Unidad 2: Programación en assembler de ARM

Subrutinas. Uso del stack

Estefanía Pereyra[1]

[1] Universidad Tecnológica Nacional-FRC
Técnicas Digitales II

- Función llamadora:
  - Pasa los argumento en los registros R0-R3.
  - Llama a una función: BL FUNC.
  - Obtiene el valor que retorna la función llamada en R0.
- Función llamada:
  - Resguarda los registros que puede llegar a modificar (R4-R11, LR, SP) en el stack.
  - Ejecuta la función.
  - Almacena el valor a retornar en R0.
  - Recupera los registros salvados en el stack.
  - Retorna a la función llamadora: MOV PC, LR.

Ejercicio 1

Escriba una función en assembler de ARM que calcule el factorial de un número pasado como parámetro a la función. Utilice el código escrito en lenguaje de alto nivel como referencia.

```
void main(){
  int num = 3;
  int result;

  result = factorial(num);

}

int factorial(int num){
  int result = 1;

  while(num>1){
    result = result * num;
    num = num - 1;
  }
  return result;
}
```

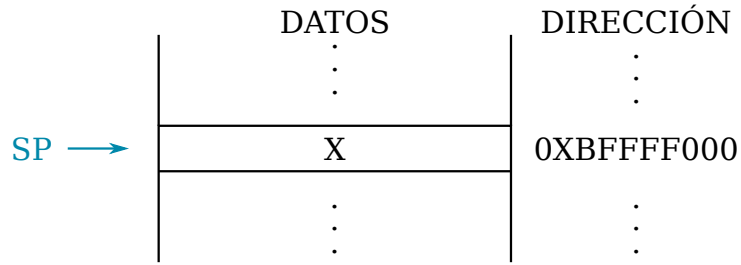
Ejercicio 2

Escriba la función del ejercicio anterior en assembler de ARM utilizando recursividad. Dibujar las imágenes del stack antes, durante y luego de la llamada a la función. Asumir $SP = 0 \times BFFFF000$ justo antes de la llamada a la función.

```
int factorial(int num)
{
  if(num <= 1)
    return 1;

  else
    return(num * factorial(num-1));
}
```

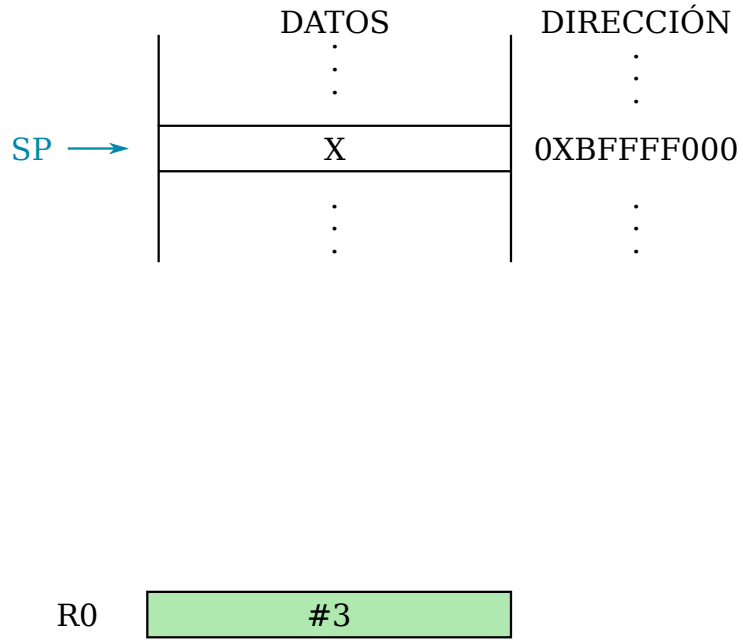
Factorial de n, n!

```
PC ⟶  0X8000  MAIN:       MOV    RO, #3
      0X8004              BL     FACTORIAL
      0X8008              ...

      0X8500  FACTORIAL:  PUSH   {R5, LR}
      0X8504              MOV    R5, R0
      0X8508              CMP    R5, #1
      0X850C              MOVLT  R0, #1
      0X8510              BLE    SALIR
      0X8514              SUB    R0, R5, #1
      0X8518              BL     FACTORIAL
      0X851C              MUL    R0, R0, R5

      0X851C  SALIR:      POP    {R5, LR}
      0X8520              MOV    PC, LR
```



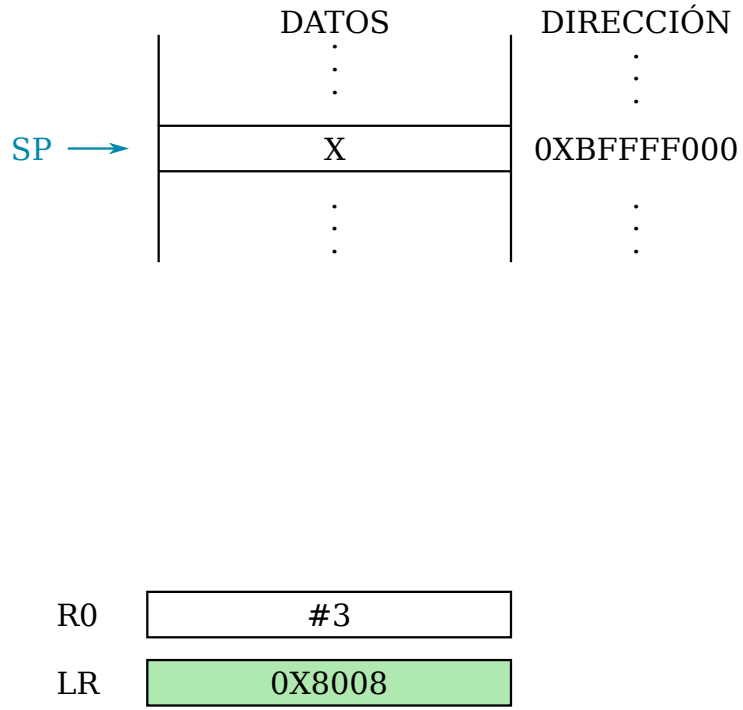
Factorial de n, n!

```
          0X8000   MAIN:       MOV    RO, #3
PC ⟶      0X8004               BL     FACTORIAL
          0X8008               ...

          0X8500   FACTORIAL:  PUSH   {R5, LR}
          0X8504               MOV    R5, R0
          0X8508               CMP    R5, #1
          0X850C               MOVLT  R0, #1
          0X8510               BLE    SALIR
          0X8514               SUB    R0, R5, #1
          0X8518               BL     FACTORIAL
          0X851C               MUL    R0, R0, R5

          0X851C   SALIR:      POP    {R5, LR}
          0X8520               MOV    PC, LR
```



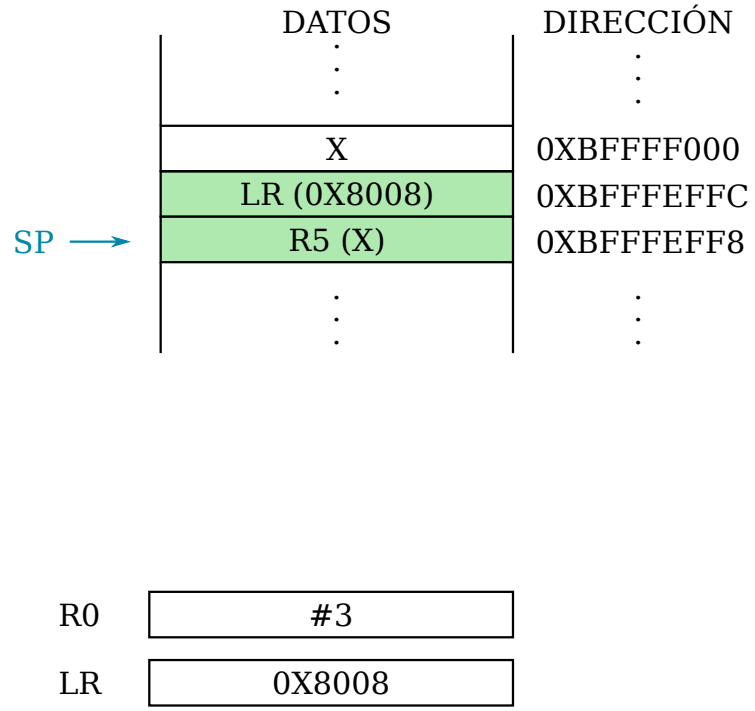
Factorial de n, n!

```
          0X8000   MAIN:       MOV    RO, #3
          0X8004               BL     FACTORIAL
          0X8008               ...

PC ⟶      0X8500   FACTORIAL:  PUSH   {R5, LR}
          0X8504               MOV    R5, R0
          0X8508               CMP    R5, #1
          0X850C               MOVLT  R0, #1
          0X8510               BLE    SALIR
          0X8514               SUB    R0, R5, #1
          0X8518               BL     FACTORIAL
          0X851C               MUL    R0, R0, R5

          0X851C   SALIR:      POP    {R5, LR}
          0X8520               MOV    PC, LR
```



