

08 - STRINGS

Jorge Muñoz
IIC1103 – Introducción a la Programación

ITERAR CARÁCTER A CARÁCTER

Código de antes

```
for var in str :
```

Bloque del FOR

Código de después

Frase? Hola Mundo!
Empezamos a iterar
El caracter es ...
H
El caracter es ...
o
El caracter es ...
l
El caracter es ...
a
El caracter es ...

El caracter es ...
M
El caracter es ...
u
El caracter es ...
n
El caracter es ...
d
El caracter es ...
o
El caracter es ...
!
Hemos acabado de iterar

```
frase = input("Frase? ")  
print("Empezamos a iterar")  
  
for c in frase:  
    print("El caracter es ...")  
    print(c)  
  
print("Hemos acabado de iterar")
```



- Hello World
 - https://en.wikipedia.org/wiki/%22Hello,_World!%22_program
- Hello World en centenares de idiomas
 - <http://helloworldcollection.de/>



¿TE ACUERDAS DE FOR IN RANGE?



¡VOLVIÓ! ... EN
FORMA DE FOR IN STR

ADN (FOR)



INTERESES

“bioinformática”

“medicina, ciencias de la salud, biología”



1. A DNA sequence contains four bases linked together in a single strand.

2. Complementary bases join together following base pairing rules.

3. Complementary bases bond together, forming a double-stranded molecule of DNA.

DNA Bases

adenine	A	T	thymine
guanine	G	C	cytosine

© 2006 Encyclopædia Britannica, Inc.

ADN: AACAAATCGAGTCACAT

Base (A,T,G,C): T

El numero de T es 3





I'LL WAIT
FOR YOU HERE



1. A DNA sequence contains four bases linked together in a single strand.

2. Complementary bases join together following base pairing rules.

3. Complementary bases bond together, forming a double-stranded molecule of DNA.

DNA Bases

adenine	A	T	thymine
guanine	G	C	cytosine

© 2006 Encyclopædia Britannica, Inc.

ADN: AACAAATCGAGTCACAT

Base (A,T,G,C): T

El numero de T es 3



```
#Entrada
adn = input("ADN: ")
b = input("Base (A,T,G,C): ")

#Para cada letra mirar si es igual
num = 0
    
```



```
#Salida
print("El numero de", b, "es", num)
```



```
#Entrada
adn = input("ADN: ")
b = input("Base (A,T,G,C): ")

#Para cada letra mirar si es igual
num = 0
for c in adn:
    if c == b:
        num += 1

#Salida
print("El numero de", b, "es", num)
```

Jorge, ... perdona que *interrumpa* ... pero es que lo
de hacer **funciones** no me quedó 100% claro.
¿Crees que podrías repasarlo?

SÍ CLARO



```
#Entrada
adn = input("ADN: ")

#Para cada letra mirar si es igual A
numA = 0
for c in adn:
    if c == "A":
        numA += 1

#Para cada letra mirar si es igual T
numT = 0
for c in adn:
    if c == "T":
        numT += 1

#Para cada letra mirar si es igual G
numG = 0
for c in adn:
    if c == "G":
        numG += 1

#Para cada letra mirar si es igual C
numC = 0
for c in adn:
    if c == "C":
        numC += 1

#Salida
print("Hay", numA, "As", numT, "Ts", numG, "Gs", numC, "Cs")
```

ADN: AAATGGC
Hay 3 As 1 Ts 2 Gs 1 Cs



```
def contar(adn,b):  
    pass  
  
#CÓDIGO PRINCIPAL  
adn = input("ADN: ")  
  
numA = contar(adn,"A")  
numT = contar(adn,"T")  
numG = contar(adn,"G")  
numC = contar(adn,"C")  
  
print("Hay", numA, "As", numT, "Ts", numG, "Gs", numC, "Cs")
```

