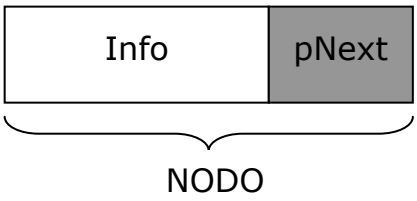
PSEUDOCODIFICA LISTA DINAMICA COSTITUITA DA UNA SERIE DI NODI

ALGORITMO ListaSerieNodiDinamica

TIPO NODO = **RECORD**
Info : INT
pNext: PUNTATORE A NODO
FINE RECORD

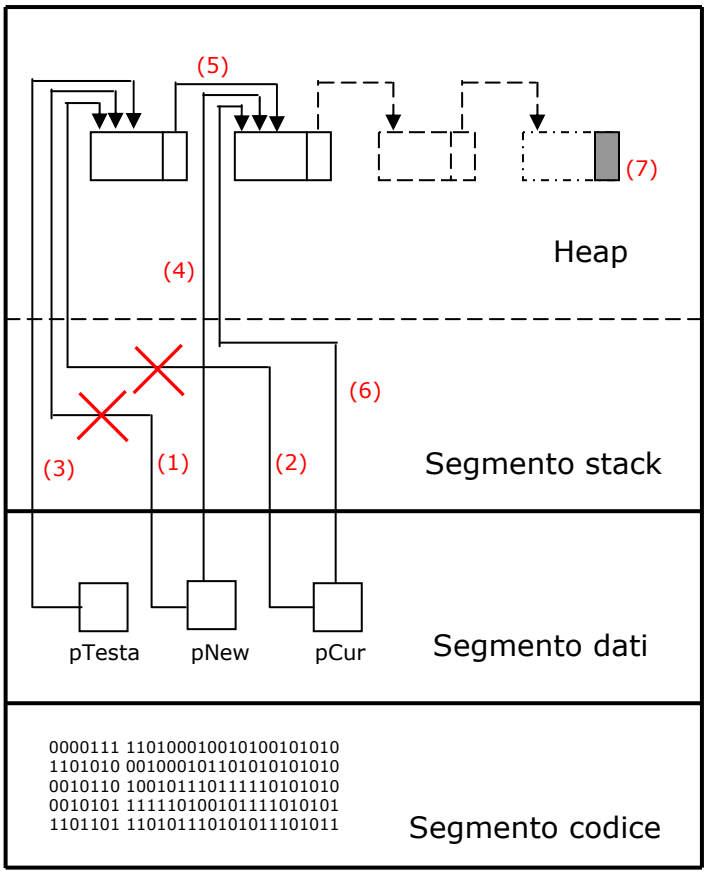


PROCEDURA main( )

pTesta, pCur, PNew: PUNTATORE A NODO
i: INT
continua: BOOL
risp . CHAR

INIZIO

```
Scrivi ("Vuoi inserire un nodo?")
Leggi (risp)
i ← 1
Continua ← FALSO
RIPETI
  SE (i = 1) AND (risp = 'S')
    ALLORA
      Alloca (pNew, DimensioneDi (NODO))    (1)
      SE (pNew ≠ NULL)
        ALLORA
          Leggi (pNew→Info)
          pCur ← pNew        (2)
          pTesta ← pCur      (3)
          continua ← VERO
        ALTRIMENTI
          Continua ← FALSO
      FINE SE
    ALTRIMENTI
      SE (risp = 'S')
        ALLORA
          Alloca (pNew, DimensioneDi (NODO))    (4)
            SE (pNew ≠ NULL)
              ALLORA
                Leggi (pNew→Info)
                (pCur→pNext) ← pNew    (5)
                pCur ← pNew        (6)
              ALTRIMENTI
                continua ← FALSO
            FINE SE
        FINE SE
  FINE SE
  SE (risp = 'S')
    ALLORA
     /* Richiesta successiva utente */
     Scrivi ("Vuoi inserire un nodo?")
     Leggi (risp)
     i ← i + 1
   FINE SE
FINCHE' (risp != 'S')
SE (continua = VERO)
  ALLORA
    (pCur → pNext) ← NULL
FINE SE
```

