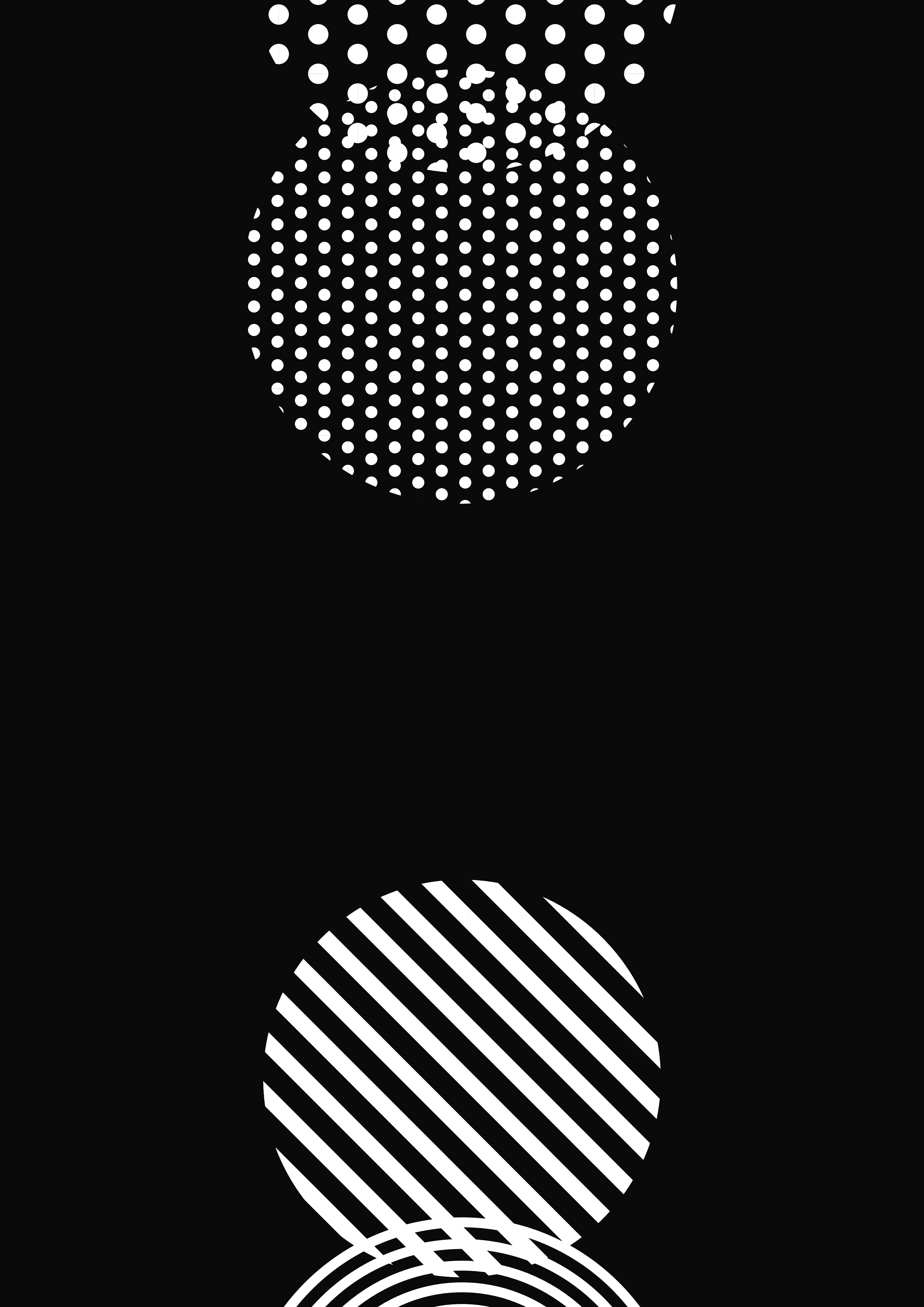


Exercice 1

1.

(a)

$$\dfrac{\dfrac{\Gamma \vdash 1 : int \quad \Gamma \vdash 2 : int}{\Gamma \vdash (1+2) : \tau_2} \qquad \dfrac{\Gamma \vdash 3 : int \qquad \Gamma \vdash [\,] : \tau_2 \ list}{\Gamma \vdash (3 :: [\,]) : \tau_2 \ list}}{\Gamma \vdash (1+2) :: (3 :: [\,]) \ : \ \tau_1}$$

$\tau_2 = int$

(b)

$$\dfrac{\dfrac{\Gamma \vdash 1 : int \quad \Gamma \vdash [\,] : int \ list}{\Gamma \vdash (1 :: [\,]) : \tau_2} \qquad \dfrac{\Gamma \vdash 2 : int \qquad \Gamma \vdash [\,] : int \ list}{\Gamma \vdash (2 :: [\,]) : \tau_2 \ list}}{\Gamma \vdash (1 :: [\,]) :: (2 :: [\,]) \ : \ \tau_1}$$

$\tau_2 = int \ list$

donc $int \ list = int$ contradiction.

