```
'''
Chp1 : TP N°2 Les Piles
Groupe : Ipein/ST3/GB
Date : 21-09-2023
'''
# Pile
def creer_pile():
   #p = []# p = list()
   return []

def pile_vide(p):
   return len(p)==0 #p==[]

def sommet(p):
   return p[-1] # p[len(p)-1]

def taille(p): return len(p)
def empiler(p,x):
   p.append(x) # p += [x]

def depiler(p):
   s = p.pop(-1)
   return s
#
p=creer_pile()
try:
   s = depiler(p)
except IndexError :
   print("Pile vide, pas de sommet")

# python

def depiler1(p):
   if taille(p):
       return p.pop()
   else:
       raise Exception("pile vide")
try:
   s = depiler1(p)
except:
   print("pile vide")
else:
   x  = s + 1

def depiler(p):
   som = p.pop()
   return som

sommet = depiler(p)

# ex1
def conversion(n):
   p = creer_pile()
   if n==0: empiler(p,0) #return '0b0'
   while n != 0 : # not ! equal =
       n,r = divmod(n,2)
       empiler(p,r)

   result = 'Ob'
   while taille(p)>0:
       result += str(depiler(p))

   return result
#ex 2
def verif_parentheses(ch):
   # ch : expr arithmétique
   L = []
   p = creer_pile()
   for i in range(len(ch)):
       if ch[i] == '(':
           empiler(p,i)
       elif ch[i] == ')':
           if pile_vide(p):
               return False
           else:
               x = depiler(p)
               L+= [(x,i)]
   if pile_vide(p):
       return L
   return False

def calc_expr_arith(ch):
   p = creer_pile()
   for c in ch:
       if c.isdigit():
           empiler(p,c)
       else:
           #mth1
           n1,n2 = int(depiler(p)),int(depiler(p))
           if c== '+': r = n1 + n2
           elif c== '-' : r = n2 - n1
           elif c=='*' : r = n1 * n2
           elif c=='/': r = n2/n1
           empiler(p,r)
           # mth2
           n1,n2 = depiler(p), depiler(p)
           empiler(p,eval(n2 + c + n1))

   return sommet(p)

# Ex 3
def permut_cir(p,n):
   p1 = creer_pile()
   p2 = creer_pile()

   for i in range(n):
       empiler(p1, depiler(p))
   for i in range(taille(p)):
       empiler(p2, depiler(p))

   #vider p1 dans p
   while taille(p1)>0:
       empiler(p,depiler(p1))

   # vider p2 dans p
   while not pile_vide(p2):
       empiler(p,depiler(p2))

# prog principal
p = creer_pile()
empiler(p,1);empiler(p,2);empiler(p,3)
permut_cir(p,2)

#Ex 4
# https://anis-saied.github.io/ipein
def somme(p):
   if pile_vide(p): return 0
   else:
       s = depiler(p)
       if type(s) != list: # s : entier
           return s + somme(p)
       else:
           return somme(s) + somme(p)
```