Recursion

SDC OSW 3541

Department of Computer Science & Engineering
ITER, Siksha 'O' Anusandhan Deemed To Be University
Jagamohan Nagar, Jagamara, Bhubaneswar, Odisha - 751030

Book(s)

Text Book(s)

Jeri R Hanly, & Elliot B. Koffman

Problem Solving & Program Design in C

Seventh Edition

Pearson Education

Contents

Introduction

Definition:- when a function calls itself directly or indirectly its known as recursion.

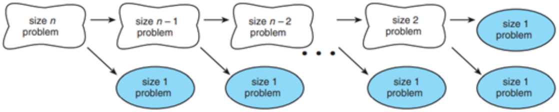


Figure 1: Splitting Problem into Smaller Problems

The given problem is divided into various sub-problems, the consecutive sub problems are solved to obtain the solution of the given problem.

Introduction (Cont..)

Two most essential elements of the recursion are base case and recursive statement.

Base case is a case where terminating condition becomes true, terminating condition is a condition that indicates the termination of the recursive function and prevent the recursion from the infinite loop.

Recursive statement is a statement that holds the call to corresponding function, which will get called recursively every time function gets executed.

Types of Recursion

There's mainly two type of recursions.

Direct recursion:

If the recursive statement is in function itself then it's known as direct recursion call.

Indirect recursion:

If the recursive statement is in another function or we can say if the call is made by another function its known as indirect recursion call.

Types of Recursion (Cont..)

```
//Direct Recursion Example:
#include<stdio.h>
int add(int n){
        int ans;
if (n==1)
return n;
     else{
        ans=n+add(n-1);
        return ans;
         }
}

void main(){
        printf("%d", add(5));
}
```

Types of Recursion (Cont..)

```
//Indirect Recursion Example

#include <stdio.h>
void odd();
void even();
int num = 1;
void odd ()
{
    if (num <= 10)
    {
        printf (" %d ", num
            + 1);
        num++;
        even();
    }
    return;
}
```

```
//continued
 void even ()
{
    if ( num <= 10)
    {
        printf (" %d ", num -
            1);
        num++;
        odd();
    }
    return;
}
int main ()
{
    odd();
    return 0;
}
```

Tracing of Recursion

```
#include<stdio.h>
int multiply(int m, int n)
{
int ans;
if (n == 1)
ans = m; /* base case */
else
ans = m + multiply(m, n - 1); /* recursive statement */
return (ans);
 }
 void main(){
        int c;
        c=multiply(6,3);
        printf("ANSWER %d", c);
 }
```

Tracing of Recursion (Cont.)

Illustration of splitting problem into smaller problems:

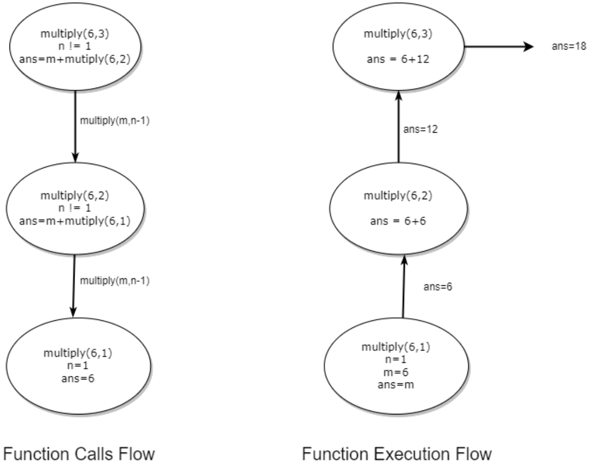


Figure 2: Recursion tracing using Splitting Problem

Tracing of Recursion (Cont.)

Illustration using activation frame:

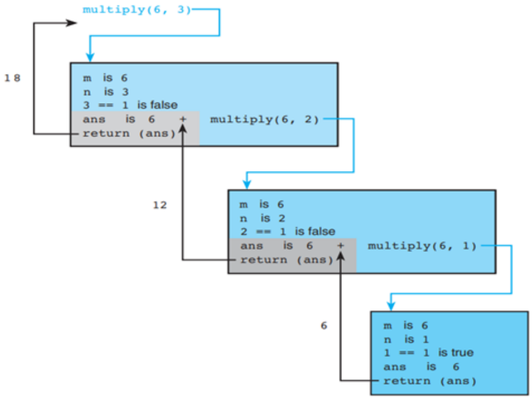


Figure 3: Recursion tracing using Activation Frame

Tracing of Recursion (Cont.)

