

SOLUCIÓ GOD SAVE THE R0&R1:

```
SALVAR: MOVL R1, -(SP)
        MOVL 4(SP), -(SP)
        MOVL R0, 8(SP)
        RSB
```

```
RESTAURAR: MOVL 8(SP), R0
           MOVL (SP)+, 4(SP)
           MOVL (SP)+, R1
           RSB
```

altres solucions :

```
SALVAR: SUBL #8, SP
        MOVL 8(SP), (SP)
        MOVL R1, 4(SP)
        MOVL R0, 8(SP)
        RSB
```

```
RESTAURAR:
           MOVL 4(SP), R1
           MOVL 8(SP), R0
           MOVL (SP), 8(SP)
           ADDL #8, SP
           RSB
```

SOLUCIÓ Fil d'execució

El fil d'execució d'aquest programa és el següent (a sota apareix el contingut de la pila al final de la execució de l'instrucció):

Inst	I	J	K	L	O	P	L	O	P	Q	L	O	P	R	L	O	P	S
SP		Q	L Q	L L Q	L Q	Q	Q Q	Q		R	R R	R		S	S S	S		

Cada vegada que s'arriba a L es produeix el cicle L, O, P. Hi surt quatre vegades, una per cada BSB i una altra per l'execució seqüencial de les instruccions (no hi ha error amb l'adreça de retorn per haver guardat Q a la pila "manualment").

No es produeix cap desbordament de pila. El programa acaba sense problemes i deixa la pila tal i com la va trobar en començar.

SOLUCIÓ la LUT

Aquest problema, com tots els de programació, admet moltes solucions, fruit de certes decisions, afortunades o no, que fa el programador.

En aquesta solució, els paràmetres que són adreces són *longs*, i els altres *bytes*. Es fa servir adreçament indexat, màscares i les dades es treuen de la pila amb desplaçaments. Els paràmetres s'esborren de la pila en tornar de la subrutina. Si el contingut de una casella de la matriu no es troba a la LUT, no es canvia.

a) La subrutina "APLICA_LUT"

? R0 és l'adreça del vector columna

? R1 és la longitud del vector columna

? R2 és l'adreça base de la LUT

? R3 és la longitud de la LUT

? R4 és el contingut de una casella del vector

i la posició a la LUT del nou valor d'aquesta casella.

La pila, en començar la subrutina, es troba així:

@ retorn	SP
#N_LUT	SP+4
#LUT	SP+5
#N	SP+9
@ vector columna	SP+10

```
aplica_lut: PUSHR ^M<R0,R1,R2,R3,R4>
```

```
          MOVL 20+4(SP),R3
```

```
          MOVB 20+5(SP),R2
```

```
          MOVL 20+9(SP),R1
```

```
          MOVB 20+10(SP),R0
```

```
bucle:   DECB R1
```

```
          MOVB (R0)[R1],R4
```

```
          CMPB R3,R4
```

```
          BLEQ noferres
```

```
          MOVB (R2)[R4],(R0)[R1]
```

```
noferres: CMPB #0,R1
```

```
          BLSS bucle
```

```
          POPR ^M<R0,R1,R2,R3,R4>
```

```
          RSB
```

b) El programa principal

És només un bucle que, per a cada columna de la matriu, empla els paràmetres de la subrutina, la crida i esborra els paràmetres de la pila. Sense fer servir X, com demana l'enunciat.

? R0 és el comptador (des de zero) de les columnes

? R1 apunta a X després d'haver llegit A

```
          MOVB #0,R0
          MOVL A,R1
bucle:   MOVL (R1)[R0],-(SP)
          MOVB #N, -(SP)
          MOVL #LUT, -(SP)
          MOVB #N_LUT, -(SP)
          BSB aplica_lut
          ADDL #10,SP
          INCB R0
          CMPB R0,#N
          BLSS bucle
```

SOLUCIÓ LLARGS I CURTS:

estructura de dades:

```
;definim les longituds com a constants
long0=10
long1=5
...
longn-1=7

M: .LONG vect0,vect1...vectn-1

vect0: .WORD long0
       .BLKW long0
vect1: .WORD long1
       .BLKW long1

...
vectn-1: .WORD longn-1
         .BLKW longn-1
i: .BLKL 1
Z: .BLKW 1
```

com cridar i tornar de la subrutina:

```
MOVL i,-(SP)
MOVL #M,-(SP)
MOVW z,-(SP)
BSBW posar
ADDL #10,SP
```

subrutina:

```
posar: PUSHR #^M<R1,R2,R3,R4,R5>
       MOVW 20+4(SP),R1 ; R1=x
       MOVL 20+6(SP),R2 ; R2=#M
       MOVW 20+10(SP),R3 ; R3=i

       MOVL (R2)[R3],R4 ; R4=#vecti
       CLRL R5
       MOVW (R4),R5 ; R5=longi
       DECL R5
for:   CMPL R5,#0
       BEQL end
       ADDL3 #1,R5,R6
       MOVW (R4)[R5],(R4)[R6]
       DECL R5
       BRB for
end:   MOVW R1,2(R4)
       POPR #^M<R1,R2,R3,R4,R5>
       RSB
```

la pila en el punt 1:

SP :	R1
SP+4:	R2
SP+8:	R3
SP+12:	R4
SP+16:	R5
SP+20:	@retorn
SP+24:	z
SP+26:	#M
SP+30:	i