```python
"""
Montecarlo para problemas de conteo.
(c) Carlos M. martinez, mayo 2022
"""

import random
import math
import time
from scipy.stats import norm
import functools
from cm2c.fing.mmc.utils import sortearPuntoRN
from pathos.multiprocessing import ProcessPool as Pool

_VERSION = "Problemas de conteo MMC v0.1.1 - Carlos Martinez mayo 2022"


def version():
    return _VERSION
# end def

def MonteCarlo_Conteo(n, r, delta, SortearEnX, PerteneceAS1):
    """
    Entradas:

    n: tamaño muestra
    delta: 1-delta es el intervalo de confianza
    SortearEnX es una funcion que sortea un elemento de X
    PerteneceAS1 es una funcion que devuelve true si un elemento de X pertenece a S1

    Salidas:
        Zest : estimador del conteo
        VZest : estimador de la desviacion estandard del estimador
        (I1, I2) : intervalo de confianza según Agresti - Coull

    OJO: Esta version asume que F se compone de un único subjconjunto S1
    PerteneceA deberia ser en el futuro un ARRAY de funciones, una para cada subconjunto.
    """

    S = 0

    for i in range(0,n):
        a = SortearEnX()
        if PerteneceAS1(a):
            S = S + 1
        # endif
    # endfor
    Zest = round( r * S / n )
    VZest = round( Zest * (r - Zest) / (n - 1) )
    (tI1, tI2) = Agresti_Coull(S, n, delta)
    (I1, I2) = (r*tI1, r*tI2)

    return (Zest, math.sqrt(VZest), (I1, I2), (I2-I1)/2, S )
# end def

# def Agresti_Coull(S, n, delta, r):
# esto aplica asi nomas ? o depende de r
def Agresti_Coull(S, n, delta):
```

```
58      """
59      Intervalo de confianza segun Agresti Coull.
60      Parámetros:
61        - S: estimador, cantidad de puntos que caen dentro del volumen
62        - n: cantidad de replicas (puntos sorteados)
63        - delta: margen, si el intervalo de conf es 95%, entonces delta = 0.05
64      """
65      kappa = norm.ppf(1-delta/2)
66
67      Xg = S + kappa**2/2
68      ng = n + kappa**2
69
70      pg = Xg / ng
71      qg = 1 - pg
72
73      disc = kappa * math.sqrt(pg*qg)*( 1/math.sqrt(ng))
74
75      return (pg-disc, pg+disc)
76  # end def
```