```
######################## PROBLÈME 1: #########################
# 1)
# vp(x)
val := proc(X, p)
  local v, x ;
  v := 0 ; x := X ;
  while (irem(x, p, 'x') = 0) do v := v+1 ; od ;
  v ;
end:
# 'true' ssi N est pseudo-premier pour la base a
pp := proc(N, a)
  local t, e, q, b, i ;
  e := val(N - 1, 2) ;
  q := (N - 1) / 2^e ;
  b := a &^ q mod N ;
  if (b = 1) then RETURN(true) ; fi ;
  for i from 0 to e - 1 do
    if (b + 1 mod N = 0) then RETURN(true) ; fi ;
    b := b &^ 2 mod N ;
  od:
  false:
end:
# 4)
with(numtheory):
ischarm := proc(N)
  local L, p, e, z ;
  L := ifactors(N)[2] ;
  for z in L do
    p := z[1] ;
    e := z[2] ;
    if e > 1 or irem(N-1, p-1) > 0 then RETURN(false) ; fi ;
  od:
  evalb( nops(L) > 1 ) ; # évalue comme booléen
end:
charm := proc(N)
  local L, i ;
  L := NULL ;
  for i from 2 to N do
    if (ischarm(i)) then L := L, i ; fi ;
  od ;
```

```
  L ;
end:
# 5)
premier := proc(N)  # on suppose N ¿ 2
  local a, i ;
  a := rand(2..N-1) ;
  for i to 10 do
    if not(pp(N, a())) then RETURN(false) ; fi ;
  od ;
  "PEUT-ETRE" ;
end:
######################## PROBLÈME 2: #########################
# 3)
with(numtheory):
PL := proc(N)
  local res, L, p, e, z, A, a, certif ;
  if (N < 2) then RETURN(false) ; fi ;
  if (N = 2) then RETURN(true) ; fi ;
  A := rand(2..N-1) ;  # on suppose N ¿ 2
  L := ifactors(N - 1)[2] ;
  res := NULL ;
  for z in L do
    p := z[1] ; certif := PL(p) ;
    for i to 100 do # évite une boucle infinie si N non premier
      a := A() ;
      b := a &^ ((N - 1)/p) mod N ;
      if (b <> 1) then
        if (b &^ p mod N <> 1) then RETURN(false) ; fi ;
        break ;
      fi ;
    od ;
    if (i > 100) then RETURN("ECHEC") ; fi ;
    res := res, [p, a, certif]
  od:
  res ;
end:
```