Estructura de Computadores

Ejercicio

- Dadas las siguientes declaraciones en C, donde `NF` y `NC` son constantes, traduce a MIPS la línea marcada en verde. Asume que `i` y `j` están en los registros `$t0` y `$t1`.

```
int mat[NF][NC];
int f() {
    int i, j;
    ...
    return mat[i+5][j-1];
}
```

```
int mati[5][4];
void func(int veci[4]) {
   int j;
   for (j=0; j<4; j++)
      veci[j] = j;
}
```

```
void main() {
   int i;
   for (i=0; i<5; i++)
      func(&mati[i][0]);
}
```

1. Traduce a MIPS la subrutina `func` utilizando acceso secuencial al vector `veci` (`j` está en `$t0`).
2. Traduce a MIPS la subrutina `main` utilizando acceso secuencial a la matriz `mati`. Presta atención a los registros que utilizas para guardar la variable `i` y el puntero.

Ejercicio

- Traduce a MIPS las dos líneas marcadas en verde en el código en C. Utiliza la técnica de acceso secuencial para los accesos a la matriz A. Asume que las variables `i` y `suma` ocupan los registros `$t0` y `$t1`.

```
void func(int A[N][N]) {
    int i, suma=0;
    for (i=0; i<N; i+=3)
        suma += A[i][N-1-i];
}
```

Donada la següent funció en C:

```
short acces_aleatori(short M[][100], int i) {
        return M[i+2][i-1];
}
```

Completa els requadres del següent fragment de codi en assemblador MIPS per tal que sigui la traducció correcta de la funció anterior:

```
acces_aleatori:
        li      $t0, [          ]
        mult    $t0, [          ]
        mflo    $t0
        addu    $t0, $t0, [          ]
        lh      $v0, [          ]($t0)
        jr      $ra
```

Tenim el següent codi en C, on M i V són variables globals:

```
int M[100][100], V[100];

for (i=10;i<90; i++)
                M[i][i+10]=V[i-10];
```

Completa els quadres de la traducció en assemblador MIPS que hi ha a continuació:

```
        la      $t0, M +
        la      $t1, V
        addiu   $t2, $t0,
bucle:  lw      $t3,          ($t1)
        sw      $t3, 0($t0)
        addiu   $t1, $t1,
        addiu   $t0, $t0,
        bne     $t0, $t2, bucle
```

Donada la següent subrutina en assemblador MIPS:

```
acces_aleatori:
        li      $t0, 400
        mult    $t0, $a1
        mflo    $t0
        addu    $t0, $t0, $a0
        lw      $v0, 840($t0)
        jr      $ra
```

Sabem que és la traducció de la següent funció en C (de la qual desconeixem els valors dels requadres). Completa els requadres amb les expressions vàlides en C per tal que la traducció sigui correcta:

```
int acces_aleatori (int M[][100], int i)
{
        return M[          ][          ];
}
```

Donada la següent declaració en C:

```
void func(int mat[][40], int i) {
        mat[i][20-i] = mat[20-i][i];
}
```

Completa els requadres del següent fragment de codi per tal que sigui la traducció optimitzada de la funció func (pista: determina les dues adreces de memòria en funció de `i`, i observa si contenen algun terme en comú):

```
func:   li      $t2,  [          ]
        mult    $t2, $a1
        mflo    $t2
        subu    $t1, $a0, $t2
        addu    $t0, $a0, $t2
        lw      $t3,  [          ]  ($t1)
        sw      $t3,  [          ]  ($t0)
        jr      $ra
```