Difference between Subarray, Subsequence and Subset

In this post, we will discuss the difference between a subarray/substring, a subsequence and a subset.

1. Subarray

A subarray is a slice from the array which is contiguous (i.e. occupy consecutive positions) and inherently maintains the order of elements. For example, the subarrays of the array `{1, 2, 3}` are `{1}, {1, 2}, {1, 2, 3}, {2}, {2, 3}, and {3}`.

Below is a C program to generate all subarrays of the specified array:

```
#include <stdio.h>

// Function to print a subarray formed by [start, end]
void printSubarray(int arr[], int start, int end)
{
    printf("{");

    for (int i = start; i < end; i++)
        printf("%d, ", arr[i]);

    printf("%d}, ", arr[end]);
}

// Function to print all subarrays of the specified array
void printallSubarrays(int arr[], int n)
{
    // consider all subarrays starting from i
    for (int i = 0; i < n; i++)
    {
        // consider all subarrays ending at j
        for (int j = i; j < n; j++)
            printSubarray(arr, i, j);
    }
}

// Program to print all subarrays of the specified array
int main()
{
    int arr[] = { 1, 2, 3, 4, 5 };
    int n = sizeof(arr)/sizeof(arr[0]);

    printallSubarrays(arr, n);

    return 0;
}
```

Download Run Code

Output:

```
{1}, {1, 2}, {1, 2, 3}, {1, 2, 3, 4}, {1, 2, 3, 4, 5}, {2}, {2, 3}, {2, 3, 4}, {2, 3, 4, 5}, {3}, {3, 4}, {3, 4, 5}, {4}, {4, 5}, {5}
```

Please note that there are exactly `n*(n + 1)/2` subarrays in an array of size n. Also, there is no such thing as contiguous subarray. The prefix contiguous is sometimes applied to make context more clear. So a contiguous subarray is just a another name for a subarray.

2. Substring

A substring of a string `S` is a string `S'` that occurs in `S`. A substring is almost similar to a subarray but it is in context of strings.

For example, the substrings of the string `'apple'` are `'apple', 'appl', 'pple', 'app', 'ppl', 'ple', 'ap', 'pp', 'pl', 'le', 'a', 'p', 'l', 'e', and ''`. Below is a C++ program that generates all non-empty substrings of the specified string:

```
#include <iostream>
using namespace std;

// Function to print all non-empty substrings of the specified string
void printallSubstrings(string str)
{
    int n = str.length();

    // consider all substrings starting from i
    for (int i = 0; i < n; i++)
    {
        // consider all substrings ending at j
        for (int j = i; j < n; j++)
        {
            cout << "'" << str.substr(i, j + 1) << "', ";
        }
    }
}

// Program to print all non-empty substrings of the specified string
int main()
{
    string str = "techie";
    printallSubstrings(str);

    return 0;
}
```

Download Run Code

Output:

```
't', 'te', 'tec', 'tech', 'techi', 'techie', 'ec', 'ech', 'echi', 'echie', 'echie', 'chi', 'chie', 'chie', 'chie', 'hie', 'hie', 'hie', 'ie', 'ie', 'e'
```

3. Subsequence

A subsequence is a sequence that can be derived from another sequence by deleting some elements without changing the order of the remaining elements. For example, `{A, B, D}` is a subsequence of sequence `{A, B, C, D, E}` that is obtained after removing `{C}` and `{E}`.

People are often confused between a subarray and a subsequence. A subarray will always be contiguous but a subsequence need not be contiguous. That is, unlike subarrays, subsequences are not required to occupy consecutive positions within the original sequences. But we can say that both contiguous subsequence and subarray are same.

In other words, the subsequence is a generalization of substring or substring is a refinement of the subsequence. For example, `{A, C, E}` is a subsequence of `{A, B, C, D, E}`, but not a substring and `{A, B, C}` is both a subarray and a subsequence.

Please note that a Subsequence can be in context of both arrays and strings. Generating all subsequences of an array/string is equivalent to generating power set of the array/string. For a given set S, the power set can be found by generating all binary numbers between 0 to 2^n-1 where n is the size of the given set. This is demonstrated below:

```
#include <iostream>
#include <cmath>
using namespace std;

// Function to print all subsequences of the specified string
void findPowerSet(string str)
{
    int n = str.length();

    // N stores total number of subsets
    int N = pow(2, n);

    // generate each subset one by one
    for (int i = 0; i < N; i++)
    {
        cout << "'";

        // check every bit of i
        for (int j = 0; j < n; j++)
        {
            // if j'th bit of i is set, print S[j]
            if ((i & (1 << j)))
                cout << str[j];
        }
        cout << "', ";
    }
}

// Program to print all subsequences of the specified string
int main()
{
    string str = "apple";
    findPowerSet(str);
}
```

Download Run Code

Output:

```
'', 'a', 'p', 'ap', 'p', 'ap', 'pp', 'app', 'l', 'al', 'pl', 'apl', 'pl', 'apl', 'ppl', 'appl', 'e', 'ae', 'pe', 'ape', 'pe', 'ape', 'ppe', 'appe', 'le', 'ale', 'ple', 'aple', 'ple', 'aple', 'pple', 'apple'
```

4. Subset

A subset is any possible combination of the original set. The term subset is often used for subsequence but this is wrong. A subsequence always maintain the relative order of elements of the array (i.e. increasing index) but there is no such restriction on a subset. For example, `{3, 1}` is a valid subset of `{1, 2, 3, 4, 5}` but it is neither a subsequence or a subarray.

It is worth noting that all subarrays are subsequences and all subsequences are subset but the reverse is not true. For instance, the subarray `{1, 2}` of the array `{1, 2, 3, 4, 5}` is also a subsequence and a subset.

Thanks for reading.

Please use Ideone or C++ Shell or any other online compiler link to post code in comments.
Like us? Please spread the word and help us grow. Happy coding 😉

Array, Basic, String
Algorithm, Must Know

← Greedy coloring of graph
Job Sequencing Problem with Deadlines →