

Recursion

function calling itself

A.C.E Run

R.C.D → calling

A.C.L 3

→ A.C.L → small

Q

[PM I] → 11th

① smaller input ✓
② K → Formula collect ✓

Case 2

$i=N$
 $\sum_{i=1}^N i = 1+2+3+4+5+...+N = \frac{N(N+1)}{2}$

$i=1$
 $\sum_{i=1}^1 i = 1 = \frac{1 \times (1+1)}{2}$ ✓

$i=2$
 $\sum_{i=1}^2 i = 1+2 = \frac{2 \times (2+1)}{2}$ ✓

$i=K$
 $\sum_{i=1}^K i = 1+2+3+...+K = \frac{K(K+1)}{2}$ ✓

Recursion B.P

SP → Recursion
Self + sp

Recursion 4th 12th 5th

$5! = 4! \times 5$

$4! = 3! \times 4$

$3! = 2! \times 3$

$2! = 1! \times 2$

$1! = 1$

Recursion

$N! = (N-1)! \times N$

Calculus

public static void main(String[] args) {
// TODOAuto-generated method stub
int n = 5;
System.out.println(fac(n));
}

public static int fac(int n) {
if (n == 0) {
return 1;
}
int fn = fac(n-1); // sp
return n * fn; // self work + sp
}

Stackoverflow

main

$n=5$
 $f_n = fac(5)$
 $f_n = fac(4) \times 5$
 $f_n = fac(3) \times 4 \times 5$
 $f_n = fac(2) \times 3 \times 4 \times 5$
 $f_n = fac(1) \times 2 \times 3 \times 4 \times 5$
 $f_n = fac(0) \times 1 \times 2 \times 3 \times 4 \times 5$

public static void main(String[] args) {
// TODOAuto-generated method stub
int n = 5;
System.out.println(fac(n));
}

public static int fac(int n) {
if (n == 0) {
return 1;
}
int fn = fac(n-1); // sp
return n * fn; // self work + sp
}

downfall

main

$n=0$
 $f_n = fac(0)$
 $f_n = fac(-1) \times 0$
 $f_n = fac(-2) \times (-1) \times 0$
 $f_n = fac(-3) \times (-2) \times (-1) \times 0$
 $f_n = fac(-4) \times (-3) \times (-2) \times (-1) \times 0$
 $f_n = fac(-5) \times (-4) \times (-3) \times (-2) \times (-1) \times 0$

$a^n \rightarrow [a, n] \times a$

$(n) \times n$

public static void main(String[] args) {
// TODOAuto-generated method stub
int a = 5;
int n = 4;
System.out.println(pow(a, n));
}

public static int pow(int a, int n) {
if (n == 0) {
return 1;
}
int p = pow(a, n-1);
return p * a;
}

downfall

main

$a=5, n=0$
 $p = pow(5, 0)$
 $p = 1$
 $a=5, n=1$
 $p = pow(5, 1)$
 $p = 1 \times 5$
 $a=5, n=2$
 $p = pow(5, 2)$
 $p = 1 \times 5 \times 5$
 $a=5, n=3$
 $p = pow(5, 3)$
 $p = 1 \times 5 \times 5 \times 5$
 $a=5, n=4$
 $p = pow(5, 4)$
 $p = 1 \times 5 \times 5 \times 5 \times 5$

$N=5$

$\begin{bmatrix} 5 \\ 4 \\ 3 \\ 2 \\ 1 \end{bmatrix}$

public static void main(String[] args) {
// TODOAuto-generated method stub
int n = 5;
PD(n);
}

public static void PD(int n) {
if (n == 0) {
return;
}
System.out.println(n);
PD(n-1);
}

downfall

main

$n=1$
 $PD(1)$
 $n=2$
 $PD(2)$
 $n=3$
 $PD(3)$
 $n=4$
 $PD(4)$
 $n=5$
 $PD(5)$

public class Print_Inc {
public static void main(String[] args) {
// TODOAuto-generated method stub
int n = 5;
PI(n);
}

public static void PI(int n) {
if (n == 0) {
return;
}
PI(n-1);
System.out.println(n);
return;
}

main

$n=0$
 $PI(0)$
 $n=1$
 $PI(1)$
 $n=2$
 $PI(2)$
 $n=3$
 $PI(3)$
 $n=4$
 $PI(4)$
 $n=5$
 $PI(5)$

public static void main(String[] args) {
// TODOAuto-generated method stub
int[] arr = {1, 2, 5, 4, 3, 4, 7, 4, 3, 6};
int item = 4;
System.out.println(Index(arr, item, 0));
}

public static int Index(int[] arr, int item, int idx) {
if (arr[idx] == item) {
return idx;
}
return Index(arr, item, idx+1);
}

main

$idx=0$
 $Index(2, 4, 2)$
 $Index(3, 4, 3)$
 $Index(4, 4, 4)$
 $Index(5, 4, 5)$
 $Index(6, 4, 6)$
 $Index(7, 4, 7)$
 $Index(8, 4, 8)$
 $Index(9, 4, 9)$

Recursion

Statement execute

Head Recursion

tail → Recursion

Fact using tail

public static void main(String[] args) {
// TODOAuto-generated method stub
int n = 5;
System.out.println(fac(n, 1));
}

public static int fac(int n, int ans) {
return fac(n-1, ans*n);
}

main

$n=5$
 $fac(5, 1)$
 $fac(4, 5)$
 $fac(3, 20)$
 $fac(2, 60)$
 $fac(1, 120)$
 $fac(0, 120)$