# L9 - Recursion II

Thursday, April 2, 2020 10:43 PM

**Recursion**

Recursive Formulation/Recursive Decomposition

To conceive/decompose a problem in terms of simpler problem (**of the same nature as that of the original problem**) and some trivial activities.

In other words, in the solution to the problem, the same problem re-occurs, many times, on a smaller size (simpler problem(s)) until the result is achieved.

***Single Basis***

## Factorial

$$Fact(n) = \begin{cases} 1 \ ; & if\ n = 1 \quad Basis\ step \\ n * Fact(n-1)\ ; & if\ n > 1 \quad Recursive\ step \end{cases}$$

## Sum of n numbers

$$Sum(n) = \begin{cases} 0 \ ; & if\ n = 0; \quad Basis\ step \\ Sum(n-1) + \ last\ element\ ; & if\ n > 0; \quad Recursive\ step \end{cases}$$

**Print a list of Elements**

Display the list of elements/numbers in reverse order

$$print(n) = \begin{cases} display\ n^{th}\ (first) element; & if\ n = 1; \quad Basis\ Step \\ display\ n^{th}\ (last) element\ and\ then\ print(n-1); & if\ n > 1\ ; \quad Recursive\ Step \end{cases}$$

**Print a list of Elements**

Display the list of elements/numbers in reverse order

$$print(n) = \begin{cases} do\ nothing\ (finish); & if\ n = 0 \\ display\ n^{th}\ (\text{last})\ element\ and\ then\ print(n-1); & if\ n > 0 \end{cases};$$

```
Print(Arr, n)
1. If n == 1
2.     display Arr[n]
3.     return
4. else if n > 1
5.     display Arr[n]
6.     Print(Arr,n-1)
```

```
Print2(Arr, n)
1. If n == 0
2.     return
3. display Arr[n]
4. Print2(Arr,n-1)
```

Arr: 2 15 7

Print2(Arr,3) -> print2(Arr, 2) -> print2(Arr, 1) -> print2(Arr,0)
<- 7 <- 15 <- 2 <-

***<u>Multiple Basis Steps</u>***

## <u>Fibonacci</u>

$$fib(n) = \begin{cases} 0; & if\ n = 0 \quad Basis\ Step \\ 1; & if\ n = 1 \quad Basis\ Step \\ fib(n-1) + fib(n-2); & if\ n > 1 \quad Recursive\ Step \end{cases}$$

```
Recursive_Fibonacci(n)
1. If n <= 1
2.     return n
3. return Recursive_Fibonacci(n-1) + Recursive_Fibonacci(n-2)
```

```
if n == 0
    return 0
if n == 1
    return 1
```

**Palindrome** ∈, $a$ , $ab$, $aab$, $aaba$, aa, aba,

str: abcdefedcba str[1..11] = str[1..n] first == last

$$pal(str, first, last) = \begin{cases} true; & if\ n \leq 1\ \ (if\ first \geq last) \\ false; & if n > 1\ and\ \ firstElement\ != lastElement \\ pal(str\ , first + 1, last - 1); & if\ n > 1\ and\ \ firstElement == lastElement \end{cases}$$

pal(str, first, last)

1. if first >= last
2. return true
3. if str[first] != str[last]
4. return false
5. return pal(str,first+1,last-1)

Examples:

str: "aaa"

pal("aaa",1,3) -> pal("aaa",2,2)

<- true <- true

str: "abba"

pal("abba",1,4) -> pal("abba",2,3) -> pal("abba",3,2)

<- true <- true <- true

str: "aaaaabaa"

pal("aaaaabaa",1,8) -> pal("aaaaabaa",2,7) -> pal("aaaaabaa",3,6)

<- false <- false <- false