Si `vector` és un `array` d'enters de 4 bytes definit al rang `1..100`, pel següents programes:

Programa 1

```
void invertir (vector v) {
   int i, j , aux ;

   i = 1;
   j = 100;

   while (i < j) {
      aux  = v[i];
      v[i] = v[j];
      v[j] = aux;

      i = i + 1;
      j = j - 1;
   }
}
```

Programa 2

```
boolean cercar ( vector v, int x) {
   int i = 1;

   while ((v[i] != x) && (x < 100)) {
      i = i + 1;
   }

   return x;
}
```

- Construeix els codis intermedis que generarien les rutines semàntiques.
- Aplica les optimitzacions de finestreta als codis generats.

Programa 1

Segons el vist a classe, el codi de tres adreces pot ser aquest:

	C3@	Codi font corresponent
1	e_{menor}: **skip**	**void** invertir **(**vector v**) {**
2	**pmb** $n_{invertir}$	
3	t_1 = 1	i = 1;
4	i = t_1	
5	t_2 = 100	j = 100;
6	j = t_2	
7	e1: **skip**	
8	**if** i < j **goto** e_2	
9	**goto** e_3	
10	e_2: **skip**	
11	t_3 = -1	i < j
12	**goto** e_4	
13	e_3: **skip**	
14	t_3 = 0	
15	e_4: **skip**	
16	**if** t_3 = 0 **goto** e_5	**if** (·) **{**
17	t_4 = i - 1	
18	t_5 = t_4 * 4	aux = v[i];
19	aux = v[t_5]	
20	t_6 = j - 1	
21	t_7 = t_6 * 4	
22	t_8 = v[t_7]	
23	t_9 = i - 1	v[i] = v[j];
24	t_{10} = t_9 * 4	
25	v[t_{10}] = t_8	
26	t_{11} = j - 1	
27	t_{12} = t_{11} * 4	v[j] = aux;
28	v[t_{12}] = aux	
29	t_{13} = 1	
30	t_{14} = i + t_{13}	i = i + 1;
31	i = t_{14}	
32	t_{15} = 1	
33	t_{16} = j - t_{15}	j = j - 1;
34	j = t_{16}	
35	**goto** e_1	}
36	e_5: **skip**	}
37	**rtn** $n_{invertir}$	

Optimitzacions

1. Brancaments adjacents. No hi ha cap canvi
2. Brancaments sobre brancaments. No hi ha cap canvi. **Nota**: s'aplica un millora en la gestió de les expressions condicionals que estalvia un bot.
3. Assignació de booleans. No hi ha cap canvi
4. Operacions constants. No hi ha cap canvi
5. Eliminació de codi inaccessible. No hi ha cap canvi
6. Desplaçament de constants: No hi ha cap canvi
7. Normalització d'operacions commutatives: suposant que les variables `v`, `i`, `j` i `aux` tenen un valor de `nv` inferior a les variables temporals, i aquestes el tenen pel seu número d'ordre...

	C3@ de partida	C3@ amb l'optimització aplicada
1	$e_{invertir}$: skip	$e_{invertir}$: skip
2	**pmb** $n_{invertir}$	**pmb** $n_{invertir}$
3	t_1 = 1	t_1 = 1
4	i = t_1	i = t_1
5	t_2 = 100	t_2 = 100
6	j = t_2	j = t_2
7	e_1: **skip**	e_1: **skip**
8	**if** i < j **goto** e_2	**if** i < j **goto** e_2
9	**goto** e_3	
10	e_2: **skip**	
11	t_3 = -1	t_3 = 0
12	**goto** e_4	**goto** e_4
13	e_3: **skip**	e_2: **skip**
14	t_3 = 0	t_3 = -1
15	e_4: **skip**	e_4: **skip**
16	**if** t_3 = 0 **goto** e_5	**if** t_3 = 0 **goto** e_5
17	t_4 = i - 1	t_4 = -1 + i
18	t_5 = t_4 * 4	t_5 = 4 * t_4
19	aux = v[t_5]	aux = v[t_5]
20	t_7 = j - 1	t_7 = -1 + j
21	t_8 = t_7 * 4	t_8 = 4 * t_7
22	t_9 = v[t_8]	t_9 = v[t_8]
23	t_{10} = i - 1	t_{10} = -1 + i
24	t_{11} = t_{10} * 4	t_{11} = 4 * t_{10}
25	v[t_{11}] = t_9	v[t_{11}] = t_9
26	t_{12} = j - 1	t_{12} = -1 + j
27	t_{13} = t_{12} * 4	t_{13} = 4 * t_{12}
28	v[t_{13}] = aux	v[t_{13}] = aux
29	t_{14} = 1	t_{14} = 1
30	t_{15} = i + t_{14}	t_{15} = i + t_{14}
31	i = t_{15}	i = t_{15}
32	t_{16} = 1	t_{16} = 1
33	t_{17} = j - t_{16}	t_{17} = j - t_{16}
34	j = t_{17}	j = t_{17}
35	**goto** e_1	**goto** e_1
36	e_5: **skip**	e_5: **skip**
37	**rtn** $n_{invertir}$	**rtn** $n_{invertir}$