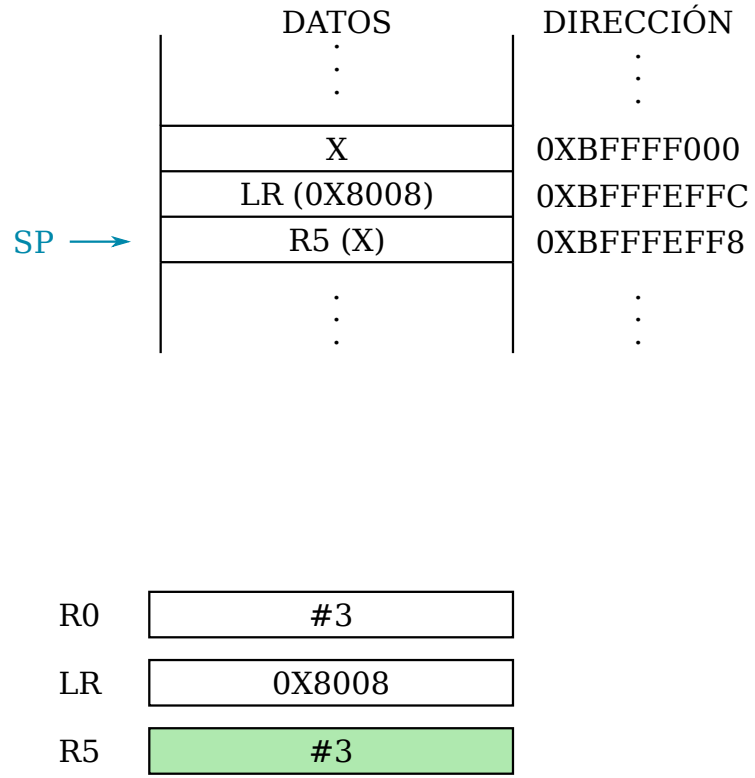
Factorial de n, n!

```
          0X8000   MAIN:       MOV    RO, #3
          0X8004               BL     FACTORIAL
          0X8008               ...

          0X8500   FACTORIAL:  PUSH   {R5, LR}
PC →      0X8504               MOV    R5, R0
          0X8508               CMP    R5, #1
          0X850C               MOVLT  R0, #1
          0X8510               BLE    SALIR
          0X8514               SUB    R0, R5, #1
          0X8518               BL     FACTORIAL
          0X851C               MUL    R0, R0, R5

          0X851C   SALIR:     POP    {R5, LR}
          0X8520               MOV    PC, LR
```



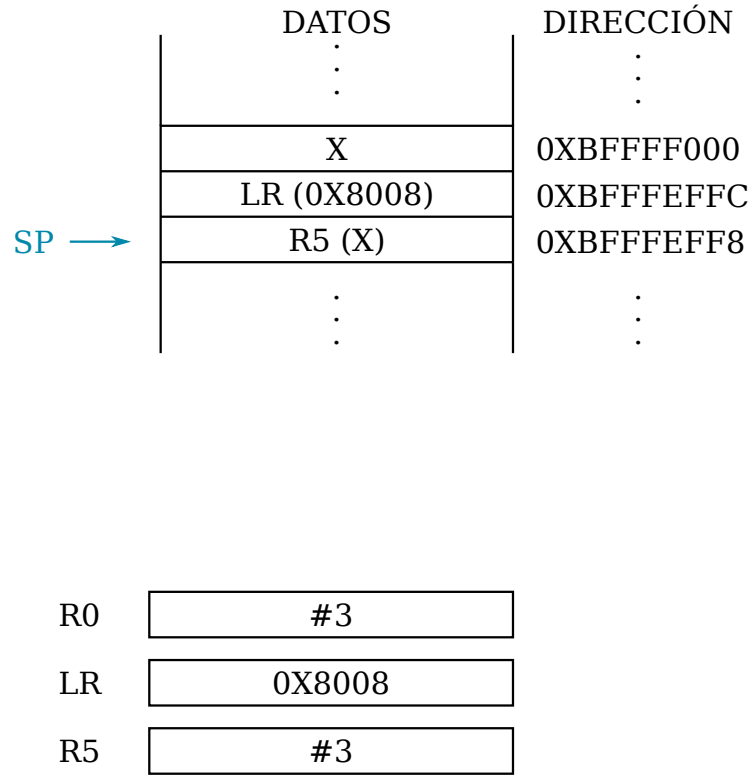
Factorial de n, n!

```
          0X8000   MAIN:       MOV    RO, #3
          0X8004               BL     FACTORIAL
          0X8008               ...

          0X8500   FACTORIAL:  PUSH   {R5, LR}
          0X8504               MOV    R5, R0
PC ⟶      0X8508               CMP    R5, #1
          0X850C               MOVLT  R0, #1
          0X8510               BLE    SALIR
          0X8514               SUB    R0, R5, #1
          0X8518               BL     FACTORIAL
          0X851C               MUL    R0, R0, R5

          0X851C   SALIR:     POP    {R5, LR}
          0X8520               MOV    PC, LR
```



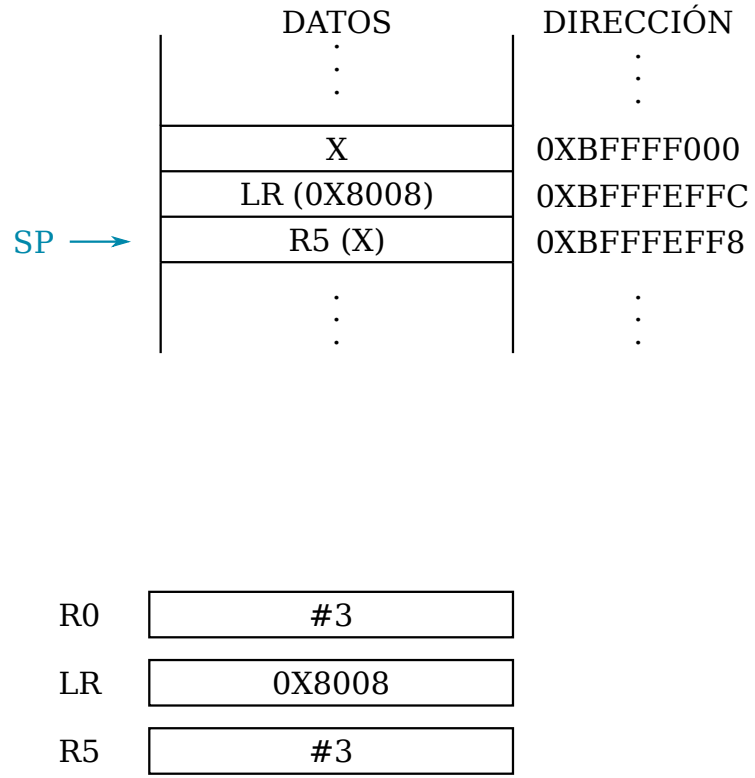
Factorial de n, n!

```
          0X8000   MAIN:       MOV    RO, #3
          0X8004               BL     FACTORIAL
          0X8008               ...

          0X8500   FACTORIAL:  PUSH   {R5, LR}
          0X8504               MOV    R5, R0
          0X8508               CMP    R5, #1
PC ⟶      0X850C               MOVLT  R0, #1
          0X8510               BLE    SALIR
          0X8514               SUB    R0, R5, #1
          0X8518               BL     FACTORIAL
          0X851C               MUL    R0, R0, R5

          0X851C   SALIR:     POP    {R5, LR}
          0X8520               MOV    PC, LR
```



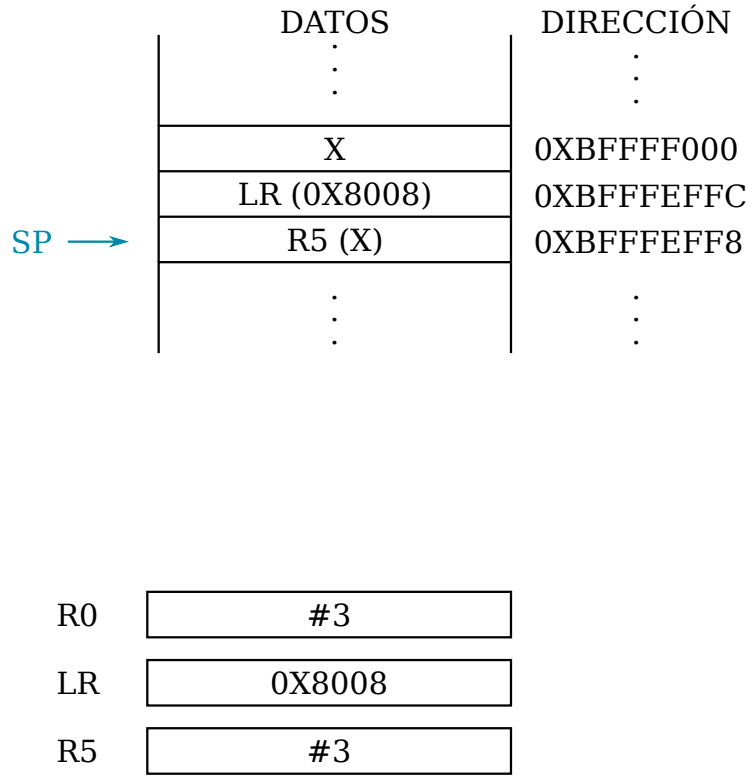
Factorial de n, n!

```
                0X8000   MAIN:       MOV    RO, #3
                0X8004               BL     FACTORIAL
                0X8008               ...

                0X8500   FACTORIAL:  PUSH   {R5, LR}
                0X8504               MOV    R5, R0
                0X8508               CMP    R5, #1
                0X850C               MOVLT  R0, #1
PC ⟶            0X8510               BLE    SALIR
                0X8514               SUB    R0, R5, #1
                0X8518               BL     FACTORIAL
                0X851C               MUL    R0, R0, R5

                0X851C   SALIR:     POP    {R5, LR}
                0X8520               MOV    PC, LR
```



