X-PC-2011

```
> allouer:=proc(n)
  return Array(1..n);
  end:
> taille:=proc(t)
  return op(2,ArrayDims(t));
  end:
> estPermutation:=proc(t)
  local n,u,i;
  n:=taille(t);
  u:=allouer(n);
  for i from 1 to n do
           if t[i]>0 and t[i]<n+1 then
                       u[t[i]]:=1
           else return faux
           fi;
  od;
  for i from 1 to n do
           if u[i]=0 then
                       return faux
           fi;
  od;
  return vrai;
  end:
> t:=Array([1,1,2]);estPermutation(t);
                                                    t := [1, 1, 2]
                                                         faux
> composer:=proc(t,u)
  local comp,k,n;
  n:=taille(t);
  comp:=allouer(n);
  for k from 1 to n do
          comp[k]:=u[t[k]]
  od;
  return comp;
  end:
> u:=Array(1..3,[2,3,1]);t:=Array(1..3,[3,1,2]);print(composer(t,u));
                                                    u := [2, 3, 1]
                                                    t := [3, 1, 2]
                                                      [1, 2, 3]
> inverser:=proc(t)
  local inv,k,n;
  n:=taille(t);
  inv:=allouer(n);
  for k from 1 to n do
           inv[t[k]]:=k
  od;
  return inv;
  end:
> print(inverser(u));
                                                      [3, 1, 2]
> ident:=proc(a)
  local egal,k,n;
  n:=taille(a);
  egal:=vrai;
  k:=1;
  while egal=vrai and k<n do
          if a[k]<>k then
                  egal:=faux
          else
                  k:=k+1
          fi;
  od;
  return egal;
  end:
```

Variante avec boucle for et sortie par return :

```
> ident:=proc(a)
  local k,n;
  n:=taille(a);
  for k from 1 to n do
          if a[k]<>k then
                  return faux
          fi;
  od;
  return vrai;
  end:
> ordre:=proc(t)
  local ord,puissance,n;
  n:=taille(t);
  ord:=1;
  puissance:=t;
  while ident(puissance)=faux and ord<40 do
            puissance:=composer(puissance,t);
            ord:=ord+1
  od;
  return ord;
  end:
> u:=Array(1..3,[2,3,1]);ordre(u);v:=Array(1..3,[2,1,3]);ordre(v);
                                              u := [2, 3, 1]
                                                    3
                                              v := [2, 1, 3]
                                                    2
> periode:=proc(t,i)
  local period,candidat,n;
  n:=taille(t);
  period:=1;
  candidat:=t[i];
  while candidat<>i and period<40 do
           candidat:=t[candidat];
           period:=period+1
  od;
  return period;
  end;
```

```
periode := proc(t, i)
local period, candidat, n;
    n := taille(t);
    period := 1;
    candidat := t[i];
    while candidat ≠ i and period < 40 do candidat := t[candidat]; period := period + 1 end do;
    return period
end proc
```

```
> periode(u,1);periode(v,1),periode(v,3);
                                                    3
                                                   2, 1
> estDansOrbite:=proc(t,i,j)
  local period,itere,k,n;
  n:=taille(t);
  period:=periode(t,i);
  itere:=i;
  k:=0;
  while j<>itere and k<period do
            itere:=t[itere];
            k:=k+1
  od;
  if j<>itere then
```

```
          return faux
  else
          return vrai
  fi;
  end;
```

```
estDansOrbite := proc(t, i, j)
local period, itere, k, n;
    n := taille( t );
    period := periode( t, i );
    itere := i;
    k := 0;
    while j ≠ itere and k < period do itere := t[ itere ]; k := k + 1 end do;
    if j ≠ itere then return faux else return vrai end if
end proc
```

Variante avec boucle for :

```
> estDansOrbite:=proc(t,i,j)
  local period,itere,k,n;
  n:=taille(t);
  period:=periode(t,i);
  itere:=i;
  for k from 1 to period do
            if j=itere then
                     return vrai
            else
                     itere:=t[itere];
            fi;
  od;
  return faux
  end;
```

```
estDansOrbite := proc(t, i, j)
local period, itere, k, n;
    n := taille( t );
    period := periode( t, i );
    itere := i;
    for k to period do if j = itere then return vrai else itere := t[ itere ] end if end do;
    return faux
end proc
```

```
> estDansOrbite(u,2,1);
                                              vrai
```

```
> estTransposition:=proc(t)
  local tailleSupport,k,n;
  n:=taille(t);
  tailleSupport:=0;
  for k from 1 to n do
            if t[k]<>k then
                    tailleSupport:=tailleSupport+1
            fi;
  od;
  if tailleSupport=2 then
              return vrai
  else
              return faux
  fi;
  end;
```

```
estTransposition := proc(t)
local tailleSupport, k, n;
    n := taille( t );
    tailleSupport := 0;
    for k to n do if t[ k ] ≠ k then tailleSupport := tailleSupport + 1 end if end do;
    if tailleSupport = 2 then return vrai else return faux end if
end proc
```

```
> estTransposition(u);
                                              faux
```

```
> estTransposition(v);
                                         vrai
> estCycle:=proc(t)
  local kDansSupport,k,tailleSupport,n;
  n:=taille(t);
  tailleSupport:=0;
  kDansSupport:=-1;
  for k from 1 to n do
            if t[k]<>k then
                    tailleSupport:=tailleSupport+1;
                    kDansSupport:=k
            fi;
  od;
  if tailleSupport<>0 and periode(t,kDansSupport)=tailleSupport then
            return vrai
  else
            return faux
  fi;
  end;
estCycle := proc(t)
local kDansSupport, k, tailleSupport, n;
    n := taille(t);
    tailleSupport := 0;
    kDansSupport := -1;
    for k to n do if t[k] ≠ k then tailleSupport := tailleSupport + 1; kDansSupport := k end if end do;
    if tailleSupport ≠ 0 and periode(t, kDansSupport) = tailleSupport then return vrai else return faux end if
end proc
> estCycle(u);
                                         vrai
> estCycle(v);
                                         vrai
> w:=Array(1..4,[2,1,4,3]);estCycle(w);
                                    w := [2, 1, 4, 3]
                                         faux
>
```