Problemas Tema 2

Problema 1. (1) Operadores lógicos

Expresión	valor binario	valor hex	Expresión	valor binario	valor hex
`x & y`	**0000 0010**	**0x02**	`x && y`	**0000 0001**	**0x01**
`x | y`	**1111 0111**	**0xF7**	`x || y`	**0000 0001**	**0x01**
`~x | ~y`	**1111 1101**	**0xFD**	`!x || !y`	**0000 0000**	**0x00**
`x & !y`	**0000 0000**	**0x00**	`x && ~y`	**0000 0001**	**0x01**

Problema 2. (1) Desplazamientos

`x`		`x << 4`		`x >> 3` (lógico)		`x >> 3` (aritmético)	
hex	binario	hex	binario	hex	binario	hex	binario
`0xF0`	**1111 0000**	**0x00**	**0000 0000**	**0x1E**	**0001 1110**	**0xFE**	**1111 1110**
`0x0F`	**0000 1111**	**0xF0**	**1111 0000**	**0x01**	**0000 0001**	**0x01**	**0000 0001**
`0xCC`	**1100 1100**	**0xC0**	**1100 0000**	**0x19**	**0001 1001**	**0xF9**	**1111 1001**
`0x55`	**0101 0101**	**0x50**	**0101 0000**	**0x0A**	**0000 1010**	**0x0A**	**0000 1010**
`0x80`	**1000 0000**	**0x00**	**0000 0000**	**0x10**	**0001 0000**	**0xF0**	**1111 0000**
`0x02`	**0000 0010**	**0x20**	**0010 0000**	**0x00**	**0000 0000**	**0x00**	**0000 0000**

Problema 5. (2) Traducción

```
      movl $A, %eax
      movl $tabla, %ecx
      movl $0, %ebx
for:  cmpl $256, %ebx
      jge fifor
      movsbl (%eax, %ebx), %edx
      movb (%ecx, %edx), %dl
```

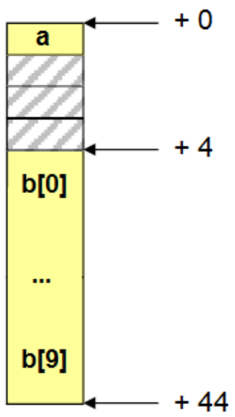
```
          movb %dl, (%eax, %ebx)

          incl %ebx

          jmp for

fifor:
```

Problema 6. (2) Traducción

```
sorpresa: pushl %ebp

          movl %esp, %ebp

          movl 8(%ebp), %ebx

          movl 12(%ebp), %ecx

          cmpl $-10, %ebx

          jle else

          cmpl $10, %ebx

          jge else

          movl %ebx, (%ecx)

          jmp fi

else:     leal 8(%ebp), %ebx

          movl %ebx, 12(%ebp)

fi:       movl 12(%ebp), %eax

          popl %ebp

          ret
```

Problema 9. (1) Structs

a)

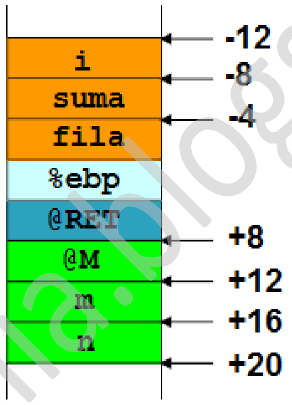


b) @s[i].b[j] = @s + i * 44 + 4 + j * 4

c)
```
imull $44, %esi, %eax
addl %ebx, %eax
imull $44, 4(%eax, %edi, 4), %eax
movb (%ebx, %eax), %dl
```

Problema 10. (1) Subrutinas

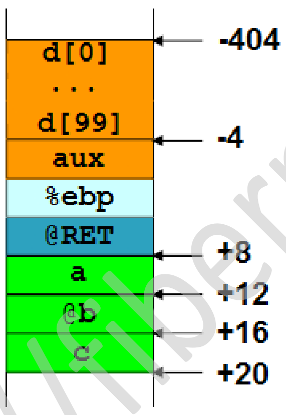
```
calula:  pushl %ebp
         movl %esp, %ebp
         subl $12, %esp
         pushl %ebx
         movl $0, -8(%ebp)
         movl $0, -4(%ebp)
         movl 12(%ebp), %ebx
for:     cmpl 16(%ebp), %ebx
         jge fifor
         leal -4(%ebp), %eax
         pushl %eax
         movl -4(%ebp), %edx
         imull $10, %edx
         addl %ebx, %edx
         movl 8(%ebp), %ecx
         movl (%ecx, %edx, 4), %edx
         pushl %edx
```



```
        call Normaliza
        addl $8, %esp
        addl %eax, -8(%ebp)
        incl %ebx
        jmp for
fifor:  movl -8(%ebp), %eax
        incl %eax
        popl %ebx
        movl %ebp, %esp
        popl %ebp
        ret
```

Problema 14.

