

Recursive Algorithm

Recursive algorithm is a kind of method which calls itself directly or indirectly the corresponding is called as recursive algorithm. It make a small value of input value. It use to solve same problem with same solution. It is a kind of repeat process for the single process. It can use for a problem which can't solve with looping. Performance of recursive is slow because it called itself. Recursive algorithm is use in the complex process problem and it is easy to understand but the performance is not good enough. To use recursive it is very careful to use although it is easy to understand. Because if it wrong usage, it can happen memory stark over flow. Or if the base case is wrong it can have problem with the recursive algorithm.

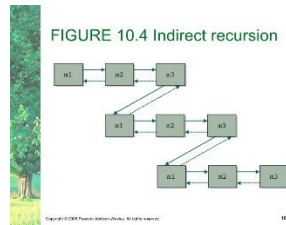
In recursive algorithm it has direct recursive and indirect recursive which called itself directly or go through other methods. Direct recursion is a method which invokes itself and direct recursive is the expression is same in the first paragraph. So here is an example of direct recursive algorithm.

```
public static void main(String[] args) {  
    // TODO code application logic here  
    printHello(5);  
}  
  
public static void printHello(int count)  
{  
    if (count >= 1)  
    {  
        System.out.println("Hello");  
        printHello(count-1);  
    }  
}
```

In photo, you will see printHello method with parameter. If the count is less than equal to 1 it will print "Hello". You will see the count-1 in it's method. It will reduce count until reach 1 which have the total count in main method. In printHello method. It call itself by reducing count. So it is a directly call is called direct recursive algorithm.

Indirect recursive opposite of direct recursive algorithm. Indirect recursion is when method called form a different method and its turn called the original method. It use more than one method. Indirect it only use one method itself. Indirect recursive called more than one method and use recursively.

A good example is directory traversal program. One method for navigating the hierarchy and one for processing the files. If the file turns out to be a directory then the original process folder method is called and so on.



When use Recursive Algorithm?

It use when the process is huge or big and to see the code clearly and not to have much tracking trace for the program. Like quick sort, merge sort. In quick sort, you have to make partition, swap process, and check condition process. So if you make one layer code, you will find too much nervous and you will have difficult in tracing. So generally recursive algorithm is used for complex and clear process. So to clear quick sort and easy to trace recursive algorithm can use in quick sort. Not only quick. It is up to the condition of the program. If you want to have clear and to solve complex condition. You should use quick sorting. But recursive code doesn't use in every condition. If a program need to have clear to understand and complex condition, recursive algorithm can use.

So here is an example with quick sort. You can have code in one layer of quick sort. The main difference is too difficult track and very complex code and it is difficult to understand. First we have to make a partition of quick sort array and get pivot. So you will have a partition method for process. After partition you will get pivot and make quick sort. In this case I want to show how recursive work. So we will do quick sort for the recursively.

```

void quickSort(int arr[], int low, int high)
{
    if (low < high)
    {
        /* pi is partitioning index, arr[p] is now
           at right place */
        int pi = partition(arr, low, high);

        // Separately sort elements before
        // partition and after partition
        quickSort(arr, low, pi - 1);
        quickSort(arr, pi + 1, high);
    }
}

```

Recursive use parameter. So you will see a quick sort method. In this case you will see the calling method of quick sort. Parameter data in the quick sort method are recursively called. And data will be transmitted to itself. If other method called that method. Parameter data will be inserted. So in this case, you will clear and easy to find tracking.

How to use Recursive Algorithm?

As I says recursive algorithm is use to solve for complex condition which can't solve with looping. So it is use in complex case. So how to use it? Called it's method in it's scope. It is a little bit complex. Before you

use recursive algorithm you should think about the data structure with steps by steps by using pseudo code or flow chat by thinking carefully and tracing each steps. Because if you make wrong recursive, it can have memory stack overflow and that problem is doesn't same with other program problems, it can harm to your device or memory. To use recursive, you should have to think about the parameter, base state which make to stop the sequence or loop and call it's method to do again and again. If you are wrong in base case, you will face with memory over flow or which relate with that problems. Think about the condition steps by steps to prevent from memory stack problem.

Why use recursive Algorithm?

As I says, it is to solve for the complex problems. Recursion can typically allow to use with other concepts such as immutability. Mostly for the looping which is used to be altering a mutable data structure. Using immutable data structure can allow for things like simple parallelizing of your functions. So you can distribute the work among several CPU's without running the risk of corrupting your tasteful object. If you use recursive the performance is not good enough because of the memory over problems. It just need to use for the complex process for tracking easy for the programmer.

Example Usage of Recursive Algorithm

So as an example, mostly use in algorithm process. It can be in hacking, hashing, encryption. Because they do repeat and repeat same process for all. In hacking to lock the hashing code they should use recursive algorithm to apply. So sometimes hacking apps are memory over problem for hackers. Because simply base case is wrong for hacking algorithm. Hacker can't extract and doesn't know about the Company A's hashing. So they use that kind of recursive algorithm to think about the hashing design of the Company A. And to solve heavy problems to subs problems.

Recursive in Merge sort

We already know that merge sort is divide until the room doesn't reach to the last. And use selecting sort to make merge sort. So in merge it will have many steps if you write code in one layer, you will have difficult to trace of marge sort. So you have to use different methods and recursive algorithm to apply merge sort. So here is an example.

```
void mergeSort(int arr[], int l, int r)
{
    if (l < r)
    {
        // Same as (l+r)/2, but avoids overflow for
        // large l and h
        int m = l+(r-l)/2;

        // Sort first and second halves
        mergeSort(arr, l, m);
        mergeSort(arr, m+1, r);


        merge(arr, l, m, r);
    }
}
```

So you will see mergesort method and it's called it's self with parameter. That parameter form he main method and form partition method to put it. Those data will transfer to the partition method to make

condition of the sorting. With the parameter, partition will take out and sort method will do sort in main method.

Recursive in Binary Search

Binary also need to apply recursive algorithm. Binary is divided and make condition to get the data. It will have many steps to finish and it seem easy in outside. But with the code it can complex for the programmer. So recursive algorithm can help programmer to trace easily.



```
int binarySearch(int arr[], int l, int r, int x)
{
    if (r >= l)
    {
        int mid = l + (r - l)/2;

        // If the element is present at the middle
        // itself
        if (arr[mid] == x)
            return mid;

        // If element is smaller than mid, then
        // it can only be present in left subarray
        if (arr[mid] > x)
            return binarySearch(arr, l, mid-1, x);

        // Else the element can only be present
        // in right subarray
        return binarySearch(arr, mid+1, r, x);
    }

    // We reach here when element is not
    // present in array
    return -1;
}
```

You will see binarySearch method and it's called itself. it find middle number from left and right. Those left and right will get form the parameter. And parameter data will come from main method. This method is also called in main method to get data and make sort. And it returns it's method and -1. If -1 it can't find. Search data doesn't have in the array list. If its returns it will make binary search up to the condition.

In those case, in merge sort and binary search. It can easily trace all the program. It doesn't have any performance for the program. If the parameter is wrong, it can only have memory stack over flow problems. So think careful for the base to prevent from problems. I hope you will understand about recursive algorithm.