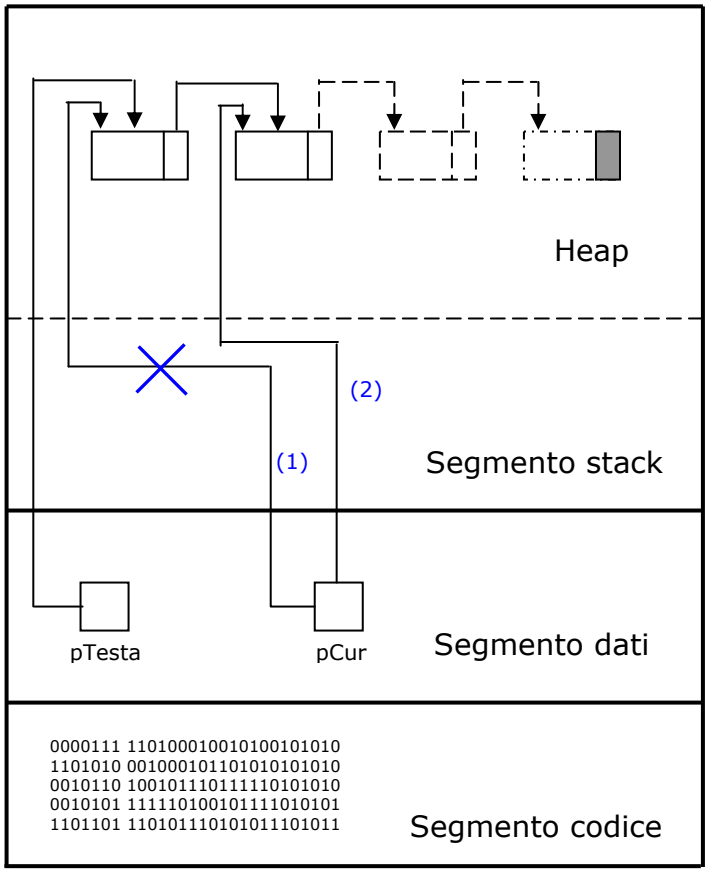


STAMPA LISTA

SE (continua = VERO)
ALLORA

```
/* visualizzazione della lista */
  pCur ← pTesta                    (1)
  MENTRE (pCur ≠ NULL) ESEGUI
    Scrivi (pCur → Info)
    pCur ← (pCur→ pNext)           (2)
  FINE MENTRE
```

```
/* visualizzazione alternativa della lista */
pCur ← pTesta
MENTRE (pCur ≠ NULL) ESEGUI
  Scrivi ((*pCur).Info)
  pCur ← ((*pCur).pNext)
FINE MENTRE
```



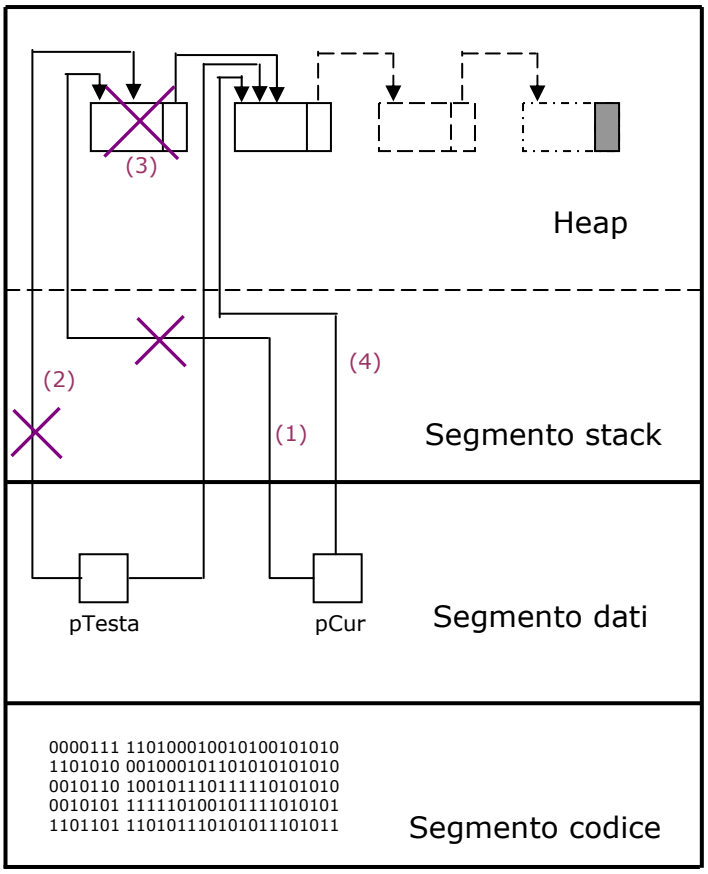
DEALLOCA LISTA

```
/* deallocazione della lista */
pCur ← pTesta                      (1)
MENTRE (pCur ≠ NULL) ESEGUI
  pTesta ← (pCur→pNext)            (2)
  Dealloca (pCur)                  (3)
  pCur ← pTesta                    (4)
FINE MENTRE
```

```
/* deallocazione alternativa della lista */
pCur ← pTesta
MENTRE (pCur ≠ NULL) ESEGUI
  pTesta ← ((*pCur).pNext)
  Dealloca (pCur)
  pCur ← pTesta
FINE MENTRE
```

