Lesson:

Problems based on Recursion - 8

PW SKILLS

Pre-Requisites

- Recursion basics
- Working rules of recursive functions

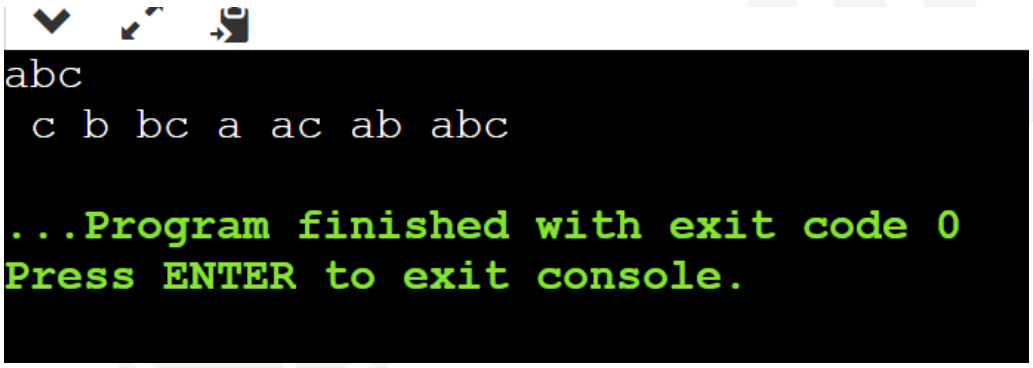
List of Concepts Involved:

- Given a string, return an Arraylist of all its subsequences.
- Given a string, print all its subsequences of it.
- Given an array of integers, print sums of all subsets in it.

Problem 1 : Given a string, we have to return an ArrayList of all its subsequences of it. A String is a subsequence of a given String that is generated by deleting some character of a given string without changing its order.
Examples:

Input : abc
Output : a, b, c, ab, bc, ac, abc.

Coding implementation: https://pastebin.com/ZQmsyjpf



Explanation:

- Subsequence: A subsequence is any part (continuous or non-continuous) of the string. The relative order of the elements must be maintained.
- For example if the string is : “abc” then all possible subsets of this array will be:
  “” , “a” , “b” , “c” , “ab” , “ac” , “cb” , “abc” these are possible subsets. Here relative order should be maintained, that means if ‘a’ is occurring before ‘b’ in the original string then in whatever subsequence we consider, ‘a’ should always appear before ‘b’.
- The function “subsequences” takes 4 parameters namely the original string, the current index , an arraylist of strings that will hold all the possible subsequences so formed and an output string that will hold the characters of any particular subsequence in one go.
- To form a subsequence or for any element we have two choices, either to include the current element in the subsequence or exclude it. For example in the subsequence “ac” we choose to include a and c but we excluded b.
- We make sure that if any element is being included in the current subset then its value must be added to the output string.

- The base case/terminating condition would be if the current index idx has already reached the last index we can simply put the output string in the result arraylist. Since we have visited one complete possible subsequence, in which multiple cases can be there. Maybe we have not taken/ selected even a single element in our subsequence or we have taken all the elements from the original string in the subsequence or maybe we have taken some elements and rejected some other elements.
- Different recursive calls will traverse/cover all such cases.
- If the base case is not hit, it means our idx (index) is in the string currently and for this particular element we have a choice of including as well excluding it.
- We first go with the case of excluding this value from our subsequence.
- So we will add nothing in the output string and will move to the next call with parameters s,idx+1,ans,res.
- The second choice could be to include this element in the subsequence. In order to do that we will simply add the current indexed character in the output string and will pass on to the next call with parameters s,idx+1,ans,res+s.charAt(idx).
- Doing both the calls as listed above we will generate all possible subsequences.

Problem 2 : Given a string, we have to print out all its subsequences of it. A String is a subsequence of a given String that is generated by deleting some character of a given string without changing its order.
Examples:

Input : abc
Output : a, b, c, ab, bc, ac, abc.

Coding implementation: https://pastebin.com/quVx6QKr

```
abc
 c b bc a ac ab abc

...Program finished with exit code 0
Press ENTER to exit console.
```

Explanation:

- Subsequence: A subsequence is any part (continuous or non-continuous) of the string. The relative order of the elements must be maintained.
- For example if the string is : “abc” then all possible subsets of this array will be:
  “” , “a” , “b” , “c” , “ab” , “ac” , “cb” , “abc” these are possible subsets. Here relative order should be maintained, that means if ‘a’ is occurring before ‘b’ in the original string then in whatever subsequence we consider, ‘a’ should always appear before ‘b’.
- The function “subsequences” takes 2 parameters namely the string, the currAns string that stores the answer from the already traversed string.
- To form a subsequence or for any element we have two choices, either to include the current element in the subsequence or exclude it. For example in the subsequence “ac” we choose to include a and c but we excluded b.

