Puzle de modos de direccionamiento relativos e indirecto

Como problema a resolver en grupo, tras el puzle de los modos de direccionamiento relativos e indirecto, tenéis que hacer un código en ensamblador que resuelva el problema de calcular la nota final de la asignatura ACL, **sin conocer a priori el número de parciales realizados**.

Dicho número de parciales (variable entera **`numParciales`**) estará almacenado en la posición de memoria 19, y las notas de los parciales (variable array de enteros **`notasParciales`**) estarán almacenadas consecutivas a partir de la posición de memoria 20.

$$notaFinal = \sum_{i=1}^{numParciales} notasParciales[i] \qquad (R10 = \sum_{i=20}^{19+mem[19]} \text{mem[i]})$$

Esquema de la memoria:

Posición:	0	1	...	19	20	21	22	...	19+N_p
Contenido:			...	N_p	P_1	P_2	P_3	...	P_{Np}

(N_p = número de parciales, P_n = nota del parcial n)

Para programar dicho código se os permite que modifiqueis la instrucción de carga (LD), de modo que su segundo operando tenga el **modo de direccionamiento que queráis de entre los estudiados en el puzle (relativos o indirectos)**

Especificad a continuación las características de la nueva instrucción elegida (si el operando segundo necesita más de un campo en la instrucción, indicar claramente el nombre del campo y su significado):

`LD Rd, Op2`

- Instrucción de carga (load): Rd ← Mem[_____]
- El operando destino (Rd) tiene modo de direccionamiento directo a registro.
- El operando fuente (Op2) tiene modo de direccionamiento _____. (si está compuesto por más de un campo, indícalos a continuación)

Una vez añadida la nueva instrucción, escribid el código ensamblador que resuelva el problema planteado. Como habéis deducido, este problema no se puede resolver sin usar un bucle. A continuación se os da el esqueleto de un código ensamblador que implementa un bucle, **tan solo tenéis que cargar previamente en r15 el número de iteraciones que queréis que ejecute el bucle**. (NOTA: quizá la lectura complementaria “Bucles en ensamblador, ejecución condicional y etiquetas” te sea de utilidad para hacer este ejercicio).

Programa ensamblador V2: Cálculo de nota ALC relativo

```
// Los comentarios de código comienzan con //

ADDi r10, r0, #0 → R10 comienza en 0
LD r15, 19 ——→ //nº de veces a hacer el bucle.
ADDi r2, r0, 0 ——→ // LD rd, rf, despla

bucle:                          // cabecera bucle (r15 num. iter.)
     beq r15, r0, salir         // si r15 = r0 (0) saltamos fuera
                                // aquí va el cuerpo del bucle
     LD r1, r2, 20 ——→ // ir sumando a y guardado en
     ADDi r2, r2, 1 ——→ // Aquí se le suma 1 a r2 para las siguientes notas
     ADD, r10, r10, r1 ——→ // almacenarlo.
.

     addi r15, r15, -1          // decrementamos contador bucle
     j bucle                    // volvemos al comienzo bucle
salir:                          // aquí va código después del bucle
```

beq → empieza desde 15

LD r1, r2, 20 → se carga en r1, la suma de r2 = 0 y la memoria 20 = 20

ADDi r2, r2, 1 → se le suma 1 a r2 para que después al volver al bucle de = 21, 22 ...

ADDi r10, r10, r1 → Para ir almacenando.