Explanation of Subset Generation Problem

Given a set represented as a string, write a recursive code to print all the subsets of it. The subsets can be printed in any order.

Examples:

```
Input:  set = "abc"
Output: "". "a", "b", "c", "ab", "ac", "bc", "abc"

Input: set = "abcd"
Output: "" "a" "ab" "abc" "abcd" "abd" "ac" "acd"
         "ad" "b" "bc" "bcd" "bd" "c" "cd" "d"
```

The idea is to consider two cases for every character. (i) Consider current character as part of the current subset (ii) Do not consider current character as part of the current subset.

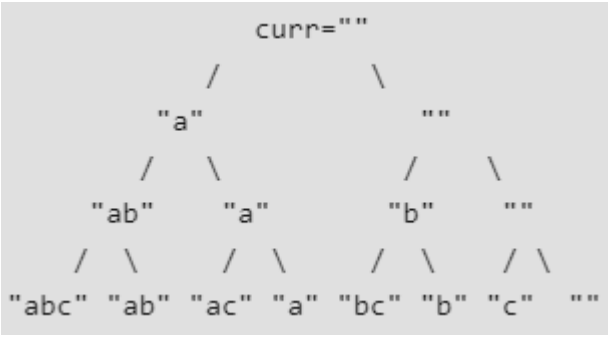
```
// CPP program to generate power set
#include <bits/stdc++.h>
using namespace std;

// str : Stores input string
// curr : Stores current subset
// index : Index in current subset, curr
void powerSet(string str, int index = 0,string curr = "")
{
    int n = str.length();
    // base case
    if (index == n) {
        cout << curr << endl;
        return;
    }

    // Two cases for every character
    // (i) We consider the character
```

```
    // as part of current subset
    // (ii) We do not consider current
    // character as part of current
    // subset
    powerSet(str, index + 1, curr + str[index]);
    powerSet(str, index + 1, curr);
}
// Driver code
int main()
{
    string str = "abc";
    powerSet(str);
    return 0;
}
```

Output:

```
abc
ab
ac
a
bc
b
c
```

Let us understand the recursion with an example "abc". Every node in the below tree represents the string **curr**.



At root, index = 0.

At next level of tree index = 1

At third level, index = 2

At fourth level index = 3 (becomes equal to string length), so we print the subset.