

1.解題說明:

使用者輸入集合個數 n 以及集合，這些集合存入 `se[]` 陣列，

`generateSubsets` 函數負責遞迴生成所有可能的子集合。

遞迴兩個選擇：每次選擇當前的元素加入子集合（並在下一層遞迴中考慮剩下的元素），或者跳過當前元素。

當遞迴到集合的最後一個元素後，使用 `printSubset` 函數列印當前生成的子集合

2.演算法設計與實作:

```
Source.cpp  * X
Project2 (全域範圍) printSubset(char su[], int si)
1 #include <iostream>
2 using namespace std;
3 void printSubset(char su[], int si) { // 函數來列印子集合
4     cout << "{ ";
5     for (int i = 0; i < si; ++i) {
6         cout << su[i] << " ";
7     }
8     cout << "}" << endl;
9 }
10 void generateSubsets(char set[], char su[], int in, int si, int subs) { // 遞迴生成所有子集合
11     if (in == si) {
12         printSubset(su, subs);
13         return;
14     }
15     generateSubsets(set, su, in + 1, si, subs); // 不選擇當前元素
16     su[subs] = set[in];
17     generateSubsets(set, su, in + 1, si, subs + 1); // 選擇當前元素
18 }
19 int main(void) {
20     int n;
21     cout << "請輸入集合的元素個數: ";
22     cin >> n;
23     char se[100], su[100];
24     cout << "請輸入集合的元素: ";
25     for (int i = 0; i < n; ++i) {
26         cin >> se[i];
27     }
28     cout << "所有的子集合如下:" << endl;
29     generateSubsets(se, su, 0, n, 0);
30 }
31
```

3.效能分析:

時間複雜度:

對於一個包含 n 個元素的集合，子集的總數是 2^n 。因為每個元素都有兩種選擇：包含在子集中跟不包含在子集中。

遞迴的深度是 n ，並且每一層都會進行兩次遞迴，因此時間複雜度是 $O(2^n)$ 。

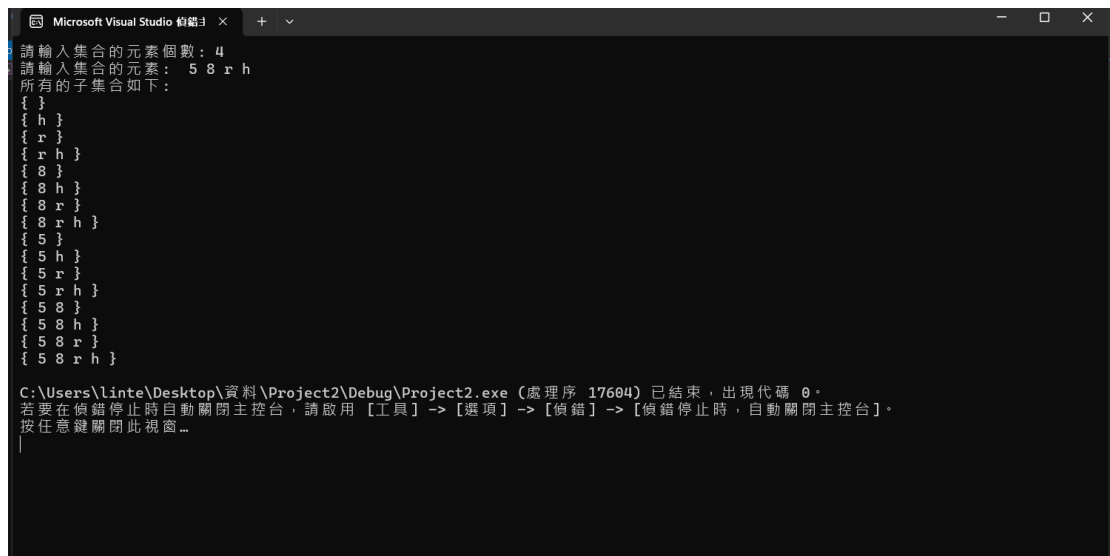
空間複雜度:

`set[]`跟 `su[]`陣列的大小是常數，不會影響空間複雜度，因此其大小為 $O(1)$

空間複雜度：

遞迴的深度最多是 n ，因此遞迴的空間複雜度是 $O(n)$

4.測試與過程:



```
Microsoft Visual Studio 編輯器
請輸入集合的元素個數: 4
請輸入集合的元素: 5 8 r h
所有的子集合如下:
{}
{h}
{r}
{r h}
{8}
{8 h}
{8 r}
{8 r h}
{5}
{5 h}
{5 r}
{5 r h}
{5 8}
{5 8 h}
{5 8 r}
{5 8 r h}

C:\Users\linter\Desktop\資料\Project2\Debug\Project2.exe (處理序 17604) 已結束，出現代碼 0。
若要在偵錯停止時自動關閉主控台，請啟用【工具】->【選項】->【偵錯】->【偵錯停止時，自動關閉主控台】。
按任意鍵關閉此視窗...
```

子集合:

- 0 個: {}
- 1 個: {h}, {r}, {5}, {8}
- 2 個: {5 8}, {5 r}, {5 h}, {8 r}, {8 h}, {r h}
- 3 個: {5 8 r}, {5 r h}, {5 8 h}, {8 r h}
- 4 個: {5 8 r h}