FINE SE

FINE



RICERCA ELEMENTO IN UNA LISTA

pos : INT
ele: INT
trovato: BOOL
…….

SE (continua = VERO)
ALLORA

```
/* ricerca elemento in una lista */
  trovato ← FALSO
  Leggi (ele)
  i ← 0
  pos ← i
  pCur ← pTesta
  MENTRE (trovato = FALSO) AND (pCur != NULL) ESEGUI
    i ← i + 1
    SE (ele = (pCur→Info))
      ALLORA
        trovato = VERO
        pos = i
    FINE SE
    pCur ← (pCur → pNext)
  FINE MENTRE

  /* Gestione del messaggio all'utente */
  SE (trovato = VERO)
    ALLORA
      Scrivi ("Elemento trovato in posizione")
      Scrivi (pos)
    ALTRIMENTI
      Scrivi ("Elemento non trovato")
  FINE SE
```

FINE SE

FINE

RICERCA ELEMENTO IN UNA LISTA CON CONTEGGIO DELLE SUE OCCORRENZE

ele: INT
trovato. BOOL
…….

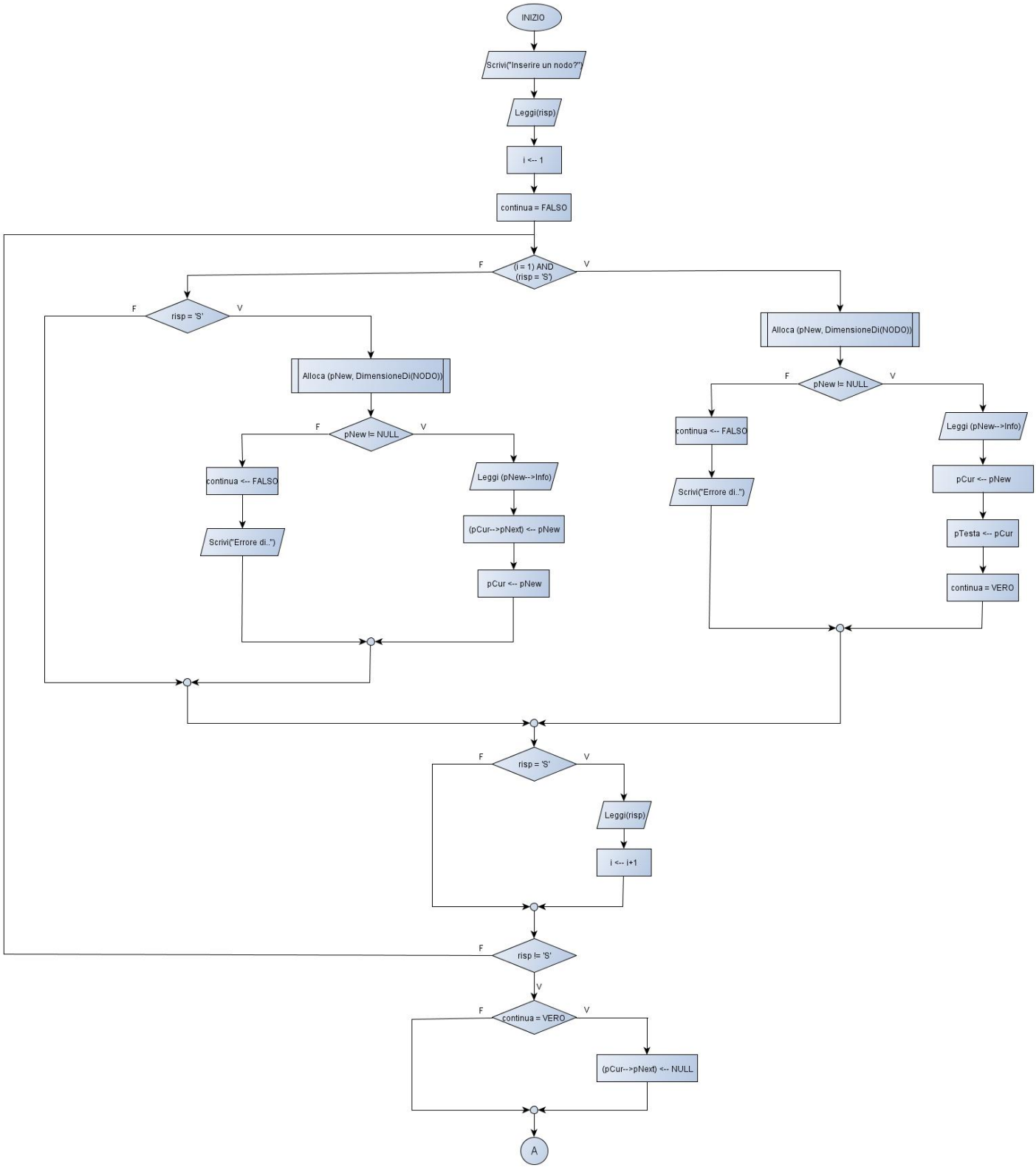
SE (continua = VERO)
ALLORA

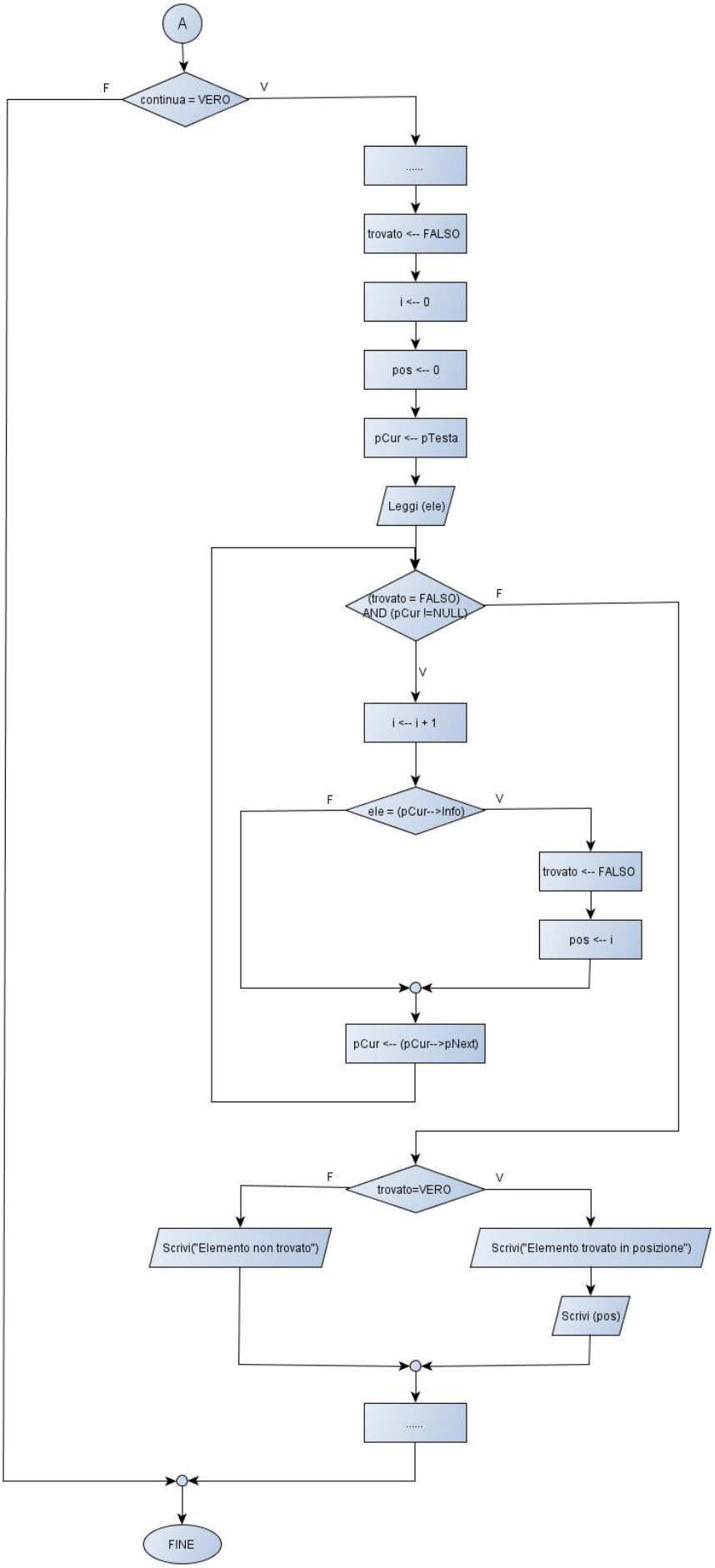
```
/* ricerca elemento in una lista con conteggio delle sue occorrenze
*/
  trovato ← FALSO
  Leggi (ele)
  i ← 0
  pCur ← pTesta
  MENTRE (pCur != NULL) ESEGUI
    i ← i + 1
    SE (ele = (pCur→Info))
      ALLORA
        trovato = VERO
        i ← i + 1
    FINE SE
    pCur ← (pCur → pNext)
  FINE MENTRE

  /* Gestione del messaggio all'utente */
  SE (trovato = VERO)
    ALLORA
      Scrivi ("Elemento trovato un numero di volte pari a ")
      Scrivi (i)
    ALTRIMENTI
      Scrivi ("Elemento non trovato")
  FINE SE
```