ADN: AAATGGC
Hay 3 As 1 Ts 2 Gs 1 Cs





I'LL WAIT
FOR YOU HERE



```
def contar(adn,b):  
    pass  
  
#CÓDIGO PRINCIPAL  
adn = input("ADN: ")  
  
numA = contar(adn,"A")  
numT = contar(adn,"T")  
numG = contar(adn,"G")  
numC = contar(adn,"C")  
  
print("Hay", numA, "As", numT, "Ts", numG, "Gs", numC, "Cs")
```

ADN: AAATGGC
Hay 3 As 1 Ts 2 Gs 1 Cs



```
def contar(adn,b):  
    num = 0  
    for c in adn:  
        if c == b:  
            num += 1  
    return num
```

#CÓDIGO PRINCIPAL
adn = input("ADN: ")

numA = contar(adn,"A")
numT = contar(adn,"T")
numG = contar(adn,"G")
numC = contar(adn,"C")

print("Hay",numA, "As", numT, "Ts", numG, "Gs", numC, "Cs")

ADN: AAATGGC
Hay 3 As 1 Ts 2 Gs 1 Cs



- Sequence Alignment Methods
 - Dynamic Programming, Metaheuristics, ...
 - https://en.wikipedia.org/wiki/Sequence_alignment
- Sequence Alignment with Dynamic Programming
 - <http://theautomatic.net/2018/11/28/how-to-measure-dna-similarity-with-python-and-dynamic-programming/>

Edit View Colors Translation Table Instructions About DNATagger

A. DNA alignment

Q9FPK4
Q9FPK3
Q94E57
Q6XC94
Q6D4B5
Q43549
Q4VPJ1
Q84L47
Q4VPI3

ATGGGTGTTTCA	G	C	G	A	T	C	T	G	C	G
AGCTACGCGATGAGGCCACC	T	C	G	T	T	G	C	T	G	C
TCGTTATCCTCCGGCCTA	G	G	T	T	G	T	C	T	G	C
GGCTCTTCAAGTCCCTTGCT	C	C	C	C	C	C	C	C	C	C
CGCTAGATGCCGAAACCTCATT	G	G	G	G	G	G	G	G	G	G
AACTCTGCTCCAGGCTTCTT	A	A	A	A	A	A	A	A	A	A
TGGTCTTCTGCTTCTT	T	T	T	T	T	T	T	T	T	T
ATGGGTGTTTCA	G	C	G	A	T	C	T	G	C	G
ATGGGTGTTTCA	T	C	G	A	T	C	T	G	C	G
AGCTACGCGATGAGGCCACC	G	C	G	A	T	C	T	G	C	G
TCGTTATCCTCCGGCCTA	T	C	G	T	T	G	T	C	G	C
GGCTCTTCAAGTCCCTTGCT	C	C	C	C	C	C	C	C	C	C
CGCTAGATGCCGAAACCTCATT	G	G	G	G	G	G	G	G	G	G
AACTCTGCTCCAGGCTTCTT	A	A	A	A	A	A	A	A	A	A
TGGTCTTCTGCTTCTT	T	T	T	T	T	T	T	T	T	T

