13)

```
# Variaveis associadas aos registradores:
# endBase -> $16

.data
A: .word -2

.text
.globl main
main:
        addi $16, $0,  0x1001   # endBase = 0x00001001
        sll  $16, $16, 0x10             # endBase = 0x10010000

        lw   $8, 0x0($16)               # t0 = mem[ 0 + endBase ]
        sra  $9, $8,   0x19             # t0 >> 31
        beq  $9, $0,   fim              # if (t0 == 0) goto fim
        sub  $8, $0,   $8               # t0 = 0 - t0
        sw   $8, 0x0($16)               # mem[0 + endBase] = t0
fim:
        nop
```

14)

```
.data
temp: .word 60
flag: .word -1

.text
.globl main
main:
        addi $8,  $8,  0x1001                   # t0 = 0x00001001
        sll  $8,  $8,  0x10                     # t0 = 0x10010000
        lw   $9, 0x0($8)                        # t1 = temp
        addi $10, $9,  -30                      # t2 = temp - 30
        addi $11, $9,  -50                      # t3 = temp - 50
        srl  $10, $10, 31                       # t2 = t2 >> 31
        srl  $11, $11, 31                       # t3 = t3 >> 31
        xor  $12, $10, $11                      # t4 = t2 xor t3
        beq  $12, $0,  foraDaFaixa              # if (t4 == 0) goto foraDaFaixa
        sw   $12, 0x4($8)                       # mem[4 + t0] = 1
        j fim                                   # goto fim
foraDaFaixa:
        sw   $12, 0x4($8)                       # mem[4 + t0] = 0
fim:
        nop
```

15)

```
# Variaveis associadas aos registradores:
# endBase -> $16
# i -> $17
# soma -> $18

.text
.globl main
main:
        addi $8,  $0, 0x1001   # t0 = 0x00001001
        sll  $16, $8, 0x10              # endBase = 0x10010000
        addi $10, $0, 0x64              # i = 100

do:
        sll  $8,  $17, 0x2              # i = i * 4
        add  $8,  $8, $16               # t0 = i * 4 + endBase
        sll  $9,  $17, 0x1              # t1 = i * 2
```

```
        addi $9,  $9,  0x1              # t1 = i * 2 + 1
        sw   $9,  0($8)                 # vet[i] = t1
        add  $18, $18, $9               # soma = soma + t1
        addi $17, $17, 0x1              # i = i + 1
        bne  $17, $10, do               # if (i != 0) goto do

        sw  $18, 400($16)               # vet[100] = soma
16)
# Programa:
# i = 0;
# j = 0;
# do {
#       do {
#               aux = vet[j];;
#               vet[j] = vet[j+1];
#               vet[j+1] = aux;
#               j++;
#       } while ( j < 100 );
#               j = 0;
#               i++;
# } while ( i < 100 );

# Variaveis associadas aos registradores
# endBase -> $16
# i -> $17
# j -> $18

.data
a: .word 5 : 20
b: .word 4 : 20
c: .word 3 : 20
d: .word 2 : 20
e: .word 1 : 20


.text
.globl main
main:
        addi $8,  $0,  0x1001           # t0 = 0x00001001
        sll  $16, $8,  0x10             # endBase = 0x10010000
        addi $9,  $0,  0x63             # t1 = 99

doExterno:
        addi $18, $0, 0x0                       # j = 0
        doInterno:
                sll  $8,  $18, 0x2              # t0 = j * 4
                add  $8,  $8,  $16              # t0 = j * 4 + endBase
                lw   $10, 0x0($8)               # aux = vet[j]
                lw   $11, 0x4($8)               # t3 = vet[j+1]

                if:
                        slt  $12, $11, $10      # if (vet[j+1] < vet[j]) t4 = 1; else t4 = 0
                        beq  $12, $0,  fimIf    # if (t4 == 0) goto fimIf
                        sw   $11, 0x0($8)       # vet[j] = vet[j+1]
                        sw   $10, 0x4($8)       # vet[j+1] = aux
                fimIf:

                addi $18, $18, 0x1              # j = j + 1
                bne  $18, $9,  doInterno        # if (j != 99) goto doInterno
```

```
        addi $17, $17, 0x1                      # i = i + 1
        bne      $17, $9,  doExterno            # if (i != 99) goto doExterno
```