(c)

$$\dfrac{\dfrac{\Gamma \vdash 1 : int \quad \Gamma \vdash [\,] : int \ list}{\Gamma \vdash (1 :: [\,]) : \tau_1 \ list} \qquad \dfrac{\Gamma \vdash [\,] : \tau_1 \qquad \Gamma \vdash [\,] : \tau_1 \ list}{\Gamma \vdash [\,] :: [\,] : \tau_1 \ list}}{\Gamma \vdash (1 :: [\,]) :: ([\,] :: [\,]) : \tau_1 \ list}$$

donc $\tau_1 = \tau_2 \ list = int \ list$.

2.

$$\dfrac{\Gamma \vdash e : \tau \ list}{\Gamma \vdash hd \ e : \tau}$$

$$\dfrac{\Gamma \vdash e : \tau \ list}{\Gamma \vdash tl \ e : \tau \ list}$$

3. Supposons que $\Gamma \vdash [e_1 ; \dots ; e_n] : \tau \ list$ est valide

Hyp rec $n = 0$ donc $\Gamma \vdash [\,] : \tau \ list$

montrons que $\Gamma \vdash [\,] : \tau \ list$

donc vrai

$n = 1$ donc $\Gamma \vdash [e_1] : \tau \ list$

donc par la règle [] on a $\Gamma \vdash e_1 : \tau$

de plus on a $\dfrac{\Gamma \vdash e_1 : \sigma \qquad \Gamma \vdash [\,] : \sigma\ list}{\Gamma \vdash e_1 :: [\,] \ : \sigma\ list}$

vérifié.

Passons au récurrence Hyp rec $n-1$

supposons si

$\Gamma \vdash [e_2 ; \dots ; e_n] : \sigma\ list$ alors $\Gamma \vdash e_2 :: \dots :: e_n :: [\,] : \sigma\ list$

supposons $\Gamma \vdash [e_1 ; \dots ; e_n] : \sigma\ list$

d'après des règles on a

$$\Gamma \vdash e_1 : \sigma \ , \ \dots , \ \sigma \vdash e_n : \sigma\ list$$

Donc $\dfrac{\Gamma \vdash e_2 : \sigma \quad \dots \quad \Gamma \vdash e_n : \sigma}{\Gamma \vdash [e_2 ; \dots ; e_n] : list}$

donc par HR $\Gamma \vdash e_2 :: \dots :: e_n :: [\,] : \sigma\ list$

de plus $\dfrac{\Gamma \vdash e_1 : \sigma \qquad \Gamma \vdash e_2 :: \dots :: e_n :: [\,] : \sigma\ list}{\Gamma \vdash e_1 :: (e_2 :: \dots :: e_n :: [\,]) : \sigma\ list}$

d'où la conclusion.

5. (a)

$$\dfrac{\Gamma \vdash_s 1 : int \qquad \dfrac{\Gamma_s \vdash 2 : int \qquad \dfrac{\Gamma \vdash_s [\,] : elist}{\Gamma_s \vdash [\,] : int\ list}}{\Gamma_s \vdash 2 :: [\,] : int\ list}}{\Gamma \vdash_s 1 :: (2 :: [\,]) \ : int\ nelist}$$

b)

$$\dfrac{\dfrac{\Gamma \vdash_s 1 : int \qquad \dfrac{\Gamma \vdash_s [\,] : elist}{\Gamma \vdash_s [\,]\ int\ list}}{\Gamma \vdash_s 1 :: [\,] : int\ list} \qquad \dfrac{\dfrac{\Gamma \vdash_s [\,] : elist}{\Gamma \vdash_s [\,] : int\ list} \qquad \dfrac{\Gamma \vdash_s [\,] : elist}{\Gamma \vdash_s [\,] : (int\ list)\ list}}{\Gamma \vdash_s [\,] :: [\,] : (int\ list)\ nelist}}{\Gamma \vdash_s (1 :: [\,]) :: ([\,] :: [\,]) : (int\ list)\ nelist}$$

