```
Algorithme Ex5
Début
   Remplir(t,n)
   Transférer1(t,n,tp,np)
   Transférer2(t,n,tp,np)
   Afficher(tp,np)
Fin

Procédure Remplir(@t:tab,@n:entier)
Début
  Répéter
    Écrire("n ?") ; Lire(n)
  Jusqu'à 5 ≤ n ≤ 40

  Pour i de 0 à n-1 Faire
    Répéter
      Écrire("t[", i, "] ?") ; Lire(t[i])
    Jusqu'à (t[i]>0) et (verif(t,i))
  Fin Pour
Fin
```

TDNT

TDNT
tab = tableau de 40 entier

TDOG

Objet	Type	
t, tp	tab	
n, np	entier	
Remplir		
Transférer1		procédure
Transférer2		
Afficher		

TDOL

Objet	Type
i	entier
verif	fonction

```
Fonction verif(t:tab, n:entier):booléen
Début
   tr ← Vrai
   Pour i de 0 à n-1 Faire
      Si t[i]=t[n] Alors
         tr ← Faux
      Fin si
   Fin Pour
   Retourner tr
Fin
```

TDOL

objet	Type
i	entier
tr	booléen

```
Procédure transférer1(t:tab, n:entier, @tp:tab, @np:entier)
Début
   np ← 0
   Pour i de 0 à n-1 Faire
      si palindrome(binaire(t[i])) Alors
         tp[np] ← t[i]
         np ← np+1
      Fin Si
   Fin Pour
Fin
```

TDOL

objet	Type
i	entier
palindrome	} fonction
binaire	

```
Procédure transférer2(t:tab, n:entier, @tp:tab, @np:entier)
Début
  Pour i de 0 à n-1 Faire
    si non palindrome(binaire(t[i])) et
       palindrome(Convch(t[i])) Alors
       tp[np] ← t[i]
       np ← np+1
    Fin Si
  Fin Pour
Fin
```

TDOL

objet	Type
i	entier
palindrome	} fonction
binaire	

```
Procédure Afficher(t:tab, n:entier)
Début
  Pour i de 0 à n-1 Faire
    b ← binaire(t[i])
    Écrire(t[i], "(10) = ", b, "(2) ->")
    Si palindrome(convch(t[i])) et
       palindrome(b) Alors
      Écrire("Doublement palindrome")
    Sinon Si palindrome(b) Alors
      Écrire("Palindrome binaire")
    Sinon
      Écrire("Palindrome")
    Fin Si
  Fin Pour
Fin
```

TDOL

objet	Type
i	entier
palindrome	} fonction
binaire	
b	chaîne

```
Fonction Palindrome(ch: chaine): booléen
Début
  Retourner (ch ≠ "") et
            (inverse(ch) = ch)
Fin
```

TDOL

objet	Type
inverse	fonction

```
Fonction inverse(ch: chaine): chaine
Début
  ch1 ← ""
  Pour i de 0 à Long(ch)-1 Faire
      ch1 ← ch[i] + ch
  Fin Pour
  Retourner ch1
Fin
```

TDOL

objet	Type
i	entier
ch1	chaîne

```
Fonction binaire(n: entier): chaine
Début
  res ← ""
  Tant Que n ≠ 0 Faire
    r ← n mod 2
    n ← n div 2
    res ← ConvCh(r) + res
  Fin Tant Que
  Retourner res
Fin
```

TDOL

objet	Type
r	entier
res	chaîne