17)

```
# Variaveis associadas aos registradores:
# endBase -> $16
# x -> $17
# y -> $18

.data
x: .word 3

.text
.globl main
main:
        addi $8,  $0,  0x1001                   # t0 = 0x00001001
        sll  $16, $8,  0x10                     # endBase = 0x10010000

        lw   $17, 0x0($16)                      # t0 = x
        sll  $8,  $17, 0x1F                     # t0 = x << 31
        srl  $8,  $8,  0x1F                     # t0 = x >> 31
        bne  $8,  $0,   impar                   # if (t0 != 0) goto impar

par:
        mult $17, $17                           # x²
        mflo $8                                 # t0 = x²
        addi $9,  $0, -2                        # t1 = -2
        mult $8,  $9                            # x² * -2
        mflo $9                                 # t1 = -2x²

        mult $8,  $17                           # x³
        mflo $10                                # t2 = x³

        mult $8,  $8                            # x⁴
        mflo $11                                # t3 = x⁴

        add  $8,  $11, $10                      # t0 = x⁴ + x³
        add  $18, $8,  $9                       # y  x⁴ + x⁴ -2x²
        j fim                                   # goto fim

impar:
        mult $17, $17                           # x²
        mflo $8                                 # t0 = x²
        mult $8,  $8                            # x⁴
        mflo $9                                 # t1 = x⁴
        mult $9,  $17                           # x⁵
        mflo $9                                 # t1 = x⁵

        mult $8,  $17                           # x³
        mflo $10                                # t2 = x³

        sub  $8,  $9, $10                       # t0 = x⁵ - x³
        addi $18, $8, 0x1                       # j = x⁵ - x³ + 1
fim:
        sw  $18,  0x4($16)                      # mem[4 + endBase] = y
```

18)

```
# Variaveis associadas aos registradores:
# endBase -> $16
# x -> $17
# y -> $18
```

```
.data
x: .word -1

.text
.globl main
main:
        addi $8,  $0,  0x1001          # t0 = 0x00001001
        sll  $16, $8,  0x10            # endBase = 0x10010000

        lw   $17, 0x0($16)                      # x = mem[0 + endBase]
        slt  $8,  $0,  $17                      # if (0 < x) t0 = 1; else t0 = 0;
        beq  $8,  $0,  xMenorIgual  # if (t1 == 0) goto xMenorIgual

xMaior:
        mult $17, $17                           # x²
        mflo $8                                         # t0 = x²
        mult $8,  $17                           # x³
        mflo $8                                         # t0 = x³

        addi $18, $8, 0x1                       # y = x³ + 1
        j fim                                   # goto fim

xMenorIgual:
        mult $17, $17                           # x²
        mflo $8                                         # t0 = x²
        mult $8,  $8                            # x⁴
        mflo $8                                         # t0 = x³

        addi $18, $8,  -1                       # y = x³ - 1
fim:
        sw   $18, 0x4($16)                      # mem[4 + endBase] = y
19)
# Variaveis associadas aos registradores:
# x -> $16
# y -> $17
# z -> $18
# result -> $19

.text
.globl main
main:
        addi $8,  $0,  0x186A          # t0 = 0x0000186A
        sll  $16, $8,  0x8             # x  = 0x00186A00

        addi $8,  $0,  0x1388          # t0 = 0x00001388
        sll  $17, $8,  0x4             # y  = 0x00013880

        addi $8,  $0,  0x61A8          # t0 = 0x000061A8
        sll  $18, $8,  0x4             # z  = 0x00061A80

        div  $16,  $18                          # x/z
        mflo $8                                         # t0 = x/z
        mult $8,   $17                          # x/z * y
        mflo $19                                        # result = xy/z

        #add  $8,  $0,  $17             # t0 = y

#doMult:
        #add  $9,  $9,  $16             # t1 = t1 + x
        #addi $8,  $8,  -1              # t0 = t0 - 1
```

```
        #bne  $8,  $0,  doMult                  # if (t0 != 0) goto doMult

        #addi $19, $0,  -1                      # result = -1
#doDiv:
        #sub  $9,  $9,  $18                     # t1 = t1 - z
        #addi $19, $19, 0x1                     # t2 = t2 + 1
        #slt  $10, $9,  $0                      # if ( t1 < 0 ) t2 = 1; else t2 = 0;
        #beq  $10, $0,  doDiv                   # if (t2 == 0) goto doDiv
```