8. Assignacions diferides

	C3@ de partida	C3@ amb l'optimització aplicada
1	$e_{invertir}$: skip	$e_{invertir}$: skip
2	**pmb** $n_{invertir}$	**pmb** $n_{invertir}$
3	$t_1 = 1$	
4	$i = t_1$	$i = 1$
5	$t_2 = 100$	
6	$j = t_2$	$j = 100$
7	e_1: **skip**	e_1: **skip**
8	**if** $i < j$ **goto** e_3	**if** $i < j$ **goto** e_2
9		
10		
11	$t_3 = -1$	$t_3 = -1$
12	**goto** e_4	**goto** e_4
13	e_2: **skip**	e_2: **skip**
14	$t_3 = 0$	$t_3 = 0$
15	e_4: **skip**	e_4: **skip**
16	**if** $t_3 = 0$ **goto** e_5	**if** $t_3 = 0$ **goto** e_5
17	$t_4 = -1 + i$	$t_4 = -1 + i$
18	$t_5 = 4 * t_4$	$t_5 = 4 * t_4$
19	$aux = v[t_5]$	$aux = v[t_5]$
20	$t_7 = -1 + j$	$t_7 = -1 + j$
21	$t_8 = 4 * t_7$	$t_8 = 4 * t_7$
22	$t_9 = v[t_8]$	$t_9 = v[t_8]$
23	$t_{10} = -1 + i$	$t_{10} = -1 + i$
24	$t_{11} = 4 * t_{10}$	$t_{11} = 4 * t_{10}$
25	$v[t_{11}] = t_9$	$v[t_{11}] = t_9$
26	$t_{12} = -1 + j$	$t_{12} = -1 + j$
27	$t_{13} = 4 * t_{12}$	$t_{13} = 4 * t_{12}$
28	$v[t_{13}] = aux$	$v[t_{13}] = aux$
29	$t_{14} = 1$	
30	$t_{15} = i + t_{14}$	$t_{15} = 1 + i$
31	$i = t_{15}$	$i = t_{15}$
32	$t_{16} = 1$	
33	$t_{17} = j - t_{16}$	$t_{17} = -1 + j$
34	$j = t_{17}$	$j = t_{17}$
35	**goto** e_1	**goto** e_1
36	e_5: **skip**	e_5: **skip**
37	**rtn** $n_{invertir}$	**rtn** $n_{invertir}$

Notes:

- Les assignacions a t_{15} i t_{17} provoquen altres optimitzacions (tipus 7)

Programa 2

El corresponent codi de tres adreces sense optimitzar pot ser aquest:

	C3@	Codi font corresponent
1	e_{cercar}: **skip**	**boolean** cercar **(** vector v**,** **int** x**) {**
2	**pmb** n_{cercar}	
3	t_1 = 1	i = 1;
4	i = t_1	
5	e_1: **skip**	**while**
6	t_2 = i - 1	
7	t_3 = t_2 * 4	
8	t_4 = v[t_3]	**((**v[i] != x**)**
9	**if** t_4 ≠ x **goto** e_4	
10	**goto** e_3	
11	e_4: **skip**	**&&**
12	t_5 = 100	
13	**if** x < t_5 **goto** e_2	**(**x **<** 100**))**
14	**goto** e_3	
15	e_2: **skip**	**{**
16	t_6 = 1	
17	t_7 = i + t_6	i **=** i **+** 1;
18	i = t_7	
19	**goto** e_1	**}** del bucle
20	e_3: **skip**	
21	**rtn** n_{cerca}, x	**}** del subprograma

Aquest és un cas en què la definició té una assignació i això sí que s'ha de reflectir al codi de tres adreces.