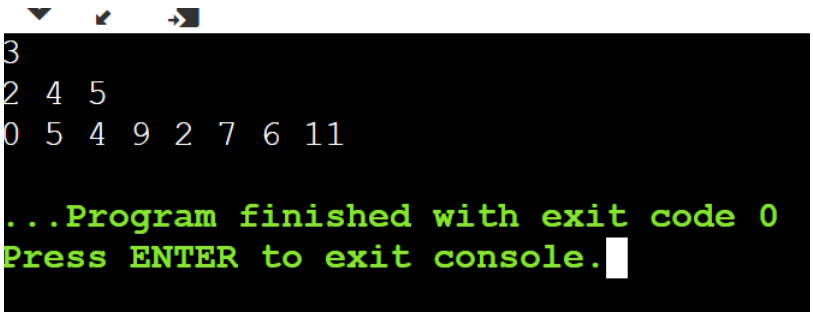
- We make sure that if any element is being included in the current subset then its value must be added to the currAns string.
- The base case/terminating condition would be if the string is empty then we have evaluated the whole string so we will simply print our currAns. Since we have visited one complete possible subsequence, in which multiple cases can be there. Maybe we have not taken/ selected even a single element in our subsequence or we have taken all the elements from the original string in the subsequence or maybe we have taken some elements and rejected some other elements.
- Different recursive calls will traverse/cover all such cases.
- If the base case is not hit, it means we have a string left for evaluating and for this particular element we have a choice of including as well excluding it.
- We first go with the case of excluding this value from our subsequence.
- So we will add nothing in the currAns and will move to the next call with parameters s.substring(1), currAns.
- The second choice could be to include this element in the subsequence. In order to do that we will simply add the current indexed character in the CurrAns and will pass on to the next call with parameters s.substring(1), currAns+ curr (current character)
- Doing both the calls as listed above we will generate all possible subsequences.

Problem 3 : Given an array of integers, print sums of all subsets in it. Output sums can be printed in any order.

Input : arr[] = {2, 3}
Output: 0 2 3 5

Input : arr[] = {2, 4, 5}
Output : 0 2 4 5 6 7 9 11
Ans 8.

Coding implementation: https://pastebin.com/kGbGqv2X



Explanation:

- In the SumSubset function we have passed 5 parameters namely the array, length of the array, current index , a result vector that will hold all the sum of different subsets obtained, and a sum value that will hold the temporary sum for one particular subset.
- Subset: A subset is any part (continuous or non-continuous) of the array. The relative order of the elements must be maintained.
- For example if the array is : {1,2,3} then all possible subsets of this array will be:
- {} , {1} , {2} , {3} , {1,2} , {1,3} , {2,3} , {1,2,3} these are possible subsets. Here relative order should be maintained, that means if 1 is occurring before 2 in the original array then in whatever subset we consider, 1 should always appear before 2.
- For the above example where the array given is {1,2,3} the answer then would be {0, 1, 2, 3, 3, 4, 5, 6}. Any value in this set is obtained by summing up values of elements present inside one subset.
- Here the first task or baby step would be to try out how to create subsets and then summing them up one by one.
- To form a subset or for any element we have two choices, either to include the current element in the subset or exclude it. For example in the subset {1,3} we choose to include 1 and 3 but we excluded 2.
- We make sure that if any element is being included in the current subset then its value must be added to the variable 'sum'.
- The base case/terminating condition would be if the current index idx has already reached the last index we can simply put the sum value in the result array. Since we have summed up one complete possible subset, in which multiple cases can be there. May be we have not taken/ selected even a single element in our subset or we have taken all the elements from the original array in the subset or maybe we have taken some elements and rejected some other elements.
- Different recursive calls will traverse/cover all such cases.
- If the base case is not hit, it means our idx (index) is in the array currently and for this particular element we have a choice of including as well excluding it.
- We first go with the case of excluding this value from our subset.
- If we exclude the current value then it makes no sense in changing the sum, sum should remain whatever it was earlier because the current element is not added.
- So we passed the parameters a,n,idx+1,res,sum for the future calls, this clearly indicates that the current element with index idx has not been included in the current subset that we are considering.
- Another option could be to include this current element in the subset.
- For that the parameters for the next call would be a,n,idx+1,res,sum+a[idx]. Here the value of sum has been incremented by the current element's value since we are including the current element in the subset. Hence the sum has been updated.
- Doing both the calls as listed above we will generate all possible subset's sum.

For the beginner level it is advisable to first learn how to form and print every subset and then move to the summation part.

Upcoming Class Teasers:

- Problems based on recursion.