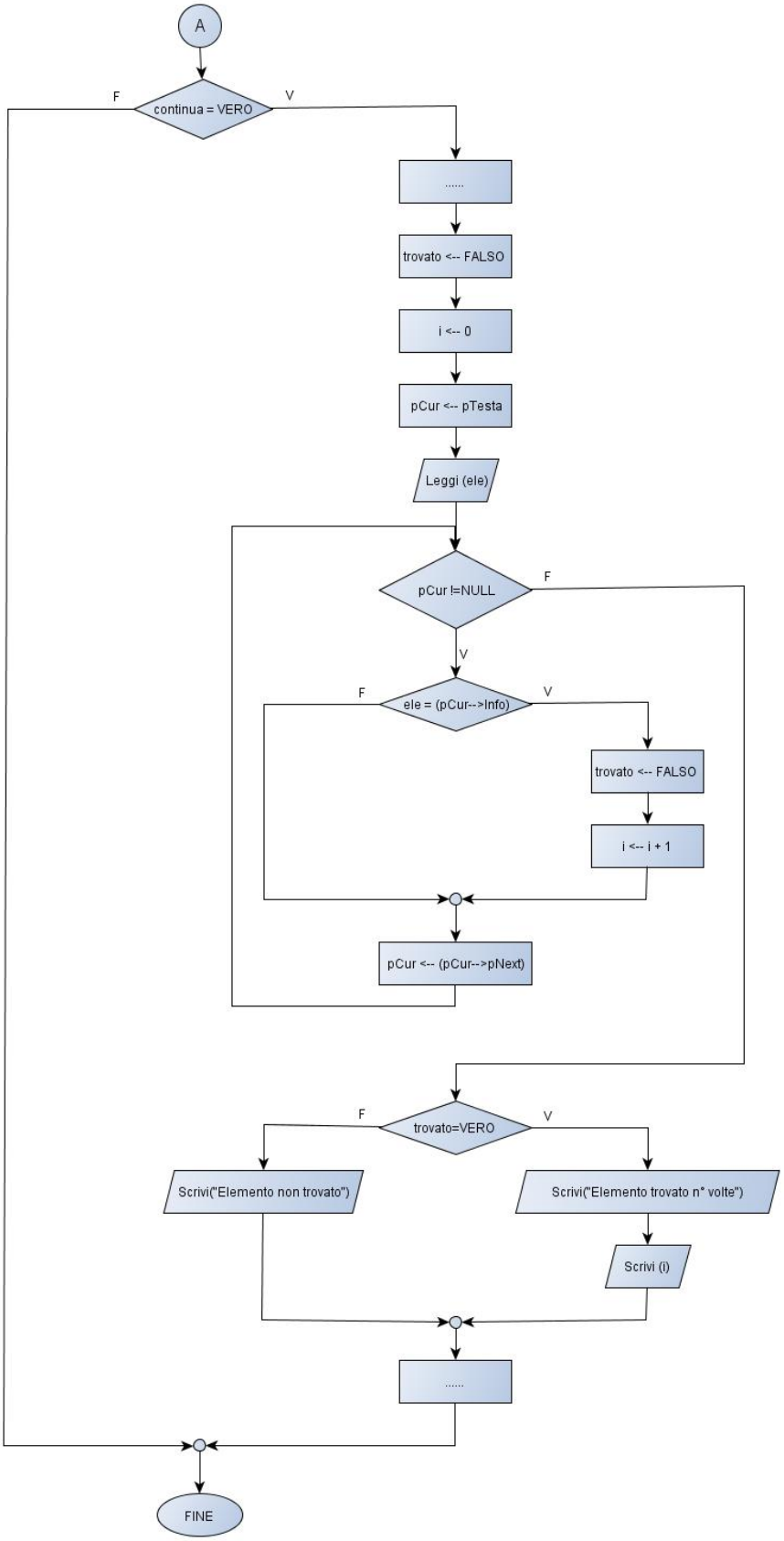
FINE SE

FINE

FLOW-CHART


INIZIO
Scrivi("Inserire un nodo?")
Leggi(risp)
i <-- 1
continua = FALSO
(i = 1) AND (risp = 'S')
F
V
risp = 'S'
F
V
Alloca (pNew, DimensioneDi(NODO))
pNew != NULL
F
V
continua <-- FALSO
Scrivi("Errore di..")
Leggi (pNew-->Info)
(pCur-->pNext) <-- pNew
pCur <-- pNew
Alloca (pNew, DimensioneDi(NODO))
pNew != NULL
F
V
continua <-- FALSO
Scrivi("Errore di..")
Leggi (pNew-->Info)
pCur <-- pNew
pTesta <-- pCur
continua = VERO
risp = 'S'
F
V
Leggi(risp)
i <-- i+1
risp != 'S'
F
V
continua = VERO
F
V
(pCur-->pNext) <-- NULL
A


A
continua = VERO
F
V
.....
trovato <-- FALSO
i <-- 0
pos <-- 0
pCur <-- pTesta
Leggi (ele)
(trovato = FALSO)
AND (pCur !=NULL)
F
V
i <-- i + 1
ele = (pCur-->Info)
F
V
trovato <-- FALSO
pos <-- i
pCur <-- (pCur-->pNext)
trovato=VERO
F
V
Scrivi("Elemento non trovato")
Scrivi("Elemento trovato in posizione")
Scrivi (pos)
.....
FINE


A
continua = VERO
F
V
.....
trovato <-- FALSO
i <-- 0
pCur <-- pTesta
Leggi (ele)
pCur !=NULL
F
V
ele = (pCur-->Info)
F
V
trovato <-- FALSO
i <-- i + 1
pCur <-- (pCur-->pNext)
trovato=VERO
F
V
Scrivi("Elemento non trovato")
Scrivi("Elemento trovato n° volte")
Scrivi (i)
.....
FINE