B. Back-translation from protein alignment

Q9FPK4
Q9FPK3
Q94E57
Q6XC94
Q6D4B5
Q43549
Q4VPJ1
Q84L47
Q4VPI3

ATGGGTGTTTCA	G	C	G	A	T	C	T	G	C	G
AGCTACGCGATGAGGCCACC	T	C	G	T	T	G	C	T	G	C
TCGTTATCCTCCGGCCTA	G	G	T	T	G	T	C	T	G	C
GGCTCTTCAAGTCCCTTGCT	C	C	C	C	C	C	C	C	C	C
CGCTAGATGCCGAAACCTCATT	G	G	G	G	G	G	G	G	G	G
AACTCTGCTCCAGGCTTCTT	A	A	A	A	A	A	A	A	A	A
TGGTCTTCTGCTTCTT	G	C	G	A	T	C	T	G	C	G
ATGGGTGTTTCA	T	C	G	A	T	C	T	G	C	G
ATGGGTGTTTCA	G	C	G	A	T	C	T	G	C	G
AGCTACGCGATGAGGCCACC	T	C	G	T	T	G	C	T	G	C
TCGTTATCCTCCGGCCTA	G	G	T	T	G	T	C	T	G	C
GGCTCTTCAAGTCCCTTGCT	C	C	C	C	C	C	C	C	C	C
CGCTAGATGCCGAAACCTCATT	G	G	G	G	G	G	G	G	G	G
AACTCTGCTCCAGGCTTCTT	A	A	A	A	A	A	A	A	A	A
TGGTCTTCTGCTTCTT	T	T	T	T	T	T	T	T	T	T

		G	C	C	C	T	A	G	C	G
G	0	1	1	1	1	1	1	1	1	1
C	0	1	2	2	2	2	2	2	2	2
G	0	1	2	2	2	2	2	3	3	3
C	0	1	2	3	3	3	3	3	4	4
A	0	1	2	3	3	3	3	4	4	4
A	0	1	2	3	3	3	3	4	4	4
T	0	1	2	3	3	4	4	4	4	4
G	0	1	2	3	3	4	4	5	5	5

LONGITUD Y ACCEDER A POSICIÓN

```
x = len("hola")  
print(x)
```

4

```
y = len("hola mundo")  
print(y)
```

10
22

```
z = len("Un limon. Medio limon.")  
print(z)
```

```
s = "hola.mundo"  
  
p = s[0]  
print(p)  
  
q = s[3]  
print(q)  
  
t = s[4]  
print(t)  
  
u = len(s) - 1 #Ultima posicion  
r = s[u]  
print(r)
```

h
a
.
o

ADN (FOR IN RANGE / WHILE)



```
#Usando for in range
def contar_fr(adn,b):
```

#CODIGO PRINCIPAL

```
adn = input("ADN: ")
```

```
numA = contar_fr(adn, "A")
```

```
numT = contar_fr(adn, "T")
```

```
numG = contar_fr(adn, "G")
```

```
numC = contar_fr(adn, "C")
```

```
print("Hay", numA, "As", numT, "Ts", numG, "Gs", numC, "Cs")
```

ADN: AAATGGC
Hay 3 As 1 Ts 2 Gs 1 Cs





I'LL WAIT
FOR YOU HERE



```
#Usando for in range
def contar_fr(adn,b):
```

#CODIGO PRINCIPAL

```
adn = input("ADN: ")
```

```
numA = contar_fr(adn,"A")
```

```
numT = contar_fr(adn,"T")
```

```
numG = contar_fr(adn,"G")
```

```
numC = contar_fr(adn,"C")
```

```
print("Hay",numA, "As", numT, "Ts", numG, "Gs", numC, "Cs")
```

ADN: AAATGGC
Hay 3 As 1 Ts 2 Gs 1 Cs



```
#Usando for in range
def contar_fr(adn,b):
    num = 0
    for i in range(0,len(adn)):
        c = adn[i]
        if c == b:
            num += 1
    return num

#CODIGO PRINCIPAL
adn = input("ADN: ")

numA = contar_fr(adn,"A")
numT = contar_fr(adn,"T")
numG = contar_fr(adn,"G")
numC = contar_fr(adn,"C")

print("Hay",numA, "As", numT, "Ts", numG, "Gs", numC, "Cs")
```

ADN: AAATGGC
Hay 3 As 1 Ts 2 Gs 1 Cs



#Usando While

```
def contar_w(adn,b):
```

#CODIGO PRINCIPAL

```
adn = input("ADN: ")
```

```
numA = contar_w(adn, "A")
numT = contar_w(adn, "T")
numG = contar_w(adn, "G")
numC = contar_w(adn, "C")
```

```
print("Hay", numA, "As", numT, "Ts", numG, "Gs", numC, "Cs")
```