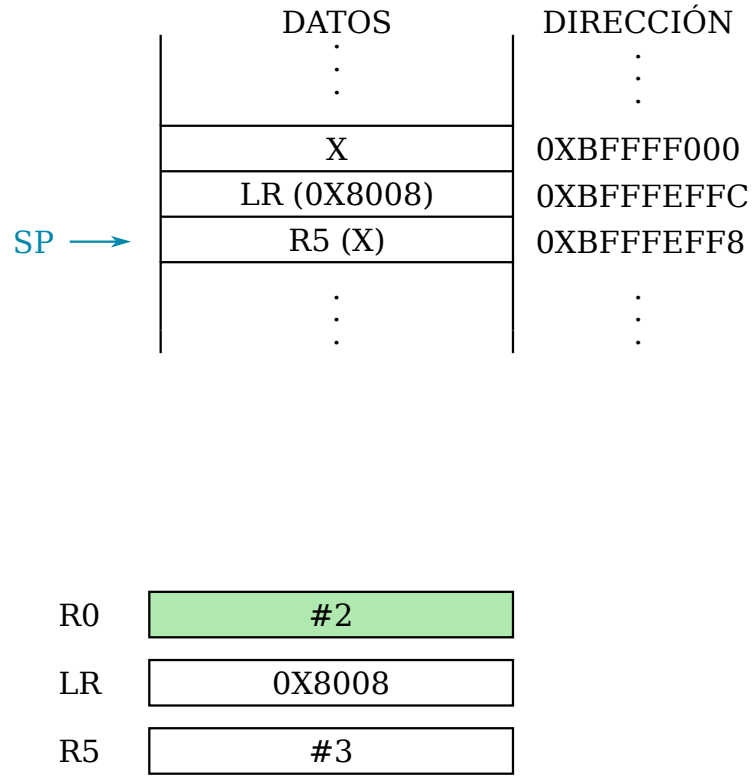
Factorial de n, n!

```
          0X8000   MAIN:       MOV    RO, #3
          0X8004               BL     FACTORIAL
          0X8008               ...

          0X8500   FACTORIAL:  PUSH   {R5, LR}
          0X8504               MOV    R5, R0
          0X8508               CMP    R5, #1
          0X850C               MOVLT  R0, #1
          0X8510               BLE    SALIR
PC ⟶      0X8514               SUB    R0, R5, #1
          0X8518               BL     FACTORIAL
          0X851C               MUL    R0, R0, R5

          0X851C   SALIR:     POP    {R5, LR}
          0X8520               MOV    PC, LR
```



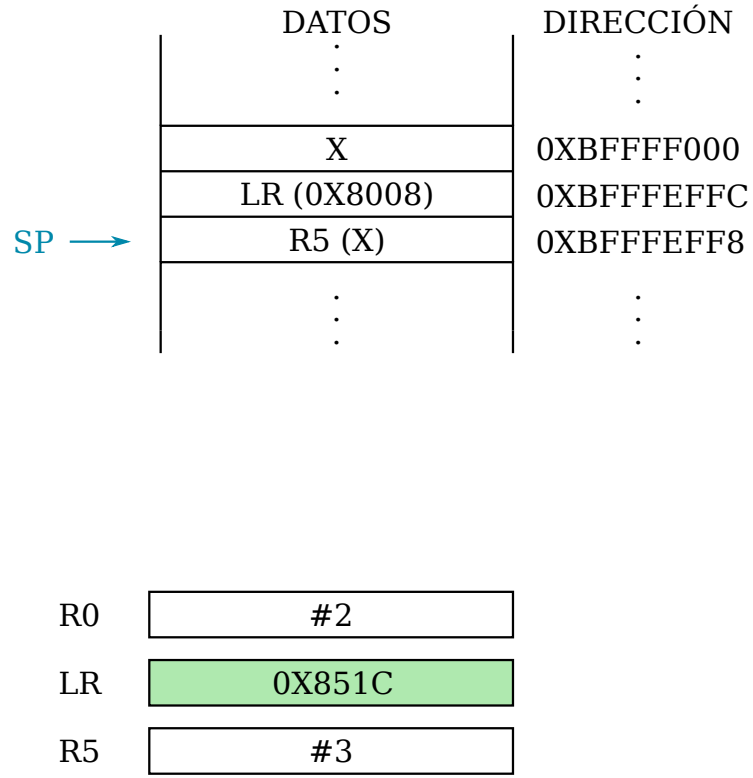
Factorial de n, n!

```
          0X8000   MAIN:       MOV    RO, #3
          0X8004               BL     FACTORIAL
          0X8008               ...

          0X8500   FACTORIAL:  PUSH   {R5, LR}
          0X8504               MOV    R5, R0
          0X8508               CMP    R5, #1
          0X850C               MOVLT  R0, #1
          0X8510               BLE    SALIR
          0X8514               SUB    R0, R5, #1
PC ---->  0X8518               BL     FACTORIAL
          0X851C               MUL    R0, R0, R5

          0X851C   SALIR:     POP    {R5, LR}
          0X8520               MOV    PC, LR
```



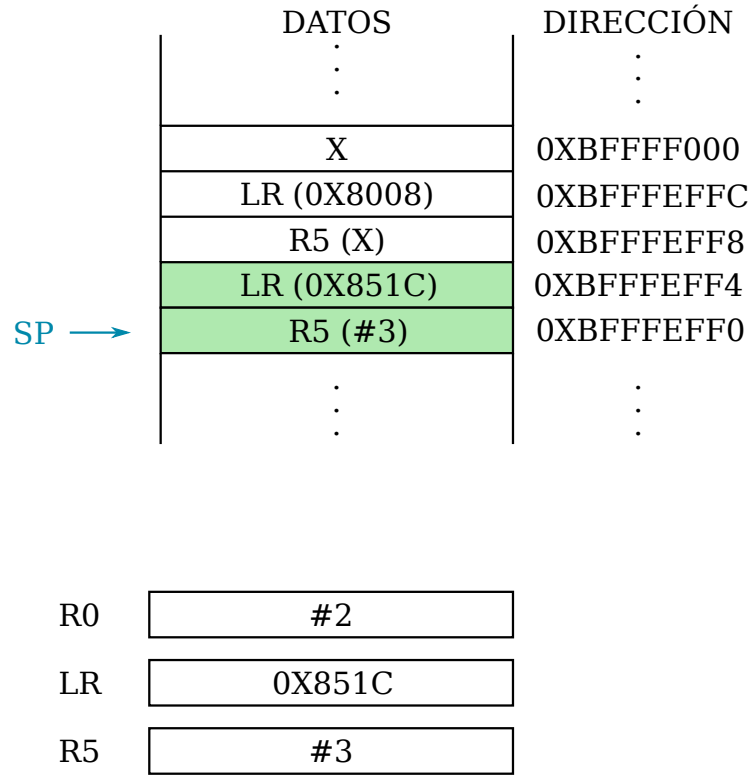
Factorial de n, n!

```
            0X8000   MAIN:       MOV    RO, #3
            0X8004               BL     FACTORIAL
            0X8008               ...

PC ⟶        0X8500   FACTORIAL:  PUSH   {R5, LR}
            0X8504               MOV    R5, R0
            0X8508               CMP    R5, #1
            0X850C               MOVLT  R0, #1
            0X8510               BLE    SALIR
            0X8514               SUB    R0, R5, #1
            0X8518               BL     FACTORIAL
            0X851C               MUL    R0, R0, R5

            0X851C   SALIR:     POP    {R5, LR}
            0X8520               MOV    PC, LR
```



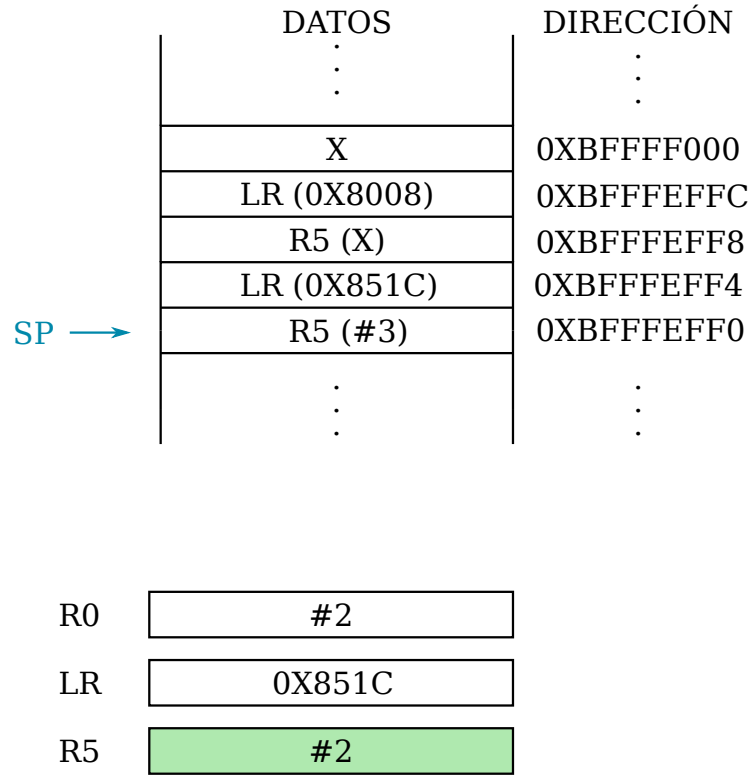
Factorial de n, n!

```
          0X8000   MAIN:       MOV    RO, #3
          0X8004               BL     FACTORIAL
          0X8008               ...

          0X8500   FACTORIAL:  PUSH   {R5, LR}
PC ⟶      0X8504               MOV    R5, R0
          0X8508               CMP    R5, #1
          0X850C               MOVLT  R0, #1
          0X8510               BLE    SALIR
          0X8514               SUB    R0, R5, #1
          0X8518               BL     FACTORIAL
          0X851C               MUL    R0, R0, R5

          0X851C   SALIR:     POP    {R5, LR}
          0X8520               MOV    PC, LR
```



