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(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:
```

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(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(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(u);
```

vrai

```
> estCycle(v);
```

vrai

```
> w:=Array(1..4,[2,1,4,3]);estCycle(w);
```

w := [2, 1, 4, 3]

faux

```
> periodes:=proc(t)
  local p,k,j,i,pp,n;
  n:=taille(t);
  p:=allouer(n);
```

```
   for k from 1 to n do
         if p[k]=0 then
                pp:=periode(t,k);
                j:=k;
                for i from 1 to pp do
                       p[j]:=pp;
                       j:=t[j]
                od;
         fi;
   od;
   return p;
   end:
> periodes(u);periodes(w);
```

[3, 3, 3]

[2, 2, 2, 2]

```
> x:=Array(1..5,[2,5,4,3,1]);periodes(x);
```

x := [2, 5, 4, 3, 1]

[3, 3, 2, 2, 3]

```
> itererEfficace:=proc(t,k)
   local iteree,p,i,j,l,iter,n;
   n:=taille(t);
   iteree:=allouer(n);
   p:=periodes(t);
   for i from 1 to n do
          j:=irem(k,p[i]);
          iter:=i;
          for l from 1 to j do
                   iter:=t[iter]
          od;
          iteree[i]:=iter
   od;
   return iteree;
   end:
> itererEfficace(x,14);
```

[5, 1, 3, 4, 2]

x est une permutation d'ordre 6 et de taille 5

```
> pgcd:=proc(a,b)
   local reste;
   reste:=irem(a,b);
   if reste=0 then
          return b
   else
          return pgcd(b,reste)
   fi;
   end:
> pgcd(24,30);
```

6

```
> ppcm:=proc(a,b)
   return a*b/pgcd(a,b);
   end:
> ppcm(24,30);
```

120

```
> ordreEfficace:=proc(t)
   local p,ord,k,pk,i,n;
   n:=taille(t);
   p:=periodes(t);
   ord:=1;
   for k from 1 to n do
         if irem(ord,p[k])<>0 then
                ord:=ppcm(ord,p[k])
         fi;
   od;
```

```
  return ord;
  end:
> ordreEfficace(x);ordreEfficace(u);
                                                6
                                                3
>
```