```
for (i=0; i<25;i++)
{
    Si A[i] <= 0
        C[i]= -B[i];
    else
        C[i] = B[i];
}
```

```
Variables: $s0 = i, $s1 = dir A[0], $s2 = dir B[0], $s3 = dir C[0]

add $s0, $zero, $zero # i=0

loop:   slti $t0, $s0, 25 # t0=1 si i<25 sino t0=0
        beq $t0, $zero, end-loop

        sll $t1, $s0, 2 # t1=4i
        add $t2, $t1, $s1 # t2 = 4i + dir A[0] = dir A[i]
        lw $t3, 0($t2) # t3 = A[i]
        add $t4, $t1, $s2 # t4 = 4i + dir B[0] = dir B[i]
        lw $t5, 0($t4) # t5 = B[i]
        add $t6, $t1, $s3 # t6 = 4i + dir C[0] = dir C[i]

        blez $t3, L1 # A[i] <= 0 salto a L1
        add $t7, $zero, $t5 # t7=B[i]
        j UNION

        L1:
                sub $t7, $zero, $t5 # t7=-B[i]

        UNION:
                sw $t7, 0($t6)
                addi $s0, $s0, 1 # i++
                j loop

        end-loop: FIN
```

```
mayor = A[0];

for (i=1; i<n; i++)
{
    if (mayor < A[i])
        mayor = A[i];
}
```

```
Variables: $s0 = dir A[0], $s1 = n, $s2 = Mayor, $t0 = i

addi $t0, $zero, 1      # i = 1
lw $t1, 0($s0)  # t1 = A[0]
add $s2, $t1, $zero     # Mayor = A[0]

loop:
        slt $t2, $t0, $s1       # t2=1 si i<n sino t2=0
        beq $t2, $zero, end-loop

        sll $t3, $t0, 2 # t3 = 4i
        add $t4, $t3, $s0       # t4 = dir A[0] + 4i = dir A[i]
        lw $t5, 0($t4)  # t5 = A[i]

        slt $t6, $s2, $t5       # t6=1 si mayor<A[i] sino t6=0
        beq $t6, $zero, VUELTA

        add $s2, $t5, $zero     # mayor = A[i]

        VUELTA:
                addi $t0, $t0, 1        # i++
                j loop

        end-loop: FIN
```

```
for (i=0; i<20; i++)
{
    C[i] = maximo(A[i], B[i]);
}
```

```
Variables: $s0 = i, $s1 = dir A[0], $s2 = dir B[0], $s3 = dir C[0]

add $s0, $zero, $zero   # i = 0

loop:
        slti $t0, $s0, 20       # t0 = 1 si i<20 sino t0 = 0
        beq $t0, $zero, end-loop

        sll $t1, $s0, 2 # t1 = 4i

        add $t2, $t1, $s1       # t2 = dir A[0] + 4i = dir A[i]
        lw $a0, 0($t2)  # a0 = A[i]

        add $t4, $t1, $s2       # t4 = dir B[0] + 4i = dir B[i]
        lw $a1, 0($t4)  # a1 = B[i]

        add $t6, $t1, $s3       # t6 = dir C[0] + 4i = dir C[i]

        jal máximo      #La función leerá $a0 y $a1

        sw $v0, 0($t6)  # C[i] = máximo(A[i], B[i])
                        # El resultado viene en $v0

        addi $s0, $s0, 1        # i++
        j loop


end-loop: FIN


maximo:
        slt $t3, $a0, $a1       #t3 = 1 si A[i]<B[i] sino t3 = 0
        beq $t3, $zero, es_A

        add $v0, $a1, $zero
        jr $ra

        es_A:   add $v0, $a0, $zero
                jr $ra
```

```
for (i=0; i<20; i++)
{
    if(A[i]>0)
        B[i]=1;
    else
        B[i]=0;
}
```

```
Variables: $s0 = dir A[0], $s1 = dir B[0], $t0 = i

add $t0, $zero, $zero   # i = 0

loop:
        slti $t1, $t0, 20       # t1 = 1 si i<20 sino t1 = 0
        beq $t1, $zero, end-loop

        sll $t2, $t0, 2         # t2 = 4i

        add $t3, $t2, $s0       # t3 = 4i + dir A[0] = dir A[i]
        lw $t4, 0($t3)          # t4 = A[i]

        add $t5, $t2, $s1       # t5 = 4i + dir B[0] = dir B[i]

        bgtz $t4, es_mayor      # A[i] > 0
        add $t6, $zero, $zero   # B[i] = 0
        j guardar

        es_mayor:
                        addi $t6, $zero, 1      # B[i] = 1

        guardar:
                        sw $t6, 0($t5)
                        addi $t0, $t0, 1        # i++
                        j loop

end-loop: FIN
```

14) Realizar un programa en C que calcule la sucesión de Fibonacci, tal que:

V → $s0
a→$s1
b→$s2
c→$s3
n→$s4
i→t0

```
a = 0;
b = 1;
for(i=0; i < n; i++)
{
    if(i==0)
    {
        v[i] = 0;
    }
    else if (i==1)
    {
        v[i] = 1;
    }
    else
    {
        c = a+b;
        v[i] = c;
        a = b;
        b = c;
    }
}
```

```
Variables: $s0 = dir V[0], $s1 = a, $s2 = b, $s3 = c, $s4 = n, $t0 = i

add $s1, $zero, $zero   # a = 0
add $t0, $zero, $zero   # i = 0
addi $s2, $zero, 1      # b = 1
addi $t7, $zero, 1      # t7 = 1 (constante para comparar)

loop:
        slt $t1, $t0, $s4       # t1 = 1 si i < n sino t1 = 0
        beq $t1, $zero, end-loop

        sll $t2, $t0, 2         # t2 = 4i
        add $t3, $t2, $s0       # t3 = 4i + dir V[0] = dir V[i]

        beq $t0, $zero, es_0

        beq $t0, $t7, es_1

        add $s3, $s1, $s2       # c = a + b
        add $t5, $zero, $s3     # $t5 = valor c (Usamos $t5 para no perder la dirección en $t3)
        add $s1, $zero, $s2     # a = b
        add $s2, $zero, $s3     # b = c

        j guardar

        es_0:
                add $t5, $zero, $zero   # $t5 = 0
                j guardar
        es_1:
                addi $t5, $zero, 1      # $t5 = 1

        guardar:
                sw $t5, 0($t3)          # Escribimos el valor ($t5) en la dirección ($t3)
                addi $t0, $t0, 1        # i++
                j loop

end-loop: FIN
```