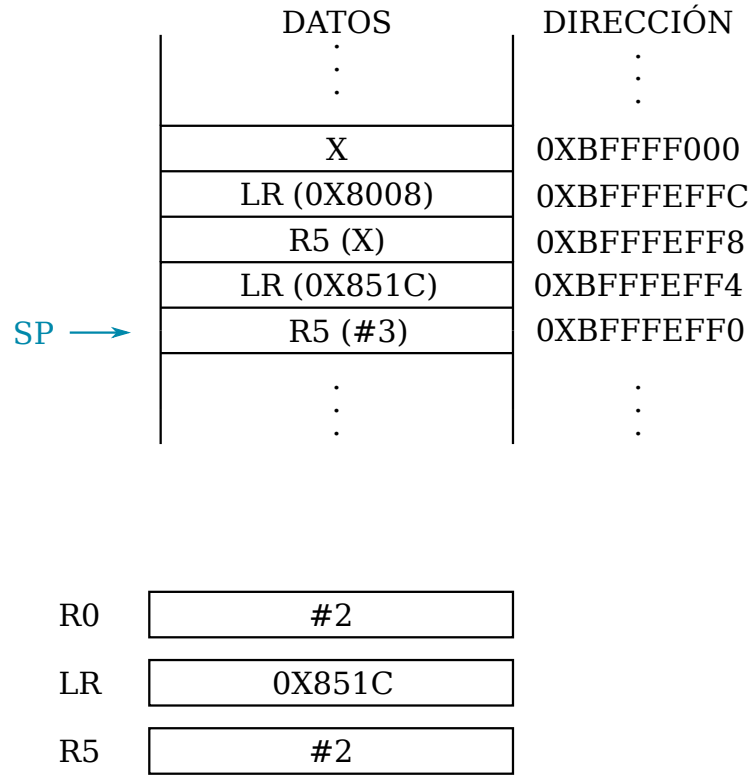
Factorial de n, n!

```
          0X8000   MAIN:       MOV    RO, #3
          0X8004               BL     FACTORIAL
          0X8008               ...

          0X8500   FACTORIAL:  PUSH   {R5, LR}
          0X8504               MOV    R5, R0
PC ⟶      0X8508               CMP    R5, #1
          0X850C               MOVLT  R0, #1
          0X8510               BLE    SALIR
          0X8514               SUB    R0, R5, #1
          0X8518               BL     FACTORIAL
          0X851C               MUL    R0, R0, R5

          0X851C   SALIR:     POP    {R5, LR}
          0X8520               MOV    PC, LR
```



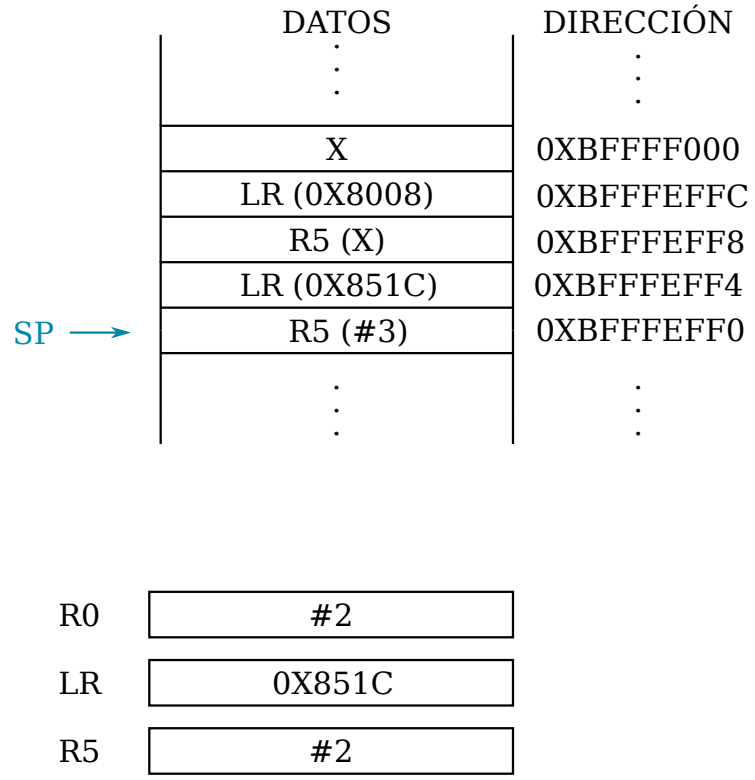
Factorial de n, n!

```
          0X8000   MAIN:       MOV    RO, #3
          0X8004               BL     FACTORIAL
          0X8008               ...

          0X8500   FACTORIAL:  PUSH   {R5, LR}
          0X8504               MOV    R5, R0
          0X8508               CMP    R5, #1
PC ---->  0X850C               MOVLT  R0, #1
          0X8510               BLE    SALIR
          0X8514               SUB    R0, R5, #1
          0X8518               BL     FACTORIAL
          0X851C               MUL    R0, R0, R5

          0X851C   SALIR:     POP    {R5, LR}
          0X8520               MOV    PC, LR
```



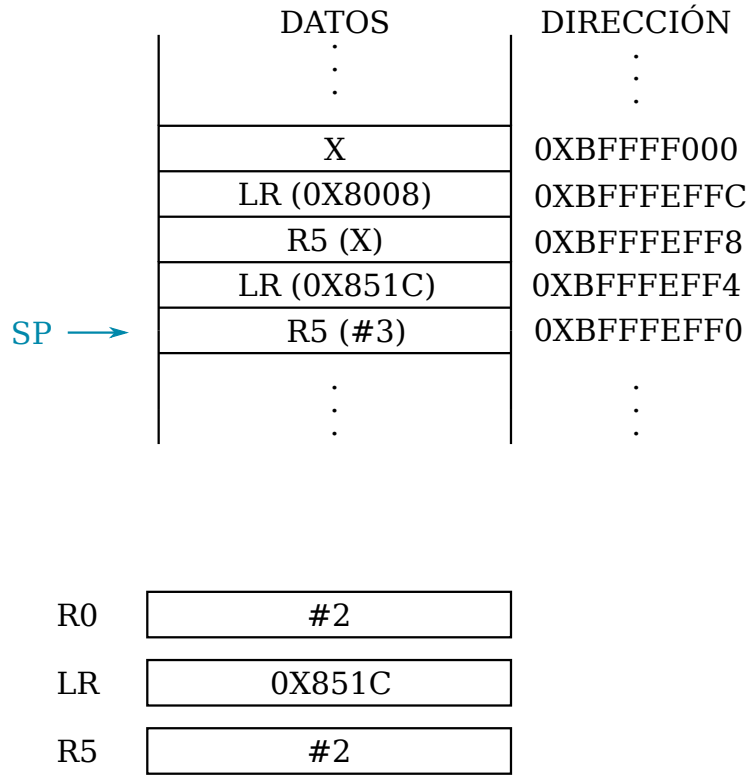
Factorial de n, n!

```
          0X8000  MAIN:      MOV   RO, #3
          0X8004             BL    FACTORIAL
          0X8008             ...

          0X8500  FACTORIAL: PUSH  {R5, LR}
          0X8504             MOV   R5, R0
          0X8508             CMP   R5, #1
          0X850C             MOVLT R0, #1
PC ⟶      0X8510             BLE   SALIR
          0X8514             SUB   R0, R5, #1
          0X8518             BL    FACTORIAL
          0X851C             MUL   R0, R0, R5

          0X851C  SALIR:     POP   {R5, LR}
          0X8520             MOV   PC, LR
```



Factorial de n, n!