a)



b)

```
leal -4(%ebp), %eax
leal -404(%ebp), %ecx
pushl %eax
pushl %ecx
pushl $0
call examen
```

c)

```
      movl $0, %ecx
for:  cmpl $100, %ecx
      jge fifor
      leal -404(%ebp), %eax
      movl (%eax, %ecx, 4), %eax
      movl 12(%ebp), %edx
      movl %eax, (%edx, %ecx, 4)
```

```
                    incl %ecx
                    jmp for
               fifor:
```

d) pushl 16(%ebp)

```
   pushl 12(%ebp)
   pushl 8(%ebp)
   call examen
```

Problema 18.

```
SumaElemento:
        pushl  %ebp
        movl %esp, %ebp

        movl  8(%ebp), %eax   ←  %eax = i
        movl  12(%ebp), %ecx  ←  %ecx = j
        sall  $2, %ecx  ←  %ecx = 4j
        leal  (,%eax,8), %edx  ←  %edx = 8i
        subl  %eax, %edx  ←  %edx = 8i - i
        leal  (%eax, %eax, 4), %eax  ←  %eax = i + 4i
        movl  mat2(%ecx, %eax, 4), %eax  ←  %eax = M[@mat2 + 4j + 4(i + 4i)] = mat2[ i ][ j ]
        addl  mat1(%ecx, %edx, 4), %eax  ←  %eax = mat2[ i ][ j ] + M[@mat1 + 4j + 4(8i - i)]
                                                   = mat2[ i ][ j ] + mat1[ i ][ j ]

        movl %ebp, %esp
        popl %ebp
        ret
```

@mat1[i][j] = @mat1 + 4 (i * N + j) = @mat1 + 4iN + 4j

@mat2[i][j] = @mat2 + 4 (i * M + j) = @mat2 + 4iM + 4j

a) $4iM = 4i + 16i = 20i \implies M = \frac{20i}{4i} = 5$

$4iN = 32i - 4i = 28i \implies N = \frac{28i}{4i} = 7$

b) 13

c) 13

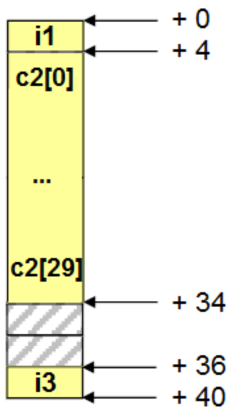
d) 9

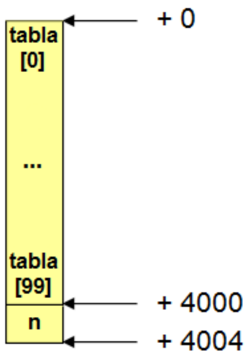
e) Si acceden a memoria: 0.5 i/c (2 c/i); si no acceden a memoria: 0.8 i/c (1.25 c/i)

9 * 2 + 4 * 1.25 = 23 ciclos

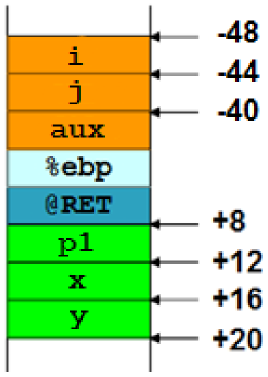
Problema 19.

a)





b)



c)
```
movl 12(%ebp), %eax
movl (%eax), %eax
addl -4(%ebp), %eax
```

d)
```
movl 8(%ebp), %eax
movl -44(%ebp), %ecx
imull $40, %ecx
addl %ecx, %eax
movl 16(%ebp), %ecx
pushl %ecx
pushl %eax
call F
addl $8, %esp
movl %eax, -40(%ebp)
```

```
e)  movl -44(%ebp), %eax
    movl 16(%ebp), %ecx
    imull %eax, %ecx
    movl %ecx, -48(%ebp)
f)  movb -13(%ebp), %al
    leal -40(%ebp), %ecx
    addl $4, %ecx
    addl -48(%ebp), %ecx
    movb %al, (%ecx)
g)      pushl %esi
        movl $0, %eax
        movl 8(%ebp), %ecx
for:    cmpl 16(%ebp), %eax
        jge fifor
        cmpl 4000(%ecx), %eax
        jge fifor
        imull $40, %eax, %edx
        addl %ecx, %edx
        movl %edx, %esi
        movl 36(%esi), %esi
        addl %eax, %esi
        movl %esi, (%edx)
        addl $5, %eax
        jmp for
fifor: popl %esi
```

h)

```
    movl -40(%ebp), %eax
    cmpl 16(%ebp), %eax
    je else
    movl -48(%ebp), %ecx
    jmp end
else: movl -44(%ebp), %ecx
end: movl  %ecx, -4(%ebp)
```

i)

```
    movl $0, %eax
    leal -40(%ebp), %ecx
while: cmpb $`.´, 4(%ecx, %eax)
    je fiwhile
    movb $`#´, 4(%ecx, %eax)
    incl %eax
    jmp while
fiwhile:
```