IPT Devoir 7- Proposition de solutions

Solution 1

```
def det(M):
    """ Developpment suivant la premiere ligne """
    if len(M)==1:
        return M[0,0]
    else:
        s=0
        n=len(M)
        for i in range(n):
            T=np.zeros((n-1,n-1))
            T[:,:i]=M[1:,:i]
            T[:,i:]=M[1:,i+1:]
            s=s+(-1)**(i)*det(T)*M[0,i]
        return s
```

Solution 2 Dans les trois cas il s'agit de tirer p éléments dans une liste E.

- Simulation 1 : tirage avec ordre et avec remise : f simule une p-liste de E.
- Simulation 2 : tirage avec ordre et sans remise car l'élément obtenu est retiré avant le tirage suivant : g simule un p-arrangement de E. On note l'instruction assert qui impose que $p \leq \text{Card}(E)$.
- Simulation 3 : tirage avec ordre et sans remise suivi d'un résultat ordonné ; l'ordre sur le résultat final neutralise l'ordre lors des tirages, le tirage devient donc sans ordre et sans répétition, tel un tirage simultané : h simule une p-combinaison de E.

Solution 3 Lancers d'une pièce

```
def longueurs(p):
    L1,r=1,rd.rand()<p
    s=rd.rand()<p
    while r==s:
        L1=L1+1
        s=rd.rand()<p
    L2=1
    r=rd.rand()<p
    while r==s:
        L2=L2+1
        r=rd.rand()<p
    return L1,L2
```

```
def series(n,p):
    L,r,l=[],rd.rand()<p,1
    for i in range(n):
        s=rd.rand()<p
        while r==s:
            l=l+1
            s=rd.rand()<p
        L.append(l)
        r,l=s,1
    return L
```

Solution 4 Intégration numérique

1. La méthode des rectangles consiste à subdiviser le segment d'intégration et à remplacer sur chaque subdivision la fonction par une constante. Ainsi, le calcul de l'intégrale se ramène à une somme d'aires de rectangles.

$$\int_a^b f(t)\mathrm{d}t \approx \frac{b-a}{n}\sum_{k=0}^{n-1} f(x_k) \text{ avec } x_j = a + j\,\frac{b-a}{n}$$

2. Le script est :

```
def arcsin(x,h=1e-5):
    if x<0:
        h=-h
    n=int(x/h)
    t,I=0,1
    for i in range(1,n):
        t=t+h
        I=I+1/np.sqrt(1-t**2)
    return I*h
```

Solution 5 Modèle d'une pile non bornée

```
def creer_file():
    return []

def defiler(f):
    assert len(f)>0,'file vide'
    return f.pop(0)

def enfiler(f,v):
    f.append(v)

def taille(f):
    return len(f)

def est_vide(f):
    return len(f)==0
```