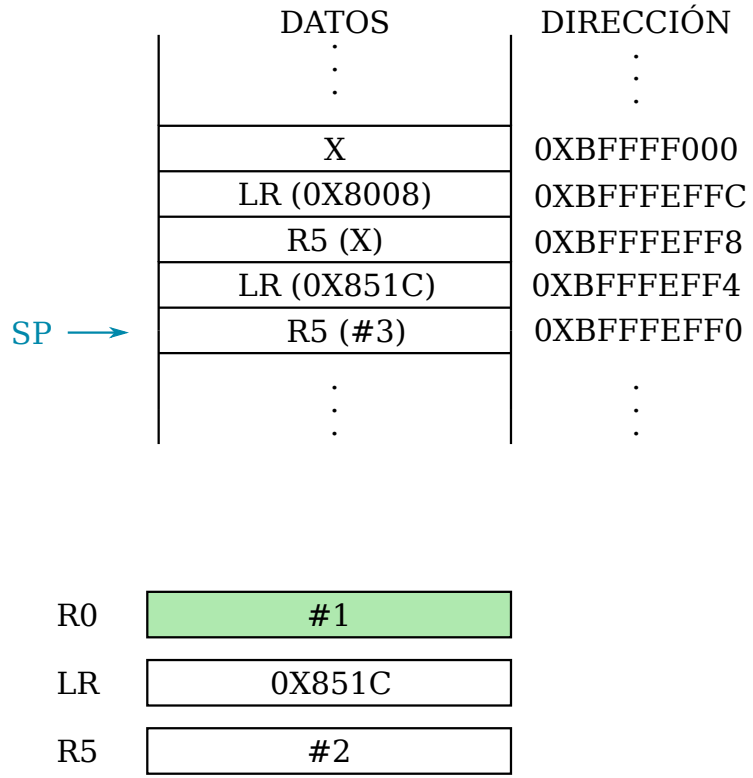
```
          0X8000   MAIN:      MOV    RO, #3
          0X8004              BL     FACTORIAL
          0X8008              ...

          0X8500   FACTORIAL: PUSH   {R5, LR}
          0X8504              MOV    R5, R0
          0X8508              CMP    R5, #1
          0X850C              MOVLT  R0, #1
          0X8510              BLE    SALIR
PC ⟶      0X8514              SUB    R0, R5, #1
          0X8518              BL     FACTORIAL
          0X851C              MUL    R0, R0, R5

          0X851C   SALIR:    POP    {R5, LR}
          0X8520              MOV    PC, LR
```



Factorial de n, n!

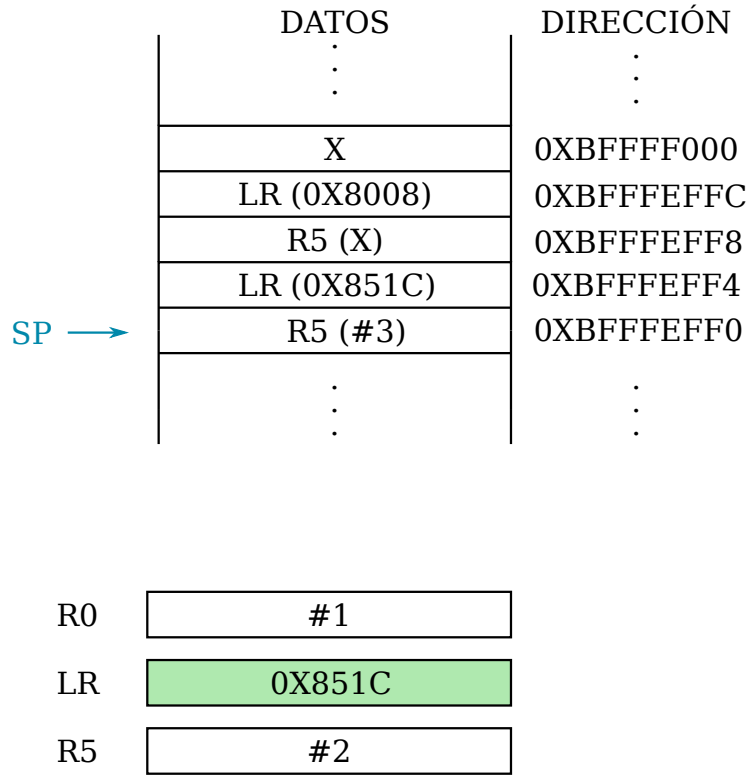
```
0X8000  MAIN:      MOV   RO, #3
0X8004             BL    FACTORIAL
0X8008             ...

0X8500  FACTORIAL: PUSH  {R5, LR}
0X8504             MOV   R5, R0
0X8508             CMP   R5, #1
0X850C             MOVLT R0, #1
0X8510             BLE   SALIR
0X8514             SUB   R0, R5, #1
0X8518             BL    FACTORIAL
0X851C             MUL   R0, R0, R5

0X851C  SALIR:     POP   {R5, LR}
0X8520             MOV   PC, LR
```

PC →



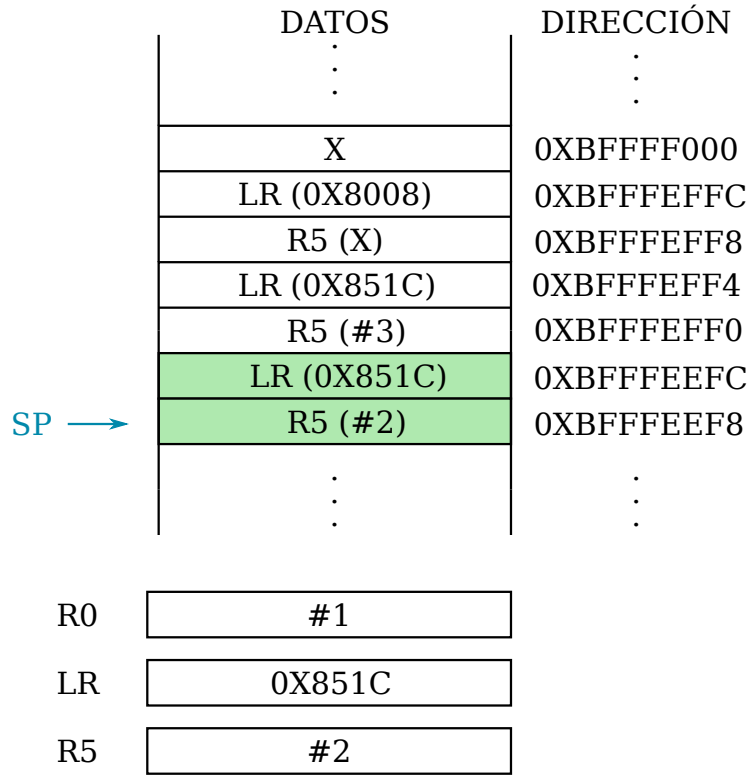
Factorial de n, n!

```
          0X8000   MAIN:       MOV    RO, #3
          0X8004               BL     FACTORIAL
          0X8008               ...

PC ⟶      0X8500   FACTORIAL:  PUSH   {R5, LR}
          0X8504               MOV    R5, R0
          0X8508               CMP    R5, #1
          0X850C               MOVLT  R0, #1
          0X8510               BLE    SALIR
          0X8514               SUB    R0, R5, #1
          0X8518               BL     FACTORIAL
          0X851C               MUL    R0, R0, R5

          0X851C   SALIR:      POP    {R5, LR}
          0X8520               MOV    PC, LR
```



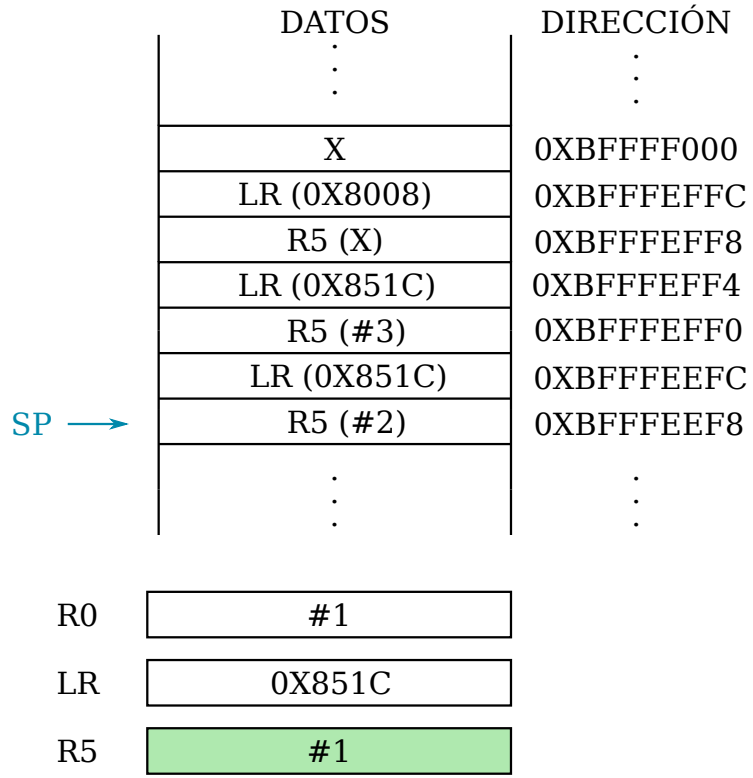
Factorial de n, n!

```
          0X8000   MAIN:      MOV    RO, #3
          0X8004              BL     FACTORIAL
          0X8008              ...

          0X8500   FACTORIAL: PUSH   {R5, LR}
PC ⟶      0X8504              MOV    R5, R0
          0X8508              CMP    R5, #1
          0X850C              MOVLT  R0, #1
          0X8510              BLE    SALIR
          0X8514              SUB    R0, R5, #1
          0X8518              BL     FACTORIAL
          0X851C              MUL    R0, R0, R5

          0X851C   SALIR:    POP    {R5, LR}
          0X8520              MOV    PC, LR
```



