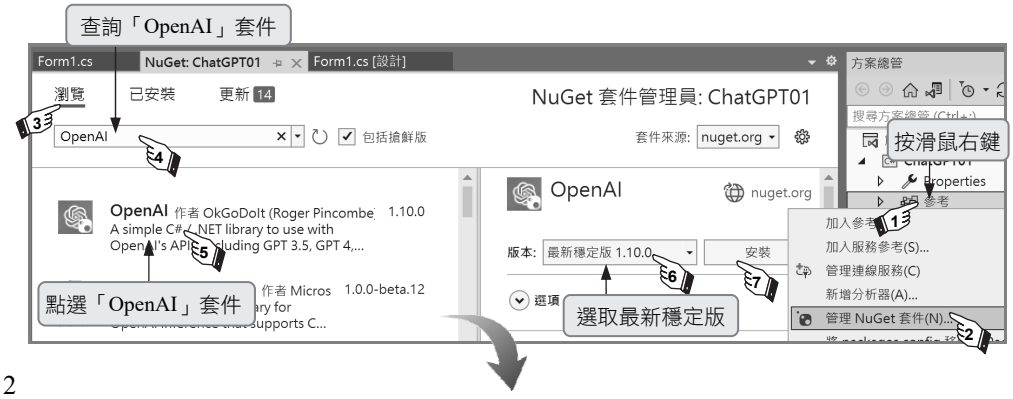
執行結果

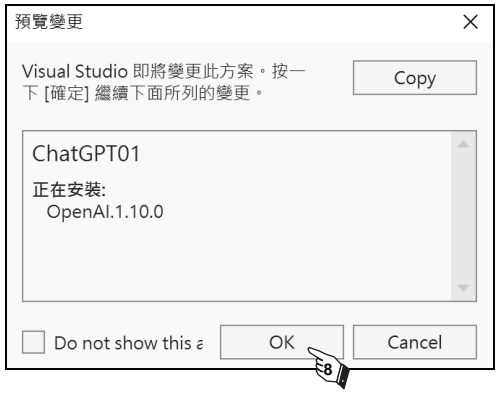


操作步驟

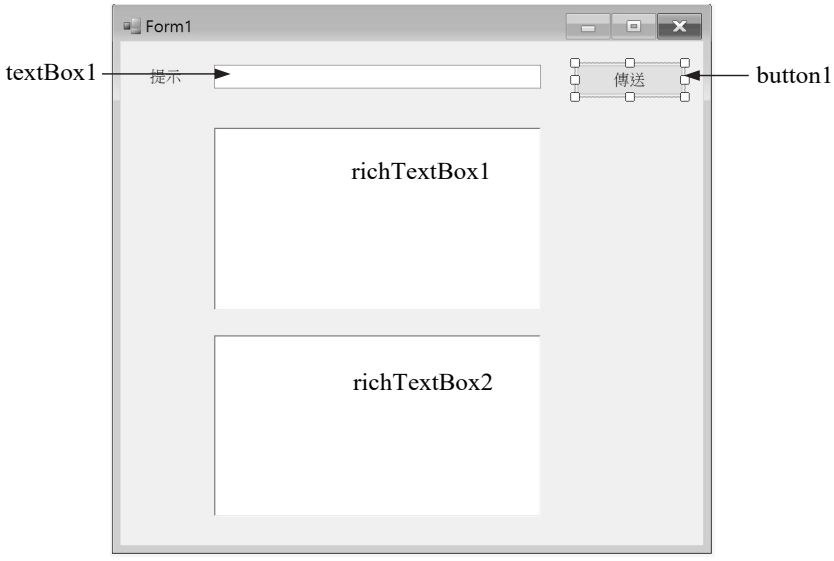
Step 01 前往 OpenAI 申請金鑰 (Key)，步驟可參考 25.3 節。

Step 02 安裝 OpenAI 套件。在方案總管視窗的「參考」按滑鼠右鍵執行【管理 NuGet 套件(N)】，接著依圖示操作安裝「OpenAI」套件。





Step 03　建立表單輸出入介面



Step 04　撰寫程式碼

程式碼　FileName:Form1.cs

```
01 using OpenAI_API;
02 using OpenAI_API.Chat;
03 using OpenAI_API.Models;
04 using Newtonsoft.Json;
05 using Newtonsoft.Json.Linq;
06
07 namespace ChatGPT01
08 {
```

```
09    public partial class Form1 : Form
10    {
11        public Form1()
12        {
13            InitializeComponent();
14        }
15
16        private async void button1_Click(object sender, EventArgs e)
17        {
18            // 從 textBox1 中取得使用者輸入的提示
19            string prompt = textBox1.Text;
20
21            // 判斷提示是否為空，若是則顯示「提示不能空白」並離開事件
22            if (string.IsNullOrEmpty(prompt))
23            {
24                MessageBox.Show("提示不能空白");
25                return;
26            }
27
28            try
29            {
30                //  使用 OpenAI API 金鑰建立 OpenAIAPI 物件 api
31                OpenAIAPI api = new OpenAIAPI("OpenAI API 金鑰");
32
33          // 透過 OpenAIAPI 物件執行對話(Chat)操作並取得回應結果 CharResult 物件
34                ChatResult chatResult = await
                      api.Chat.CreateChatCompletionAsync(new ChatRequest()
35                {
36                    Model = Model.ChatGPTTurbo,           //使用模型
37                    Messages = new ChatMessage[]         //指定聊天訊息
38                    {   //指定聊天提示
39                        new ChatMessage(ChatMessageRole.User, prompt)
40                    }
41                });
42
43                // 將回傳結果顯示在 richTextBox1 中
44                var reply = chatResult.Choices[0].Message.Content;
45                richTextBox1.Text = reply;
```

```
46
47      //將回傳結果序列化為 JSON 字串，同時將 JSON 字串格化顯示在 richTextBox2 中
48                  string jsonStr=JsonConvert.SerializeObject(chatResult);
49                  richTextBox2.Text = JObject.Parse(jsonStr).ToString();
50              }
51              catch (Exception ex)
52              {
53                  // 捕捉例外情況，顯示錯誤訊息
54                  MessageBox.Show($"發生錯誤：{ex.Message}");
55              }
56          }
57      }
58  }
```