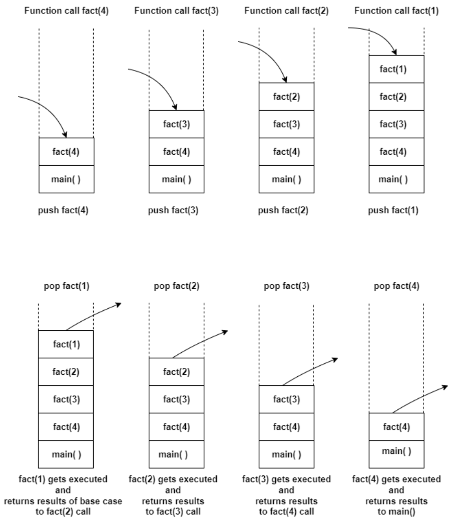

Function call fact(4)
Function call fact(3)
Function call fact(2)
Function call fact(1)
fact(4)
main( )
fact(3)
fact(2)
fact(1)
push fact(4)
push fact(3)
push fact(2)
push fact(1)
pop fact(1)
pop fact(2)
pop fact(3)
pop fact(4)
fact(1) gets executed and returns results of base case to fact(2) call
fact(2) gets executed and returns results to fact(3) call
fact(3) gets executed and returns results to fact(4) call
fact(4) gets executed and returns results to main()

Figure 4: Recursion tracing using Stacks

Difference between Recursion and Iteration

Recursion:	**Iteration:**
- When a function calls itself directly or indirectly.	- When some set of instructions are executed repeatedly.
- Implemented using function calls.	- Implemented using loops.
- Base case and recursive relation are specified.	- Includes initializing control variable, termination condition, and update of the control variable.
- The function state approaches the base case.	- The control variable approaches the termination value.
- Uses stack memory to store local variables and parameters.	- Does not use memory except initializing control variables.
- It will cause stack overflow error and may crash the system if the base case is not defined or is never reached.	- It will cause an infinite loop if the control variable does not reach the termination value.

Difference between Recursion and Iteration(Cont..)

```
//Factorial using recursion:
#include<stdio.h>
int factorial(int n)
{
int ans;
if (n == 0)
ans = 1;
else
ans = n * factorial(n - 1);
return (ans);
 }
 void main(){
        int fact;
        fact=factorial(3);
        printf("FACTORIAL is
             %d", fact);
 }
```

```
//Factorial using iterative:
#include<stdio.h>
int factorial(int n)
{
int i, product = 1;
for (i = n; i > 1; --i) {
product = product * i;
}
return (product);
}
 void main(){
        int c;
        c=factorial(3);
        printf("FACTORIAL is %d
            ", c);
 }
```

Case Study: Tower of Hanoi

Algorithm:

step1: if n is 1 then

step2: move disk 1 frompeg to topeg

step3: else move n-1 disk frompeg to auxiliarypeg using the topeg.

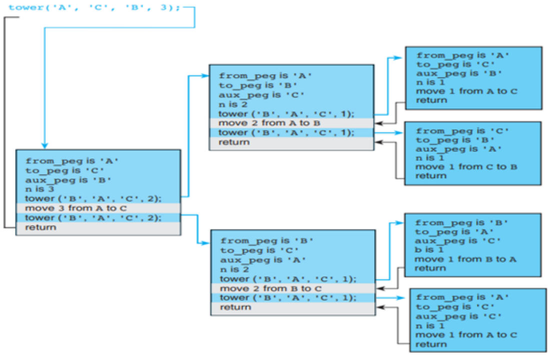
step4: move disk n from the frompeg to the topeg.

step5: move n-1 disks from the auxiliarypeg to the topeg using the frompeg.

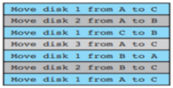
```
//Recursive function:
void tower(char from_peg, char to_peg, char aux_peg, int n)
     {
 if (n == 1) {
printf("Move disk 1 from peg %c to peg %c\n", from_peg,
    to_peg);
} else {
tower(from_peg, aux_peg, to_peg, n - 1);
printf("Move disk %d from peg %c to peg %c\n", n, from_peg,
     to_peg);
tower(aux_peg, to_peg, from_peg, n - 1); }
}
```

Case Study: Tower of Hanoi (Cont.)

Tracing using activation frame:



Output:



```
Move disk 1 from A to C
Move disk 2 from A to B
Move disk 1 from C to B
Move disk 3 from A to C
Move disk 1 from B to A
Move disk 2 from B to C
Move disk 1 from A to C
```

Figure 5: Tracing using Activation Frame

Common Programming Errors

- If recursive statement has no base case, recursion may result into infinite loop of function calls till the whole memory is used.

- A run time error message noting stack overflow or an action violation is an indicator that a recursive function is not terminating.

- The recopying of large arrays or other data structures can quickly consume all available memory. Such copying should be done inside a recursive function only.

- On most systems, pressing a control character sequence (e.g., Control S) will temporarily stop output to the screen.

Sample Questions

Questions

1) Write a program for counting the occurrences of a character in a string.

2) Write a program to find capital letters in a string using recursive function?

3) Write a program to find Fibonacci series using recursive function?

4) Write a program for recursive selection sort?

5) Write a program to find GCD of two integers using recursive function?

THANK YOU