Guía de Ejercicios Prácticos - Ficha 12

Sitio: Universidad Virtual UTN FRC
Curso: Algoritmos y Estructuras de Datos (2020)
Libro: Guía de Ejercicios Prácticos - Ficha 12
Imprimido por: Luciana Lisette Montarce
Día: lunes, 23 de noviembre de 2020, 21:23

FILADD.COM

Este archivo fue descargado de https://filadd.com

Descripción

Esta guía contiene enunciados de algunos ejercicios para aplicar los conceptos de programación en ***Python*** que se analizan en la ***Ficha 12***. Los alumnos no deben subir nada al aula virtual: la guía se propone como fuente de ejercicios generales. Se sugiere intentar resolver cada uno de estos problemas ya sea trabajando solos o en grupos de estudio, y cuando las soluciones se publiquen, controlar lo hecho con las sugerencias propuestas por sus docentes. Utilice el foro del curso para plantear dudas y consultas, que cualquier alumno puede intentar responder.

FILADD.COM

Este archivo fue descargado de https://filadd.com

Tabla de contenidos

FILADD.COM

Este archivo fue descargado de https://filadd.com

1. Pluviómetro

Se ha solicitado un programa que permita cargar las precipitaciones promedio en cada mes del país, en base a esos datos armar un menú de opciones que permita:

1. Determinar el promedio anual de lluvias
2. Determinar el promedio de lluvias para un determinado trimestre
3. Determinar el mes más seco del año
4. Determinar los meses del año en los que llovió más que el promedios de lluvia de todo el año.

FILADD.COM

Este archivo fue descargado de https://filadd.com

1.1. Solución

FILADD.COM

Este archivo fue descargado de https://filadd.com

```
__author__ = 'Catedra de Algoritmos y Estructuras de Datos'
import random


def validar_rango(mensage='Ingrese un numero: '):
    numero = 0
    while numero < 1 or numero > 4:
        numero = int(input(mensage))
        if numero < 1 or numero > 4:
            print('Valor incorrecto!!! El valor debe estar comprendio entre 1 y 4 incluidos.')
    return numero


def determinar_rango_trimestre(trimestre):
    if trimestre == 1:
        return range(3)
    elif trimestre == 2:
        return range(3, 6)
    elif trimestre == 3:
        return range(6, 9)
    else:
        return range(9, 12)


def determinar_promedio_trimestre(lluvias):
    trimestre = validar_rango('Ingrese el trimestre a calcular: ')
    suma = 0
    rango = determinar_rango_trimestre(trimestre)
    for i in rango:
        suma += lluvias[i]

    return trimestre, suma / len(rango)


def promedio_anual(lluvias):
    suma = 0
    tam = len(lluvias)

    for i in range(tam):
        suma += lluvias[i]

    return suma / tam


def buscar_mes_menor_lluvia(lluvias):
    menor = ()
    tam = len(lluvias)

    for m in range(tam):
        if m == 0:
            menor = m, lluvias[m]
        else:
            if menor[1] > lluvias[m]:
                menor = m, lluvias[m]
    return menor[0]


def principal():

    menu = 'Menu de Opciones \n' \
           '================================= \n' \
           '1 \t Determinar el promedio anual de lluvias \n' \
           '2 \t Determinar el promedio de lluvias para un determinado trimestre \n' \
           '3 \t Determinar el mes más seco del mes \n' \
           '0 \t Salir \n '\
           'Ingrese su opcion: '
    opcion = -1
```

FILADD.COM

Este archivo fue descargado de https://filadd.com

```
    lluvias = list()
    for i in range(12):
        lluvias.append(random.uniform(45.8, 160.1))

    while opcion != 0:
        opcion = int(input(menu))
        if opcion == 1:
            prom = promedio_anual(lluvias)
            print('El promedio anual de lluvias en el pais fue', prom, 'mm')
        elif opcion == 2:
            trimestre = determinar_promedio_trimestre(lluvias)
            print('El promedio de lluvias del trimestre', trimestre[0], 'fue de', trimestre[1], 'mm')
        elif opcion == 3:
            menor = buscar_mes_menor_lluvia(lluvias)
            print('El mes con menor lluvia registrada fue', menor + 1)


if __name__ == '__main__':
    principal()
```

FILADD.COM

Este archivo fue descargado de https://filadd.com

2. Último y Primero

Desarrollar un programa que permita cargar por teclado un vector de n elementos y luego:

- Informe cuántas veces se repite en el vector el último número ingresado
- Genere un nuevo vector, conteniendo sólo los elementos menores al primer valor ingresado.

FILADD.COM

Este archivo fue descargado de https://filadd.com

2.1. arreglos.py

```
__author__ = 'Cátedra de AED'

def validar_tamanio():
    n = int(input('Ingrese el tamaño del vector: '))
    while n<= 0:
        n = int(input('El tamaño debe ser positivo. Ingrese otro: '))
    return n

def crear(tam):
    v = []
    for i in range(tam):
        dato = int(input('Ingrese v[' + str(i) + ']: '))
        v.append(dato)
    return v

def contar_repeticiones(v):
    repeticiones = 0
    for elemento in v:
        if elemento == v[-1]:
            repeticiones += 1
    return repeticiones

def buscar_menores(v):
    menores = list()
    for elemento in v:
        if elemento < v[0]:
            menores.append(elemento)
    return menores
```

FILADD.COM

Este archivo fue descargado de https://filadd.com

2.2. test.py

```
__author__ = 'Cátedra de AED'

import arreglos

def test():
    print('ARREGLOS: ÚLTIMO Y PRIMERO')
    print('*'*80)
    n = arreglos.validar_tamanio()
    v = arreglos.crear(n)
    print('*'*80)
    rep = arreglos.contar_repeticiones(v)
    print('El último número se repite',rep,'veces en el vector')
    menores = arreglos.buscar_menores(v)
    print('Los valores menores al primer elemento son:',menores)

test()
```

FILADD.COM

Este archivo fue descargado de https://filadd.com

3. Busqueda de primos

Desarrollar un programa que permita generar un arreglo de n elementos, a partir del arreglo:

1. Generar un segundo vector con todos aquellos números primos
2. Determinar el promedio del vector generado en el punto 1

FILADD.COM

Este archivo fue descargado de https://filadd.com

3.1. funciones.py

```
__author__ = 'Catedra de Algoritmos y Estructuras de Datos'
import math


def validar_mayor_cero(mensage='Ingrese un numero: '):
    numero = 0
    while numero <= 0:
        numero = int(input(mensage))
        if numero <= 0:
            print('Numero incorrecto!!! El numero debe ser mayor a cero')
    return numero


def es_primo(numero):
    primo = True
    if numero % 2 == 0:
        primo = False

    else:
        tope = math.ceil(math.sqrt(numero))
        for i in range(3, tope):
            if numero % i == 0:
                primo = False
                break
    return primo
```

FILADD.COM

Este archivo fue descargado de https://filadd.com

3.2. arreglo.py

```
__author__ = 'Catedra de Algoritmos y Estructuras de Datos'
from modulos.funciones import es_primo


def crear(tam):
    vec = list()
    for i in range(tam):
        dato = int(input('Ingese el valor para la posicio vec[' + str(i) + ']'))
        vec.append(dato)
    return vec


def generar_vector_primos(vec):
    primos = []
    for item in vec:
        if es_primo(item):
            primos.append(item)
    return primos


def promedio(vec):
    tam = len(vec)
    if tam > 0:
        suma = 0
        for item in vec:
            suma += item
        return round(suma / tam, 2)
    else:
        return 0

def mostar(vec):
    tam = len(vec)
    cadena = '['
    for pos in range(tam):
        if pos == (tam - 1):
            cadena += str(vec[pos])
        else:
            cadena += str(vec[pos]) + ' - '

    cadena += ']'
    return cadena
```

FILADD.COM

Este archivo fue descargado de https://filadd.com

3.3. principal.py

```
__author__ = 'Catedra de Algoritmos y Estructuras de Datos'
from modulos import arreglo
from modulos.funciones import validar_mayor_cero


def principal():
    print('Manejo de Arreglos')
    print('==================================')

    tam = validar_mayor_cero('Ingrese el tamanio del vector: ')
    vector = arreglo.crear(tam)
    print(arreglo.mostar(vector))

    primos = arreglo.generar_vector_primos(vector)
    print(arreglo.mostar(primos))

    prom = arreglo.promedio(primos)
    print('Su promedio es: ', prom)


if __name__ == '__main__':
    principal()
```

FILADD.COM

Este archivo fue descargado de https://filadd.com

4. Mayores con el mismo índice

Cargar por teclado dos vectores de tamaño n y, a partir de ellos, generar un tercer vector que contenga, para cada componente, el mayor valor entre las componentes homólogas (mismo índice) de los otros dos vectores.

Por ejemplo, si se cargan los siguientes vectores a y b:

a = [3, 4, 6]

b = [8, 5, 1]

El resultado sería:

c = [8, 5, 6]

FILADD.COM

Este archivo fue descargado de https://filadd.com

4.1. arreglos.py

```
__author__ = 'Cátedra de AED'

def validar_tamanio():
    n = int(input('Ingrese el tamaño del vector: '))
    while n<= 0:
        n = int(input('El tamaño debe ser positivo. Ingrese otro: '))
    return n

def cargar(tam):
    v = []
    for i in range(tam):
        dato = int(input('Ingrese v[' + str(i) + ']: '))
        v.append(dato)
    return v

def elegir_mayores(a,b):
    c = [0] * len(a)
    for i in range(len(a)):
        if a[i] > b[i]:
            c[i] = a[i]
        else:
            c[i] = b[i]
    return c
```

FILADD.COM

Este archivo fue descargado de https://filadd.com

4.2. test.py

```
__author__ = 'Cátedra de AED'

import arreglos

def test():
    print('ARREGLOS: MAYORES CON EL MISMO ÍNDICE')
    print('*'*80)
    n = arreglos.validar_tamanio()
    print('\nCargar el primer vector')
    a = arreglos.cargar(n)
    print('\nCargar el segundo vector')
    b = arreglos.cargar(n)
    print('*'*80)
    print('a =',a)
    print('b =',b)
    c = arreglos.elegir_mayores(a,b)
    print('c =',c)

test()
```

FILADD.COM

Este archivo fue descargado de https://filadd.com

5. Mayores al promedio

Ingresar por teclado un arreglo unidimensional de números enteros de tamaño n, siendo n una variable que se ingresa por teclado. Se pide:

- Calcular el valor promedio entre los números ingresados en el vector.
- Determinar la cantidad de números del vector que son mayores al promedio.

FILADD.COM

Este archivo fue descargado de https://filadd.com

5.1. soporte.py

```
__author__ = 'Cátedra de Algoritmos y Estructuras de Datos'

def validar_mayor_que(inf, mensaje):
    n = inf
    while n <= inf:
        n = int(input(mensaje))
        if n <= inf:
            print("Valor incorrecto!")
    return n


def read(vec):
    for i in range(len(vec)):
        vec[i] = int(input("Ingrese el elemento vec["+str(i)+"]:"))

def write(vec):
    for i in range(len(vec)):
        print("Elemento", i, ":", vec[i])

def promedio(vec):
    prom = 0
    cant = len(vec)
    acu = 0
    for i in range(cant):
        acu += vec[i]
    prom = acu / cant
    return prom

def contar_mayores(vec, prom):
    cont = 0
    for i in range(len(vec)):
        if vec[i] > prom:
            cont += 1
    return cont
```

FILADD.COM

Este archivo fue descargado de https://filadd.com

5.2. principal

```
__author__ = 'Algoritmos y Estructuras de Datos'

from soporte import *

def procesar():
    n = validar_mayor_que(0, "Ingrese el tamaño del vector: ")
    vec = n * [0]
    read(vec)

    prom = promedio(vec)

    cont = contar_mayores(vec, prom)

    print("Promedio de valores:", prom)
    print("Cantidad de valores mayores al promedio:", cont)

if __name__ == "__main__":
    procesar()
```

FILADD.COM

Este archivo fue descargado de https://filadd.com