```python
#X-MP-PC-2013
def admet_point_fixe(t):
    for i in range(n):
        if t[i]==i:
            return 'vrai'
    return 'faux'
def nb_points_fixes(t):
    nbre=0
    for i in range(n):
        if t[i]==i:
            nbre=nbre+1
    return nbre
def itere(t,x,k):
    iter=x
    for i in range(1,k-1):
        iter=t[iter]
    return iter
def nb_points_fixes_iteres(t,k):
    nbre=0
    for i in range(n):
        if itere(t,i,k)==i:
            nbre=nbre+1
    return nbre
def attire(t,i,j):
    j_itere=j
    for k in range(n):
        if j_itere==i:
            return 'vrai'
        else:
            j_itere=t[j_itere]
    return 'faux'
def est_attracteur(t,i):
    if t[i]!=i:
        return 'faux'
    else:
        for j in range(n):
            if attire(t,i,j)=='faux':
                return 'faux'
    return 'vrai'
def admet_attracteur_principal(t):
    for i in range(n):
        if est_attracteur(t,i)=='vrai':
            return 'vrai'
    return 'faux'
def temps_de_convergence(t,x):
    if t[x]==x:
        return 0
    else:
        return 1+temps_de_convergence(t,t[x])
def allouer():
    return range(n)
def temps_de_convergence_max(t):
    tc=allouer()
    for i in range(n):
        tc[i]=n+1
    #On met tc a 0 pour l'attracteur principal
    i=0
    while t[i]!=i:
        i=i+1
    tc[i]=0
    for i in range(n):
        if tc[i]==n+1: #On cherche le premier itere de i deja visite
            j1=i
            l=0
            while tc[j1]==n+1:
                j1=t[j1]
                l=l+1
```

```
            ll=l+tc[j1] #tc[i]
            j1=i
            for k in range(l):
                tc[j1]=ll-k
                j1=t[j1]
    #Recherche du max
    maxi=tc[0]
    for i in range(1,n):
        if tc[i]>maxi:
            maxi=tc[i]
    return maxi
def est_croissante(t):
    for i in range(n-1):
        if t[i+1]<t[i]:
            return 'faux'
    return 'vrai'
def chercher(t,a,b):
    if a==b:
        return a
    elif b==a+1:
        if t[a]==a:
            return a
        else:
            return b
    else:
        c=int((a+b)/2)
        if t[c]==c:
            return c
        elif t[c]>c:
            return chercher(t,c,b)
        else:
            return chercher(t,a,c)
def point_fixe(t):
    return chercher(t,0,n-1)
def pgcd_points_fixes(t):
    pt=1
    i=1
    while t[pt]!=pt:
        pt=t[pt]
    return pt
```