Prueba 3 Aduana

September 10, 2020

1 Prueba 3 DGA (Elias Preza)

1.1 Reordenar (String Scramble)

Haga que la función StringScramble (str1, str2) tome ambos parámetros que se pasan y devuelva la cadena verdadera si una parte de los caracteres str1 se puede reorganizar para que coincida con str2; de lo contrario, devuelva la cadena falsa. Por ejemplo: si str1 es "rkqodlw" y str2 es "world", la salida debería devolver verdadero. No se introducirán signos de puntuación ni símbolos con los parámetros.

Ejemplos:

Entrada: "cdore" & str2 = "codificador" Salida: verdadero Entrada: "h3llko" & str2 = "hola" Salida: falso

[55]:
```
import re

def StringScramble(str1, str2):
    list_str1 = list(str1)
    list_str1.sort()
    list_str2 = list(str2)
    list_str2.sort()

    return (list_str2 == list_str1)

print(StringScramble('ahlo','hola'))
print(StringScramble('agots','gato'))
```

```
True
False
```

1.2 Comprobador de Fibonacci (Fibonacci Checker)

Comprobador de Fibonacci Haga que la función FibonacciChecker (num) devuelva la cadena sí si el número dado es parte de la secuencia de Fibonacci. Esta secuencia se define por: Fn = Fn-1 + Fn-2, lo que significa que para encontrar Fn se suman los dos números anteriores. Los dos primeros números son 0 y 1, luego viene 1, 2, 3, 5, etc. Si num no está en la secuencia de Fibonacci, devuelve la cadena no.

Ejemplos

Entrada: 34 Salida: si Entrada: 54 Salida: no Examinar Res

[61]:
```
import math

#---Serie fibonacci

def fibonacci (n):
    if n < 2:
        return n
    return fibonacci (n-1) + fibonacci (n-2)

for x in range(20):
    print(fibonacci(x))

#--Comprobación

def isPerfectSquare(x):
   s = int(math.sqrt(x))
   return s*s == x

def isFibonacci(n):

   return isPerfectSquare(5*n*n + 4) or isPerfectSquare(5*n*n - 4)
for i in range(1,22):
   if (isFibonacci(i) == True):
      print (i,"si es un número Fibonacci")
   else:
      print (i,"no es un número Fibonacci")
```

```
0
1
1
2
3
5
8
13
21
34
55
89
144
233
377
610
987
1597
2584
4181
```

```
1 si es un número Fibonacci
2 si es un número Fibonacci
3 si es un número Fibonacci
4 no es un número Fibonacci
5 si es un número Fibonacci
6 no es un número Fibonacci
7 no es un número Fibonacci
8 si es un número Fibonacci
9 no es un número Fibonacci
10 no es un número Fibonacci
11 no es un número Fibonacci
12 no es un número Fibonacci
13 si es un número Fibonacci
14 no es un número Fibonacci
15 no es un número Fibonacci
16 no es un número Fibonacci
17 no es un número Fibonacci
18 no es un número Fibonacci
19 no es un número Fibonacci
20 no es un número Fibonacci
21 si es un número Fibonacci
```

1.3 Selector de Stock (Stock Picker)

Selector de stock Haga que la función StockPicker (arr) tome la matriz de números almacenados en arr que contendrá números enteros que representan la cantidad en dólares que vale una sola acción, y devuelva la ganancia máxima que podría haberse obtenido comprando acciones el día xy vendiendo stock el día y donde y> x. Por ejemplo: si arr es [44, 30, 24, 32, 35, 30, 40, 38, 15], su programa debería devolver 16 porque en el índice 2 las acciones valían $ 24, y en el índice 6 las acciones valían $ 40 , por lo que si compró las acciones a 24 y las vendió a 40, habría obtenido una ganancia de $ 16, que es la ganancia máxima que podría haberse obtenido con esta lista de precios de acciones.

Si no hay ganancias que podrían haberse obtenido con los precios de las acciones, entonces su programa debería devolver -1. Por ejemplo: arr es [10, 9, 8, 2], entonces su programa debería devolver -1.

Ejemplos

Entrada: [10,12,4,5,9] Salida: 5 Entrada: [14,20,4,12,5,11] Salida: 8 Vistazo

[65]:
```python
def StockPicker(arr):
    bp=0
    sp=0
    mp=-1
    for r in arr:
        if arr[arr.index(r)+1] > r:
            bp=r
            break
    sp=max(arr[_] for _ in range(arr.index(bp),len(arr)))
    if sp-bp > mp:
```

```
        return sp-bp
    else:
        return mp
print(StockPicker([44, 30, 24, 32, 35, 30, 40, 38, 15,100]))
print(StockPicker([10,12,4,5,9]))
print(StockPicker([14,20,4,12,5,11]))
```

```
76
2
6
```

1.4 Rotación de Matriz (Array Rotation (arr))

Haga que la función Array Rotation (arr) tome el parámetro arr que se está pasando, que será una matriz de enteros no negativos y gire circularmente la matriz comenzando desde el elemento N, donde N es igual al primer número entero en la matriz. Por ejemplo: si arr es [2, 3, 4, 1, 6, 10], entonces su programa debe rotar la matriz comenzando desde la segunda posición porque el primer elemento en la matriz es 2. Por lo tanto, la matriz final será [4, 1, 6, 10, 2, 3], y su programa debería devolver la nueva matriz como una cadena, por lo que para este ejemplo su programa devolvería 4161023. El primer elemento de la matriz siempre será un número entero mayor o igual a 0 y menor que el tamaño de la matriz.

Ejemplos

Entrada: [3,2,1,6] Salida: 6321 Entrada: [4,3,4,3,1,2] Salida: 124343

[82]:
```
def leftRotate(arr, d, n):
    for i in range(d):
        leftRotatebyOne(arr, n)


def leftRotatebyOne(arr, n):
    temp = arr[0]
    for i in range(n-1):
        arr[i] = arr[i + 1]
    arr[n-1] = temp



def printArray(arr, size):
    for i in range(size):
        print ("% d"% arr[i], end =" ")


arr = [1,2,3,4,5,6,7,8,9]
leftRotate(arr, 2, 9)
printArray(arr, 9)

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

```
arr = [3,2,1,6]
leftRotate(arr, 3, 4)
printArray(arr, 4)

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

arr = [4,3,4,3,1,2]
leftRotate(arr, 4, 6)
printArray(arr, 6)
```

```
3  4  5  6  7  8  9  1  2 -----------
6  3  2  1 -----------
```

[ ]: