Algoritmos T.P. N° 1 - Parte 2 - Juan Cruz Ambrosini

from algo1 import Array

import random

*#------------------------------------------------------------------------------------------------------------------*

def check\_duplicates (arreglo):

    for i in range (0, len(arreglo)):

        for j in range (i + 1, len(arreglo)):

            if arreglo[i] == arreglo [j]:

                return True

    return False

def create\_Set(arreglo):

    tamano\_max = len(arreglo)

    conjunto = Array(tamano\_max, 0)

    tamano\_conjunto = 0

    if len(arreglo) == 0:

        return Array(0)

    for i in range (0, tamano\_max):

        es\_repetido = False

        for j in range (0,tamano\_conjunto):

            if arreglo[i] == conjunto[j]:

                es\_repetido = True

                break

        if not es\_repetido:

            conjunto[tamano\_conjunto] = arreglo[i]

            tamano\_conjunto += 1

    resultado = Array(tamano\_conjunto, 0)

    for i in range(0, tamano\_conjunto):

        resultado[i] = conjunto[i]

    return resultado

def union(arreglo\_1, arreglo\_2):

    elementos\_repetidos\_1 = check\_duplicates(arreglo\_1)

    elementos\_repetidos\_2 = check\_duplicates(arreglo\_2)

    arreglo\_union = Array(len(arreglo\_1) + len(arreglo\_2), 0)

    if not elementos\_repetidos\_1 and not elementos\_repetidos\_2:

        for i in range (0, len(arreglo\_1)):

            arreglo\_union[i] = arreglo\_1[i]

        for i in range (len(arreglo\_1), len(arreglo\_1) + len (arreglo\_2)):

            arreglo\_union[i] = arreglo\_2[i - len(arreglo\_1)]

        resultado = create\_Set(arreglo\_union)

        return resultado

    else:

        print("Hay elementos repetidos en los conjuntos, no es posible operar.")

        return None

def intersection (s, t):

    elementos\_repetidos\_1 = check\_duplicates(s)

    elementos\_repetidos\_2 = check\_duplicates(t)

    conjunto\_interseccion = Array (max(len(s), len(t)), 0)

    tamano\_interseccion = 0

    if not elementos\_repetidos\_1 and not elementos\_repetidos\_2:

        for i in range (0, len(s)):

            for j in range(0, len(t)):

                if s[i] == t[j]:

                    conjunto\_interseccion[tamano\_interseccion] = s[i]

                    tamano\_interseccion += 1

        resultado = Array(tamano\_interseccion,0)

        for i in range (0, tamano\_interseccion):

            for j in range (i, len(conjunto\_interseccion)):

                if conjunto\_interseccion[j] != None:

                    resultado [i] = conjunto\_interseccion[j]

                    break

        if len(resultado) == 0:

            print("Hay elementos repetidos en los conjuntos, no es posible operar.")

            return None

        return resultado

    else:

        print("Hay elementos repetidos en los conjuntos, no es posible operar.")

        return None

def difference(s, t):

    elementos\_repetidos\_1 = check\_duplicates(s)

    elementos\_repetidos\_2 = check\_duplicates(t)

    diferencial\_dimension = 0

    contador\_aux = 0

    conjunto\_diferencia = Array(len(s), 0)

    conjunto\_interseccion = intersection(s, t)

    if not elementos\_repetidos\_1 and not elementos\_repetidos\_2:

        if (conjunto\_interseccion == None or len(conjunto\_interseccion) == 0) :

            return s

        else:

            for i in range (0, len(s)):

                for j in range (0, len(conjunto\_interseccion)):

                    if conjunto\_interseccion[j] == s[i]:

                        conjunto\_diferencia[i] = None

                        break

                    else:

                        conjunto\_diferencia[i] = s[i]

        for i in range (0, len(conjunto\_diferencia)):

            if conjunto\_diferencia[i] == None:

                diferencial\_dimension += 1

        resultado = Array(len(conjunto\_diferencia) - diferencial\_dimension, 0)

        for i in range (0, len(conjunto\_diferencia)):

            if conjunto\_diferencia[i] != None:

                resultado[contador\_aux] = conjunto\_diferencia[i]

                contador\_aux += 1

        return resultado

    else:

        return None

*#------------------------------------------------------------------------------------------------------------------*

arreglo\_1 = Array(10,0)

arreglo\_2 = Array(10,0)

for i in range (0, 10):

    arreglo\_1[i] = random.randint(0,20)

    arreglo\_2[i] = random.randint(0,20)

print("-----------------------------------------------------------------------------")

print("Arreglo Aleatorio Numero 1: ")

print("P = ", arreglo\_1)

print("-----------------------------------------------------------------------------")

print("Arreglo Aleatorio Numero 2: ")

print("Q = ", arreglo\_2)

print("--------------------------------------- Ahora los convertiremos en conjuntos ---------------------------------------")

print("Conjunto Numero 1: ")

conjunto1 = create\_Set(arreglo\_1)

print("A = ", conjunto1)

print("-----------------------------------------------------------------------------")

print("Conjunto Numero 2: ")

conjunto2 = create\_Set(arreglo\_2)

print("B = ", conjunto2)

print("--------------------------------------- Ahora uniremos estos conjuntos ---------------------------------------")

print("A U B = ", union(conjunto1, conjunto2))

print("Prueba con los arreglos crudos: ")

print(union(arreglo\_1, arreglo\_2))

print("--------------------------------------- Veamos la interseccion de ambos ---------------------------------------")

print("Veamos la interseccion de ambos: ")

print("A ∩ B = ", intersection(conjunto1, conjunto2))

print("Prueba con los arreglos crudos: ")

print(intersection(arreglo\_1, arreglo\_2))

print("--------------------------------------- Finalmente, le restaremos el segundo conjunto al primero ---------------------------------------")

print("A - B = ", difference(conjunto1, conjunto2))

print("Prueba con los arreglos crudos: ")

print(difference(arreglo\_1, arreglo\_2))