說明

1. 第 1-5 行：引用相關命名空間。
2. 第 31 行：建立 OpenAIAPI 類別物件 api，同時指定金鑰。關於金鑰請讀者自行申請。
3. 第 48-49 行：將回應結果 CharResult 物件轉成 JSON 字串並顯示於 richTextBox2，下列使用 JSON 說明 CharResult 類別物件對應的屬性。

```
{
  "id": "chatcmpl-8imk3HM3ieW7K3HkyYUf1L43fxqHR",
  "choices": [
    {
      "index": 0,
      "message": {
        "role": "assistant",
        "content": [
          {
            "type": "text",
            "text": "量子力學是一門研究微觀世界的物理學科，它揭示了原子、分子、基本粒子等微觀粒子的行為和性質。.....",
            "image_url": null
          }
        ],
        "name": null
      },
```

回覆內容可由 chatResult.Choices[0].Message.Content 取得

```
      "finish_reason": "stop",
      "delta": null
    }
  ],
  "usage": {
    "completion_tokens": 736,
    "prompt_tokens": 16,
    "total_tokens": 752
  },
  "created": 1705684487,
  "model": {
    "id": "gpt-3.5-turbo-0613",
    "owned_by": null,
    "object": null,
    "created": null,
    "permission": [],
    "root": null,
    "parent": null
  },
  "object": "chat.completion"
}
```


25.4.2 常用聊天參數

OpenAIAPI 物件執行 Chat.CreateChatCompletionAsync() 可指定 ChatResult 聊天結果物件 chatResult，此物件提供 Message、Temperature、PresencePenalty、FrequencyPenalty 以及 MaxTokens 參數，讓生成文本更加豐富。寫法如下：

```
// 透過 OpenAIAPI 物件執行對話(Chat)操作並取得回應結果 CharResult 物件
ChatResult chatResult = await
    oenai 物件.Chat.CreateChatCompletionAsync(new ChatRequest()
 {
    Model = Model.ChatGPTTurbo,       //使用模型
    Messages = new ChatMessage[]      //指定聊天訊息
    {
      new ChatMessage(ChatMessageRole.System, "你是網路行銷小編"),
      new ChatMessage(ChatMessageRole.User, "幫我寫人中之龍 IG 貼文")
    },
```

```
    Temperature =1.0,
    PresencePenalty = 0,
    FrequencyPenalty = 0,
    MaxTokens = 600,
});
```

1. Model：主要輸入參數，指定使用的模型。不同的模型具有不同的能力和特性，可根據應用場景和需求來選擇適合的模型。
2. Message：主要輸入參數，用於指定 ChatMessage 物件，每個物件代表一種角色，分別為「System」、「User」或「Assistant」；System 可指定 ChatGPT 的角色身份；User 代表使用者可進行提示(詢問問題)；Assistant 是 ChatGPT 回覆的結果，用來做回答的歷史紀錄，讓 ChatGPT 進行連續對話。
3. Temperature：預設值為 1，指定介於 0~2 之間的浮點數資料。用來指定生成文字的多樣性。值越小，生成結果越保守，但也可能較為狹隘。值越大，生成的結果越多樣，但可能會失去一些邏輯性。
4. PresencePenalty：預設值為 0，指定介於-2.0~2.0 之間的浮點數資料。用來指定對先前生成的文本中已經出現過的詞語進行懲罰力度，即鼓勵使用未出現詞語以增加生成的多樣性。例如：生成文本有關食物內容頻繁出現為米、雞肉、豬肉等。若指定 PresencePenalty 大於 0，例如指定 0.6 或 0.9，則下次生成食物會傾向選擇不重複詞彙，例如麵、牛肉、羊肉等，進而增加生成文本的多樣性。
5. FrequencyPenalty：預設值為 0，指定介於-2.0~2.0 之間的浮點數資料。用來指定對高頻率詞語出現機率進行懲罰力度，即鼓勵使用低頻率詞語以增加生成的多樣性。例如：生成文本有關食物內容常見為米、雞肉、豬肉等。若指定 FrequencyPenalty 大於 0，例如 1.3 或 1.6，則生成食物可能會出現較不常見的詞彙，例如鮭魚奶油飯、鴨肉羹等，增加生成文本的多樣性。
6. MaxTokens：指定生成文本內容最大 token 數量，可防止模型生成過於冗長或不適合的回應。

範例：ChatGPT02.sln

製作網路行銷聊天機器人小編， 透過提示與調整 MaxTokens、Temperature、PresencePenalty、FrequencyPenalty 參數，生成需要的行銷貼文與行銷策略。