```
(defrule inicio
(vector $?n)
=>
(assert (vector-au ?n)))
(defrule ordena
?f <- (vector-au $?b ?num1 ?num2&:(< ?num2 ?num1) $?e)
=>
(retract ?f)
(assert (vector-au $?b ?num2 ?num1 $?e)))
(defrule final
(not (vector-au $?b ?num1 ?num2&:(< ?num2 ?num1) $?e))
(vector $?n)
(vector-au $?y)
=>
(printout t "orden  " ?n " es " ?y crlf))
```

---------- Ejecutar Archivo ----------

```
CLIPS>(load"ordenar.CLP")
CLIPS>(watch facts)
CLIPS>(watch rules)
CLIPS>(reset)
CLIPS>(run)
```

---------- Resultado ----------

```
FIRE    1 inicio: f-1
==> f-2     (vector-au 3 2 1 4)
FIRE    2 ordena: f-2
<== f-2     (vector-au 3 2 1 4)
==> f-3     (vector-au 2 3 1 4)
FIRE    3 ordena: f-3
<== f-3     (vector-au 2 3 1 4)
==> f-4     (vector-au 2 1 3 4)
FIRE    4 ordena: f-4
<== f-4     (vector-au 2 1 3 4)
==> f-5     (vector-au 1 2 3 4)
FIRE    5 final: *,f-1,f-5

orden  (3 2 1 4) es (1 2 3 4)
```