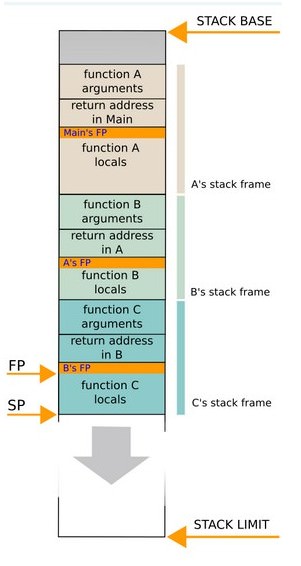
Why is the Stack so important to implement recursion? When a recursive call is made what all is pushed on the stack? What will happen if there is no exit condition in a recursive function?

1. "Recursion" is technique of solving any problem by calling same function again and again until some breaking (base) condition where recursion stops and it starts calculating the solution from there on. For eg. calculating factorial of a given number
2. Thus in recursion last function called needs to be completed first.
3. Now Stack is a LIFO data structure i.e. ( Last In First Out) and hence it is used to implement recursion.
4. The High level Programming languages, such as Pascal , C etc. that provides support for recursion use stack for book keeping.
5. In each recursive call, there is need to save the
   1. current values of parameters,
   2. local variables and
   3. The return address (the address where the control has to return from the call).
6. Also, as a function calls to another function, first its arguments, then the return address and finally space for local variables is pushed onto the stack.
7. Recursion is extremely useful and extensively used because many problems are elegantly specified or solved in a recursive way.
8. The example of recursion as an application of stack is keeping books inside the drawer and the removing each book recursively.
9. Incase base condition or exit condition is not specified in the function then recursive calls to the function can lead to an infinite loop.

There is a funciton that returns a 16 character string. It needs to called between 5K-10K times and the resutant strings need to be concatted. Which is the best way to do this in java? Why?