6. Supposons que $\Gamma \vdash_s e : \tau$ valide

$\Gamma \vdash_s n : \tau$ valide donc si on pose $\tau' = int$

$\Gamma \vdash n : \tau'$ est valide

$\Gamma \vdash_s e_1 + e_2 : \tau$ donc en posant $\tau' = int$

$\Gamma \vdash e_1 + e_2 : \tau'$ est valide

$\Gamma \vdash_s e : \tau$ nelist donc on a $\Gamma \vdash_s$

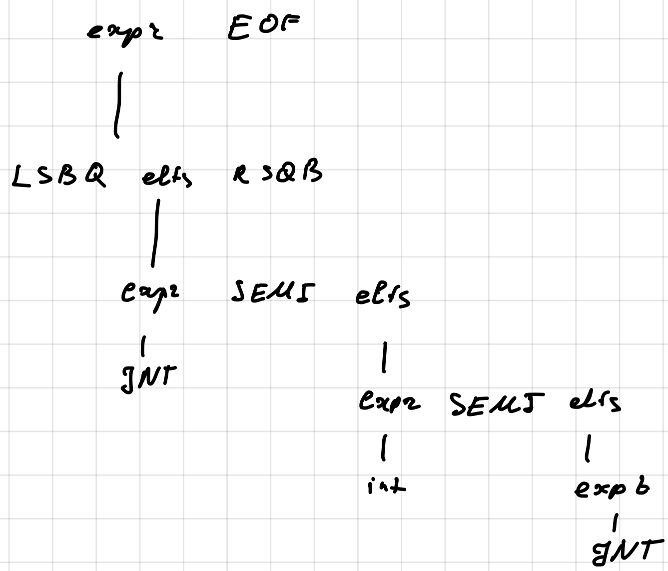
$\Gamma \vdash_s e_0 :: l : \tau$ nelist donc on a $\Gamma \vdash_s e_0 : \tau$ et $\Gamma \vdash_s l : \tau$ list
et on a $\Gamma \vdash e_0 : \tau$ et $\Gamma \vdash l : \tau$ list

d'où $\Gamma \vdash e_0 :: l : \tau$ list

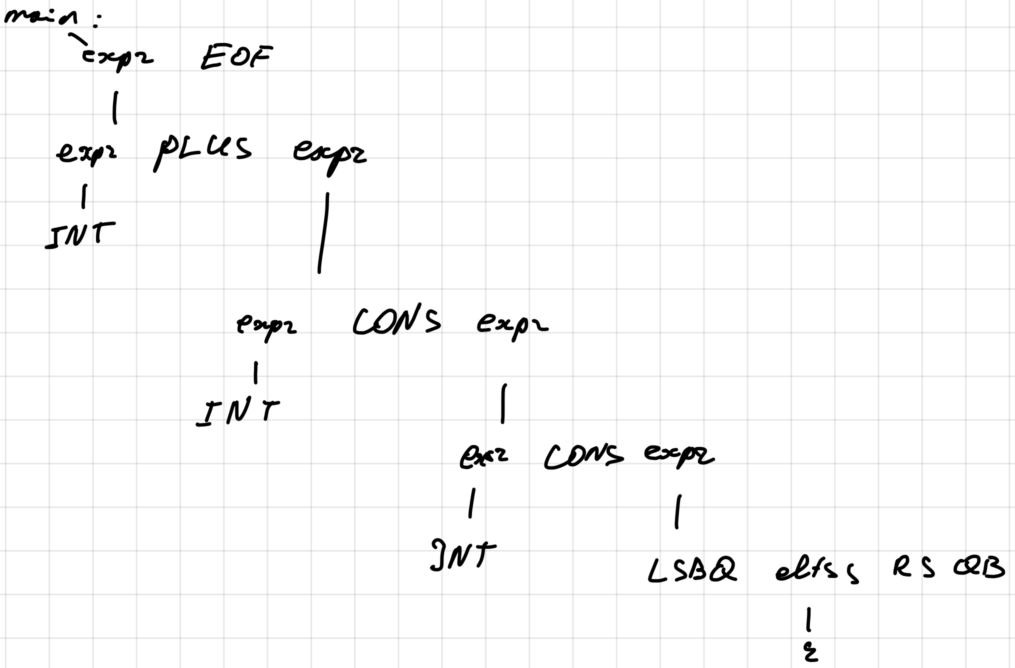
$$\frac{\Gamma \vdash_s e : \tau \text{ nelist}}{\Gamma \vdash_s hd\ e : \tau} \qquad \frac{\Gamma \vdash_s e : \tau \text{ nelist}}{\Gamma \vdash_s tl\ e : \tau \text{ list}}$$

