Java Recursive Cheat Sheet

by Jianmin Feng (taotao) via cheatography.com/79308/cs/19302/

Recursive method

a process is defined in terms of a simplier case of itself

each time local variable created and push into memory stack

when termination triggered, program return and completer most recent invocation (FILO) therefore reversed from call sequence.

Recursive could be very inefficient; it is for simplicity and beauty of code/thinking, not for efficient.

Structure

```
public void recursiveMethod(
...) {...
   if (base case) { ... }
//termination condition
   else { //non base case move
algorithm to base
      ...
      recursiveMethod(...);
//recursive call
      ...
   }
}
```

a tail recursive: there is no statement after recursive call.

Recursive vs Iterative

Every recursive algorithm can be written iteratively which is more efficient in time and memory.

Recursive for elegance and simplicity of code

Rule for recursion

1 Avaoid when involving large local array(memory overflow)

2 avoid for simple iterative method like factorial,fibonacci and linear search.

3 use recursive when significantly simplify code

Recursive vs Iterative (cont)

4 use recursive for divide and conquer problem(merge /binary sort) and branching process like traversing trees or directories

Recursive help methods

Private recursive method + public non recursive driver method (helper method)

Hide implementation of recursive from user + enhance efficiency by putting precon-ditions in helper method instead of in recursive method

Driver method example1

```
//recursive method
private static int sum(int n){
    if (n==1) return 1;
    else return n +sum(n-1);
}
//Drive method
public static int getSum(int n)
{
    if (n>0) return sum(n);
    else throw new IllegalArgum-
entException
                  ("Error: n
must be positive");
}
```

Driver method example 2

```
//recursive method
private boolean recurSearch
    (String[] a,int startIdx,-
String Key){
    if (startIdx =a.length)
return false;
    else if ( a[startIdx].com-
pareTo(key)==0)
          return true;
    else return recurSearch(a,s-
tartIdx,key);
}
//driver method
public boolean search(
    String[] a, String key)
{ return recurSearch(a,0,key);}
```

Recursion in 2 dimensional grid

```
check the starting position is
not out of ranger
    if condition met,
        perform some action to
solve problem
        recursiveCall(row+1,col)
        recursiveCall(row-1,col)
        recursiveCall(row,
col+1)
        recursiveCall(row, col-
1)
```

Practice ...

Practice more until the recursive thinking become a second nature of your mind. It is beauty and simplicity of problem solving.

Untangle the recursive call by Sketching

untangle by repeating call

untangle by box diagrams

By **Jianmin Feng** (taotao)
cheatography.com/taotao/

Not published yet.
Last updated 9th April, 2019.