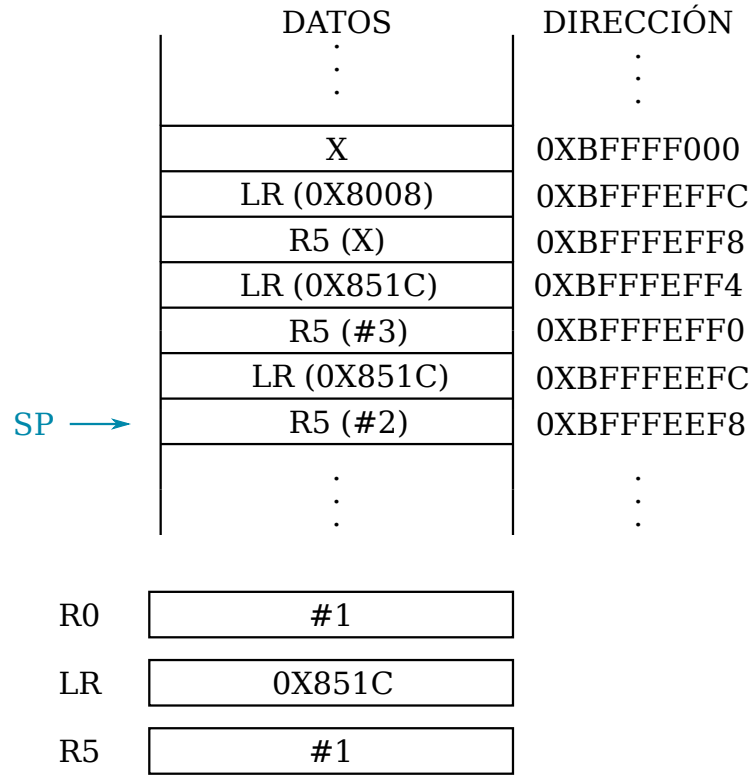
Factorial de n, n!

```
          0X8000   MAIN:       MOV    RO, #3
          0X8004               BL     FACTORIAL
          0X8008               ...

          0X8500   FACTORIAL:  PUSH   {R5, LR}
          0X8504               MOV    R5, R0
PC ⟶      0X8508               CMP    R5, #1
          0X850C               MOVLT  R0, #1
          0X8510               BLE    SALIR
          0X8514               SUB    R0, R5, #1
          0X8518               BL     FACTORIAL
          0X851C               MUL    R0, R0, R5

          0X851C   SALIR:     POP    {R5, LR}
          0X8520               MOV    PC, LR
```



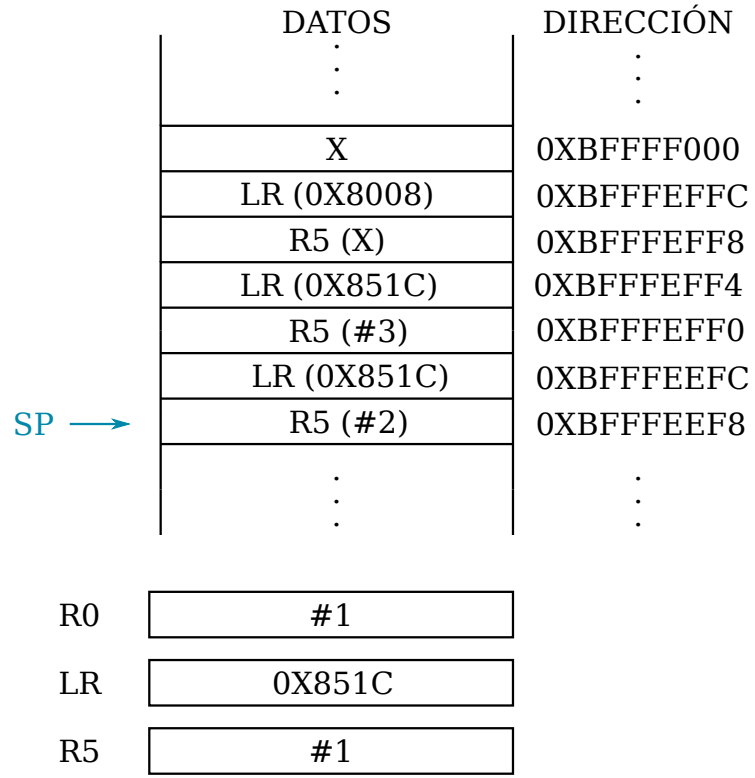
Factorial de n, n!

```
          0X8000   MAIN:       MOV    RO, #3
          0X8004               BL     FACTORIAL
          0X8008               ...

          0X8500   FACTORIAL:  PUSH   {R5, LR}
          0X8504               MOV    R5, R0
          0X8508               CMP    R5, #1
PC ---->  0X850C               MOVLT  R0, #1
          0X8510               BLE    SALIR
          0X8514               SUB    R0, R5, #1
          0X8518               BL     FACTORIAL
          0X851C               MUL    R0, R0, R5

          0X851C   SALIR:     POP    {R5, LR}
          0X8520               MOV    PC, LR
```



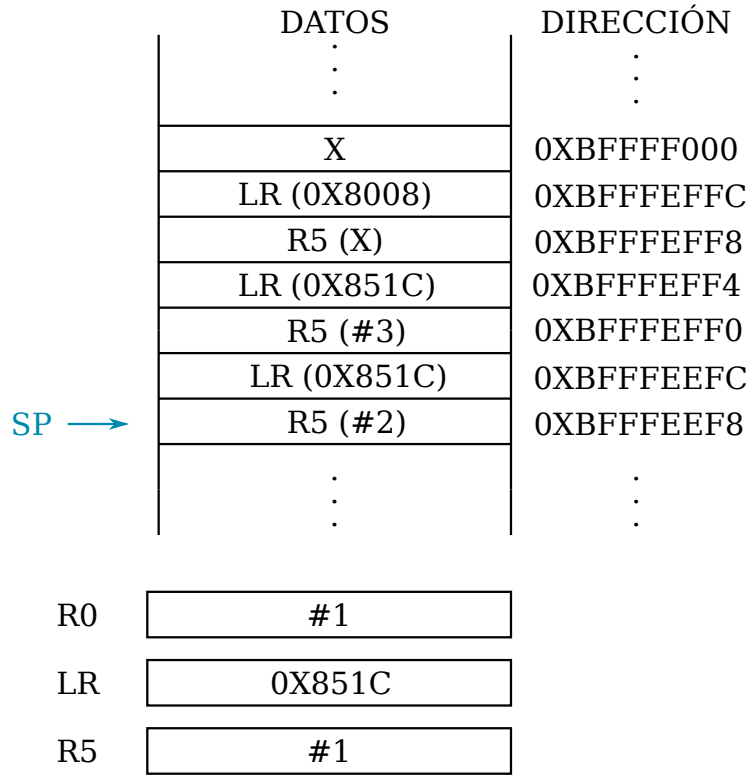
Factorial de n, n!

```
          0X8000   MAIN:       MOV    RO, #3
          0X8004               BL     FACTORIAL
          0X8008               ...

          0X8500   FACTORIAL:  PUSH   {R5, LR}
          0X8504               MOV    R5, R0
          0X8508               CMP    R5, #1
          0X850C               MOVLT  R0, #1
PC ---->  0X8510               BLE    SALIR
          0X8514               SUB    R0, R5, #1
          0X8518               BL     FACTORIAL
          0X851C               MUL    R0, R0, R5

          0X851C   SALIR:     POP    {R5, LR}
          0X8520               MOV    PC, LR
```



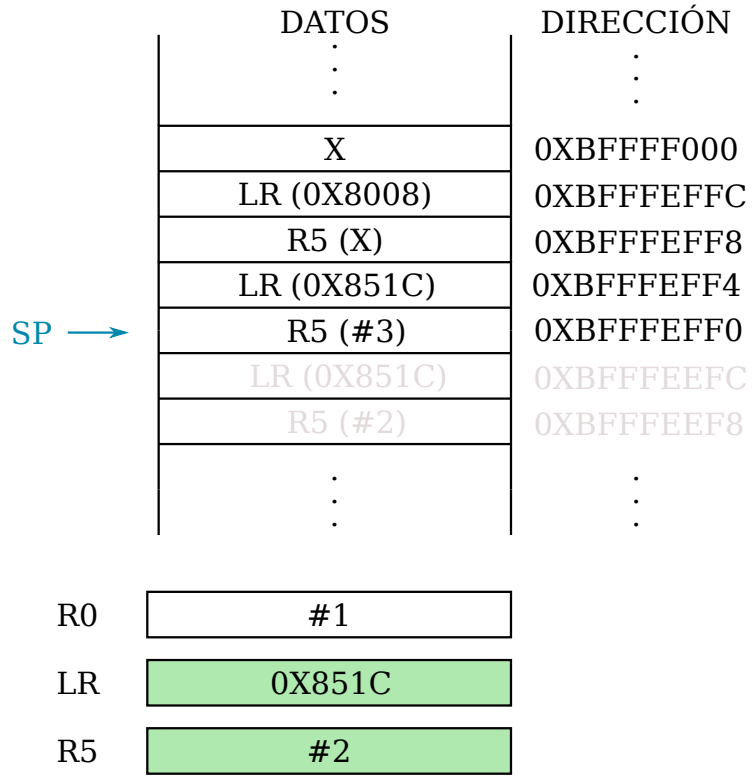
Factorial de n, n!

```
0X8000   MAIN:      MOV    RO, #3
0X8004              BL     FACTORIAL
0X8008              ...

0X8500   FACTORIAL: PUSH   {R5, LR}
0X8504              MOV    R5, R0
0X8508              CMP    R5, #1
0X850C              MOVLT  R0, #1
0X8510              BLE    SALIR
0X8514              SUB    R0, R5, #1
0X8518              BL     FACTORIAL
0X851C              MUL    R0, R0, R5

PC → 0X851C SALIR:  POP    {R5, LR}
0X8520              MOV    PC, LR
```