Optimitzacions

1. Brancaments adjacents.

	C3@ de partida	C3@ amb l'optimització aplicada
1	e_{cercar}: **skip**	e_{cercar}: **skip**
2	**pmb** n_{cercar}	**pmb** n_{cercar}
3	t_1 = 1	t_1 = 1
4	i = t_1	i = t_1
5	e_1: **skip**	e_1: **skip**
6	t_2 = i - 1	t_2 = i - 1
7	t_3 = t_2 * 4	t_3 = t_2 * 4
8	t_4 = v[t_3]	t_4 = v[t_3]
9	**if** t_4 ≠ x **goto** e_4	**if** t_4 = x **goto** e_3
10	**goto** e_3	
11	e_4: **skip**	
12	t_5 = 100	t_5 = 100
13	**if** x < t_5 **goto** e_2	**if** x ≥ t_5 **goto** e_3
14	**goto** e_3	
15	e_2: **skip**	
16	t_6 = 1	t_6 = 1
17	t_7 = i + t_6	t_7 = i + t_6
18	i = t_7	i = t_7
19	**goto** e_1	**goto** e_1
20	e_3: **skip**	e_3: **skip**
21	**rtn** n_{cerca}, x	**rtn** n_{cerca}, x

2. Brancaments sobre brancaments. No hi ha cap canvi
3. Assignació de booleans. No hi ha cap canvi

4. Operacions constants. No hi ha cap canvi
5. Eliminació de codi inaccessible. No hi ha cap canvi
6. Desplaçament de constants: No hi ha cap canvi
7. Normalització d'operacions commutatives: suposant que les variables `v`, `x` i `i` tenen un valor de `nv` inferior a les variables temporals, i aquestes el tenen pel seu número d'ordre...

	C3@ de partida	C3@ amb l'optimització aplicada
1	e_{cercar}: **skip**	e_{cercar}: **skip**
2	**pmb** n_{cercar}	**pmb** n_{cercar}
3	$t_1 = 1$	$t_1 = 1$
4	$i = t_1$	$i = t_1$
5	e_1: **skip**	e_1: **skip**
6	$t_2 = i - 1$	$t_2 = -1 + i$
7	$t_3 = t_2 * 4$	$t_3 = 4 * t_2$
8	$t_4 = v[t_3]$	$t_4 = v[t_3]$
9	**if** $t_4 \neq x$ **goto** e_3	**if** $t_4 = x$ **goto** e_3
10		
11		
12	$t_5 = 100$	$t_5 = 100$
13	**if** $x \geq t_5$ **goto** e_3	**if** $x \geq t_5$ **goto** e_3
14		
15		
16	$t_6 = 1$	$t_6 = 1$
17	$t_7 = i + t_6$	$t_7 = i + t_6$
18	$i = t_7$	$i = t_7$
19	**goto** e_1	**goto** e_1
20	e_3: **skip**	e_3: **skip**
21	**rtn** n_{cerca}, x	**rtn** n_{cerca}, x

8. Assignacions diferides

	C3@ de partida	C3@ amb l'optimització aplicada
1	e_{cercar}: **skip**	e_{cercar}: **skip**
2	**pmb** n_{cercar}	**pmb** n_{cercar}
3	$t_1 = 1$	
4	$i = t_1$	$i = 1$
5	e_1: **skip**	e_1: **skip**
6	$t_2 = -1 + i$	$t_2 = -1 + i$
7	$t_3 = 4 * t_2$	$t_3 = 4 * t_2$
8	$t_4 = v[t_3]$	$t_4 = v[t_3]$
9	**if** $t_4 = x$ **goto** e_3	**if** $t_4 = x$ **goto** e_3
10		
11		
12	$t_5 = 100$	
13	**if** $x \geq t_5$ **goto** e_3	**if** $x \geq 100$ **goto** e_3
14		
15		
16	$t_6 = 1$	
17	$t_7 = i + t_6$	$t_7 = 1 + i$
18	$i = t_7$	$i = t_7$
19	**goto** e_1	**goto** e_1
20	e_3: **skip**	e_3: **skip**
21	**rtn** n_{cerca}, x	**rtn** n_{cerca}, x

Notes:

- L'assignació a t_6 provoca altres optimitzacions (tipus 7)