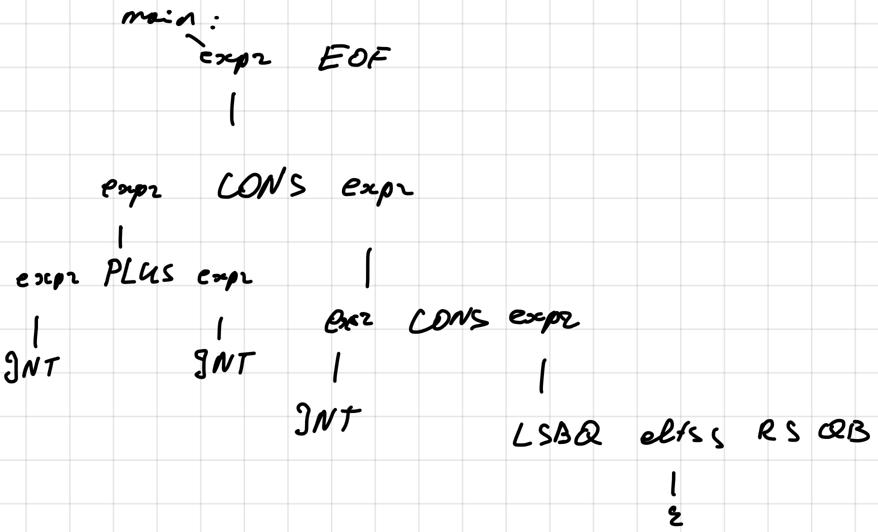
Exercice 2

1. main:



2.



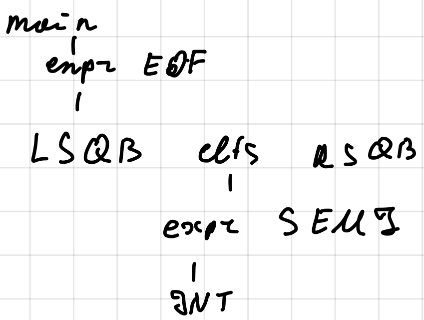


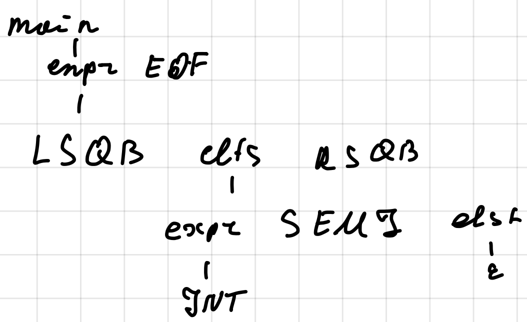
3. La deuxième est correcte

% right CONS
% left PLUS

4.

(a) Quand on a [$\underbrace{e;}_{elts}$] le parseur ne sait pas s'il doit choisir expr semi ou expr semi elts
[3;] ↓ empty





B] On pourrait enlever exp2 SEUJ

Exercice 3

0X08	0x00
	0x04
1	0x08
1	0x012
0X024	0x016
	0x020
1	0x024
2	0x028
0x44	0x032
	0x036
	0x40
1	0x44
3	0x48
0	0x52
	0x56

2. lw $v0 1 $a0

3.

```
and $t0 $a0 $a1
li $t1 1
bneq $t0 $t1 neq
move $t3 $a0
move $t4 $a1
.eqt
    and $t0 $t3 $t4
    or $t1 $t3 $t4
    bneq $t0 $t1 neq
    beqz $t1 finOR
    lw $t0 4($a0)
    lw $t1 4($a1)
    beq $t0 $t1 eqv
    b neq

.finOR



.eqv
    lw $t3 8($a0)
    lw $t4 8($a1)
    b eqt


.neq
```

```
.data
free_blocks:    .word 0
```

```
lw   $t0   free_blocks        # addr  of  f.b

lw   $t1   8($t0)
sw   $t1   free_blocks

li   $t1   1
sw   $t1   0($t0)
move $t1   $a0
sw   $t1   4($t0)
sw   $a1   8($t0)
move $a1   $t0
```

6.
```
defr:        #  defr($a0 = list)
    beqz  $a0   defr_fin
    lw  $t1   8($a0)  # next
    lw  $t2   free_blocks
    sw  $zero   0($a0)
    sw  $t2     8($a0)
    sw  $a0   free_blocks
    move $a0   $t1
    j  defr

defr_fin:
    jr  $ra
```

5. УЖЕ ПОХ

7.
```
new_256_free_blocks:
        li  $t0   256   # counter
        li  $v0   9
        li  $a0   256*12
        syscall
        move $t1  $v0    # addr
        j  loop
loop:
    beqz $t0  end
    sw  $zero  0($t1)
    lw  $t2   free_blocks
    sw  $t2   12($t1)
    sw  $t1   free_blocks
    addi $t1   12
    subi $t0   1
    j  loop
end:
    jr   $ra
```