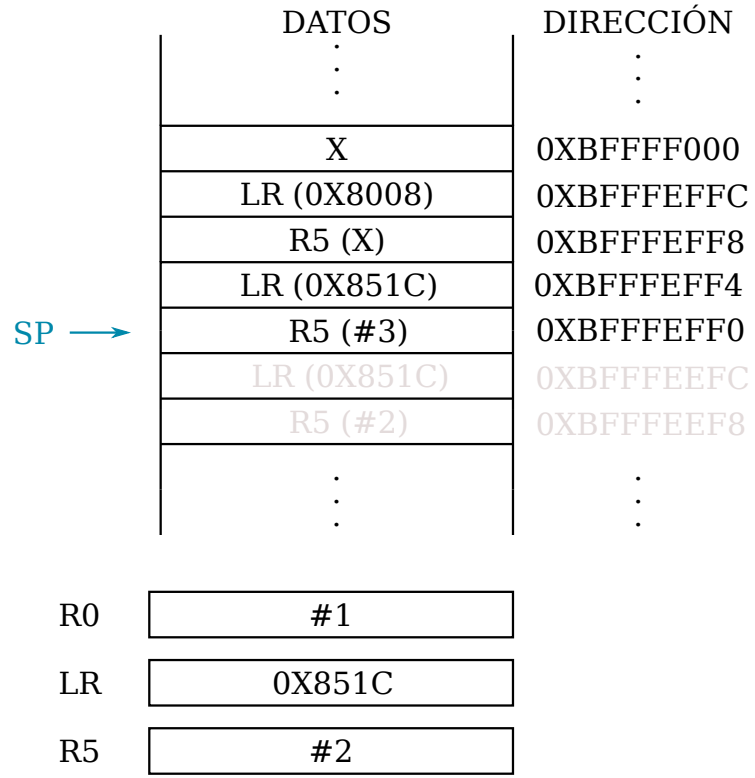
Factorial de n, n!

```
0X8000   MAIN:      MOV    RO, #3
0X8004              BL     FACTORIAL
0X8008              ...

0X8500   FACTORIAL: PUSH   {R5, LR}
0X8504              MOV    R5, R0
0X8508              CMP    R5, #1
0X850C              MOVLT  R0, #1
0X8510              BLE    SALIR
0X8514              SUB    R0, R5, #1
0X8518              BL     FACTORIAL
0X851C              MUL    R0, R0, R5

0X851C   SALIR:     POP    {R5, LR}
PC → 0X8520         MOV    PC, LR
```



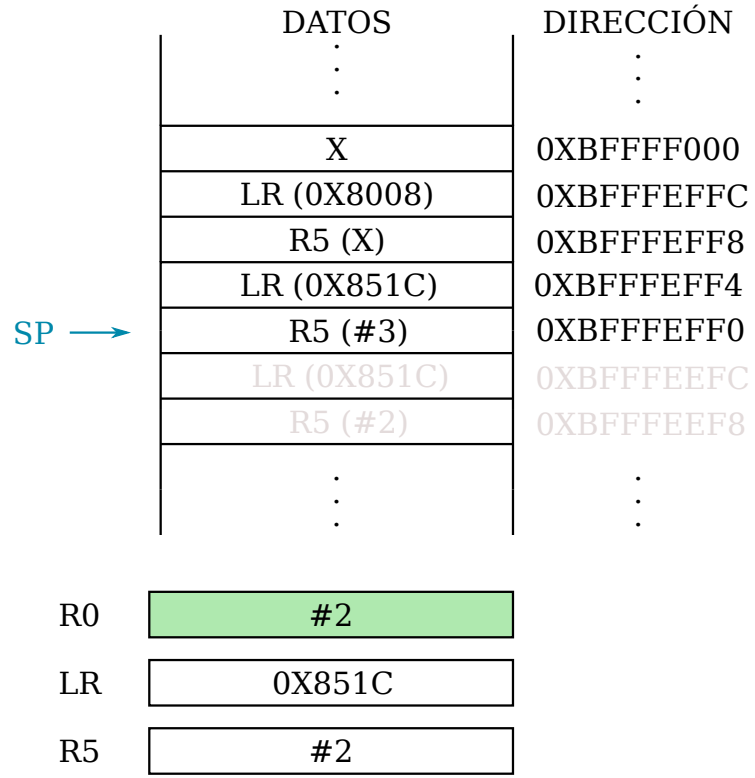
Factorial de n, n!

```
          0X8000  MAIN:      MOV   RO, #3
          0X8004             BL    FACTORIAL
          0X8008             ...

          0X8500  FACTORIAL: PUSH  {R5, LR}
          0X8504             MOV   R5, R0
          0X8508             CMP   R5, #1
          0X850C             MOVLT R0, #1
          0X8510             BLE   SALIR
          0X8514             SUB   R0, R5, #1
          0X8518             BL    FACTORIAL
PC ⟶      0X851C             MUL   R0, R0, R5

          0X851C  SALIR:     POP   {R5, LR}
          0X8520             MOV   PC, LR
```



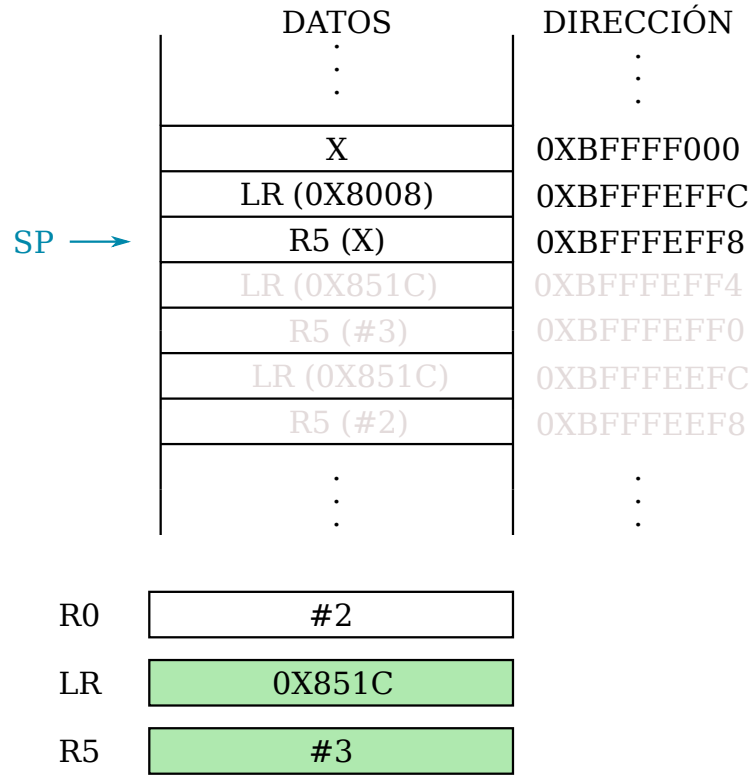
Factorial de n, n!

```
0X8000   MAIN:      MOV    RO, #3
0X8004              BL     FACTORIAL
0X8008              ...

0X8500   FACTORIAL: PUSH   {R5, LR}
0X8504              MOV    R5, R0
0X8508              CMP    R5, #1
0X850C              MOVLT  R0, #1
0X8510              BLE    SALIR
0X8514              SUB    R0, R5, #1
0X8518              BL     FACTORIAL
0X851C              MUL    R0, R0, R5

PC → 0X851C SALIR:  POP    {R5, LR}
0X8520              MOV    PC, LR
```



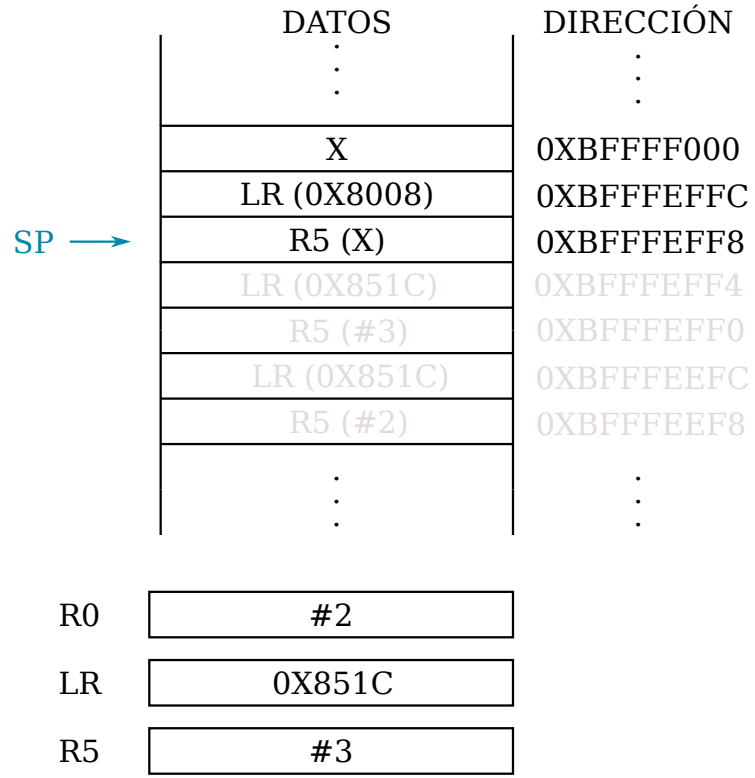
Factorial de n, n!

```
0X8000   MAIN:      MOV    RO, #3
0X8004              BL     FACTORIAL
0X8008              ...

0X8500   FACTORIAL: PUSH   {R5, LR}
0X8504              MOV    R5, R0
0X8508              CMP    R5, #1
0X850C              MOVLT  R0, #1
0X8510              BLE    SALIR
0X8514              SUB    R0, R5, #1
0X8518              BL     FACTORIAL
0X851C              MUL    R0, R0, R5

0X851C   SALIR:     POP    {R5, LR}
PC → 0X8520         MOV    PC, LR
```