20)

```
# Variaveis associadas aos registradores:
# endBase -> $16
# i -> $17
# soma -> $18

.text
.globl main
main:
        addi $8,  $0,  0x1001           # t0 = 0x00001001
        sll  $16, $8,  0x10             # t0 = 0x10010000
        addi $8,  $0,  0x63             # t0 = 99
        addi $17, $0,  0x32             # i = 50

vet:
        sll  $9,  $17, 0x2              # t1 = i * 4
        add  $9,  $9,  $16              # t1 = i * 4 + endBase
        sw   $8,  -4($9)                        # vet[i] = t0
        addi $8,  $8,  -2               # t0 = t0 - 2
        addi $17, $17, -1               # i = i - 1
        bne  $17, $0,  vet              # if (t0 != -1) goto vet

        addi $17, $0,  0x32             # i = 50
soma:
        sll  $9,  $17, 0x2              # t1 = i * 4
        add  $9,  $9,  $16              # t1 = i * 4 + endBase
        lw   $11, -4($9)                        # t3 = vet[i]
        add  $18, $18, $11              # soma = soma + vet[i]
        addi $17, $17, -1               # i = i - 1
        bne  $17, $0, soma              # if (t0 != -1) goto soma

        sw   $18, 200($16)                      # vet[50] = soma
```

21)

```
# Variaveis associadas aos registradores:
# endBase -> $16
# i -> $17

.text
.globl main
main:
        addi $8,  $0,  0x1001           # t0 = 0x00001001
        sll  $16, $8,  0x10             # t0 = 0x10010000
        addi $8,  $0,  0x62             # t0 = 98
        addi $17, $0,  0x32             # i = 50

pares:
        sll  $9,  $17, 0x2              # t1 = i * 4
        add  $9,  $9,  $16              # t1 = i * 4 + endBase
        sw   $8,  -4($9)                        # vet[i] = t0
        addi $8,  $8,  -2               # t0 = t0 - 2
        addi $17, $17, -1               # i = i - 1
        bne  $17, $0,  pares            # if (t0 != -1) goto pares
```

```
        addi $8,  $0,  0x63             # t0 = 99
        addi $17, $0,  0x64             # i = 100
        addi $10, $0,  0x32             # t2 = 50
impares:
        sll  $9,  $17, 0x2              # t1 = i * 4
        add  $9,  $9,  $16              # t1 = i * 4 + endBase
        sw   $8,  -4($9)                        # vet[i] = t0
        addi $8,  $8,  -2               # t0 = t0 - 2
        addi $17, $17, -1               # i = i - 1
        bne  $17, $10, impares                  # if (t0 != -1) goto impares
```

22)

```
# Variaveis associadas aos registradores:
# endBase -> $16
# k -> $17
# x -> $18
# y -> $19

.data
x: .word -2
y: .word 30

.text
.globl main
main:
        addi $8,  $0,  0x1001                   # t0 = 0x00001001
        sll  $16, $8,  0x10                     # t0 = 0x10010000
        lw   $18, 0x0($16)                      # x = mem[0 + endBase]
        lw   $19, 0x4($16)                      # y = mem[4 + endBase]
        add  $8,  $0,  $19                      # t0 = y
do:
        add  $9,  $9,  $18                      # t1 = t1 + x
        addi $8,  $8,  -1                       # t0 = t0 - 1
        bne  $8,  $0,  do                       # if (t0 != 0) goto do

        add  $17, $0,  $9                       # k = x * y
        sw   $17,  0x8($16)                     # mem[8 + endBase] = k
```

23)

```
# Variaveis associadas aos registradores:
# endBase -> $16
# k -> $17
# x -> $18
# y -> $19

.data
x: .word 3
y: .word 4

.text
.globl main
main:
        addi $8,  $0,  0x1001                   # t0 = 0x00001001
        sll  $16, $8,  0x10                     # t0 = 0x10010000
        lw   $18, 0x0($16)                      # x = mem[0 + endBase]
        lw   $19, 0x4($16)                      # y = mem[4 + endBase]
        add  $17, $0,  $10                      # k = x^y
        add  $8,  $0,  $18                      # t0 = x
        add  $10, $0,  $19                      # t2 = y
        add  $10, $10, -1                       # t2 = y - 1
```

```
doExterno:
        add  $9,  $0,  $18              # t1 = x
        add  $11, $0,  $0               # t3 = 0
        doInterno:
                add  $11, $11, $8       # t3 = t3 + t0
                addi $9,  $9,  -1       # t1 = t1 - 1
                bne  $9,  $0,  doInterno # if (t1 != 0) goto doInterno

        add  $8,  $0,  $11                      # t0 = t3
        addi $10, $10, -1                       # t2 = t2 - 1
        bne  $10, $0,  doExterno        # if (t2 != 0) goto doExterno

        add  $17, $0,  $11                      # k = x^y
        sw   $17, 0x8($16)                      # mem[8 + endBase] = k
```