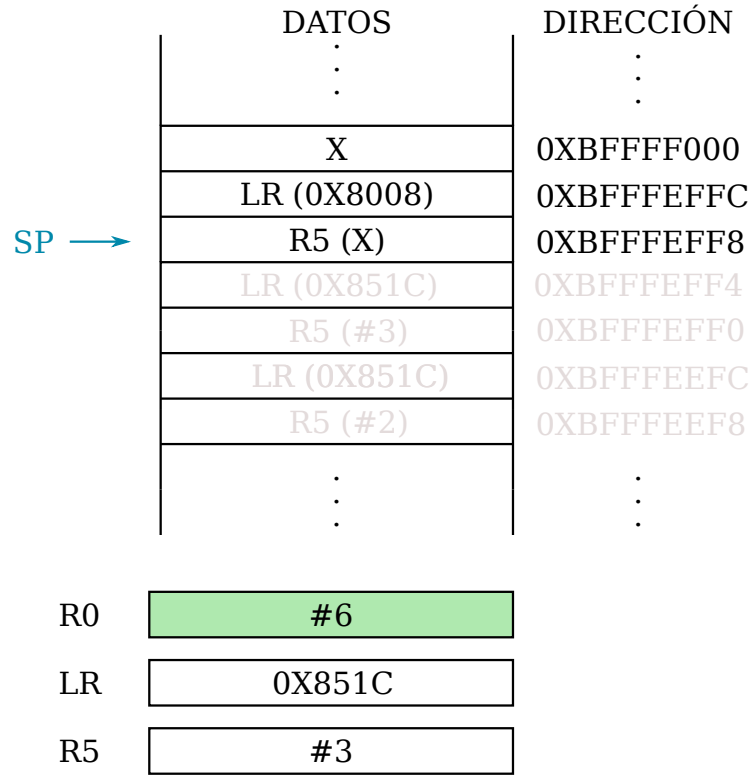
Factorial de n, n!

```
          0X8000   MAIN:       MOV    RO, #3
          0X8004               BL     FACTORIAL
          0X8008               ...

          0X8500   FACTORIAL:  PUSH   {R5, LR}
          0X8504               MOV    R5, R0
          0X8508               CMP    R5, #1
          0X850C               MOVLT  R0, #1
          0X8510               BLE    SALIR
          0X8514               SUB    R0, R5, #1
          0X8518               BL     FACTORIAL
PC ⟶      0X851C               MUL    R0, R0, R5

          0X851C   SALIR:     POP    {R5, LR}
          0X8520               MOV    PC, LR
```



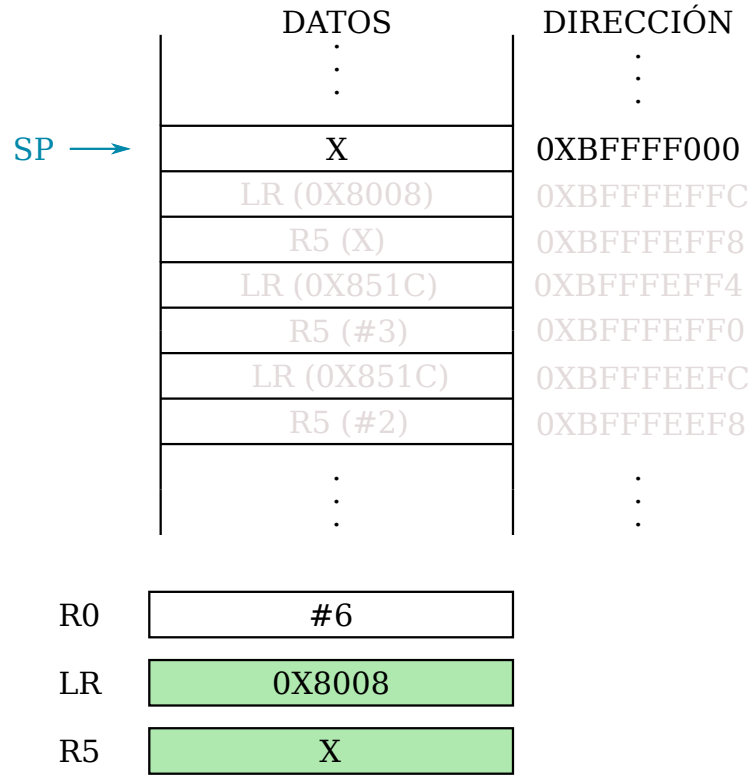
Factorial de n, n!

```
          0X8000   MAIN:      MOV    RO, #3
          0X8004              BL     FACTORIAL
          0X8008              ...

          0X8500   FACTORIAL: PUSH   {R5, LR}
          0X8504              MOV    R5, R0
          0X8508              CMP    R5, #1
          0X850C              MOVLT  R0, #1
          0X8510              BLE    SALIR
          0X8514              SUB    R0, R5, #1
          0X8518              BL     FACTORIAL
          0X851C              MUL    R0, R0, R5

PC ⟶      0X851C   SALIR:    POP    {R5, LR}
          0X8520              MOV    PC, LR
```



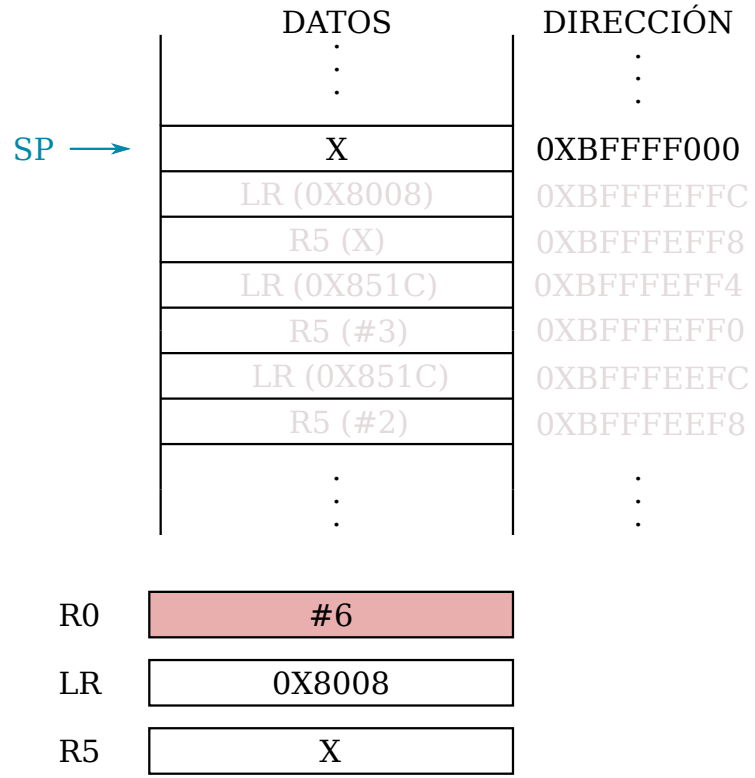
Factorial de n, n!

```
          0X8000   MAIN:       MOV    RO, #3
          0X8004               BL     FACTORIAL
          0X8008               ...

          0X8500   FACTORIAL:  PUSH   {R5, LR}
          0X8504               MOV    R5, R0
          0X8508               CMP    R5, #1
          0X850C               MOVLT  R0, #1
          0X8510               BLE    SALIR
          0X8514               SUB    R0, R5, #1
          0X8518               BL     FACTORIAL
          0X851C               MUL    R0, R0, R5

          0X851C   SALIR:     POP    {R5, LR}
PC ⟶      0X8520               MOV    PC, LR
```



Factorial de n, n!

```
          0X8000  MAIN:      MOV   RO, #3
          0X8004             BL    FACTORIAL
PC ⟶      0X8008             ...

          0X8500  FACTORIAL: PUSH  {R5, LR}
          0X8504             MOV   R5, R0
          0X8508             CMP   R5, #1
          0X850C             MOVLT R0, #1
          0X8510             BLE   SALIR
          0X8514             SUB   R0, R5, #1
          0X8518             BL    FACTORIAL
          0X851C             MUL   R0, R0, R5

          0X851C  SALIR:     POP   {R5, LR}
          0X8520             MOV   PC, LR
```



Ejercicio 3

Escriba una la función en assembler de ARM que calcule la serie de fibonacci de un número pasado parámetro. Proponga una solución iterativa y otra recursiva. Dibujar las imágenes del stack antes, durante y luego de la llamada a la función. Asumir $SP = 0 \times BFF00100$ justo antes de la llamada a la función.
Recuerden:

$$fib(0) = 1, fib(1) = 1$$
$$fib(n) = fig(n-1) + fib(n-2)$$