ADN: AAATGGC
Hay 3 As 1 Ts 2 Gs 1 Cs





I'LL WAIT
FOR YOU HERE



#Usando While

```
def contar_w(adn,b):
```

#CODIGO PRINCIPAL

```
adn = input("ADN: ")
```

```
numA = contar_w(adn, "A")
numT = contar_w(adn, "T")
numG = contar_w(adn, "G")
numC = contar_w(adn, "C")
```

```
print("Hay", numA, "As", numT, "Ts", numG, "Gs", numC, "Cs")
```

ADN: AAATGGC
Hay 3 As 1 Ts 2 Gs 1 Cs



#Usando While

```
def contar_w(adn,b):
    num = 0
    i = 0
    while i < len(adn):
        c = adn[i]
        if c == b:
            num += 1
        i += 1
    return num
```

#CODIGO PRINCIPAL

```
adn = input("ADN: ")
```

```
numA = contar_w(adn, "A")
numT = contar_w(adn, "T")
numG = contar_w(adn, "G")
numC = contar_w(adn, "C")
```

```
print("Hay", numA, "As", numT, "Ts", numG, "Gs", numC, "Cs")
```

ADN: AAATGGC
Hay 3 As 1 Ts 2 Gs 1 Cs



```
#Usando for in str
def contar(adn,b):
    num = 0
    for c in adn:
        if c == b:
            num += 1
    return num
```

#Usando While

```
def contar_w(adn,b):
    num = 0
    i = 0
    while i < len(adn):
        c = adn[i]
        if c == b:
            num += 1
        i += 1
    return num
```

#Usando for in range

```
def contar_fr(adn,b):
    num = 0
    for i in range(0,len(adn)):
        c = adn[i]
        if c == b:
            num += 1
    return num
```

CONTENIDO



```
s = "hola mundo!"  
  
#Contiene el str  
a = "m" in s  
print(a)  
  
#No contiene el str  
b = "j" in s  
print(b)  
  
True  
False  
True  
False  
True  
True  
True  
True  
  
#Cualquier str (no solo letras)  
c = "!" in s  
print(c)  
  
#Mayusculas y Minusculas son diferentes  
d = "M" in s  
print(d)  
  
# Funciona para muchas de una letra  
e = "mun" in s  
print(e)  
  
# Es un str (puede tener espacios, etc)  
f = "la mu" in s  
print(f)  
  
#Se usa como cualquier bool  
g = not ("j" in s)  
print(g)
```

C*≠#**!!



INTERESES

“Problemas de la vida de un catalán viviendo en Barcelona”

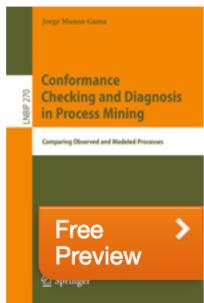
Search

[Home](#) [Subjects](#) [Services](#) [Springer Shop](#) [About us](#)

+++ Read While You Wait - Get immediate ebook access* when you order a print book +++

» Computer Science » Information Systems and Applications

Lecture Notes in Business Information Processing



© 2016

Conformance Checking and Diagnosis in Process Mining

Comparing Observed and Modeled Processes

Authors: **Munoz-Gama, Jorge**

Revised version of a PhD dissertation in process mining

[» see more benefits](#)

Buy this book

▼ eBook

46,00 €

price for Spain (gross)

[Buy eBook](#)

- ISBN 978-3-319-49451-7
- Digitally watermarked, DRM-free
- Included format: PDF
- ebooks can be used on all reading devices
- Immediate eBook download after purchase

► Softcover

57,19 €



[» FAQ](#) [» Policy](#)



3.3. Related Work

enforce the documentation of processes, while quality certification, such as the ISO 9000 standards, requires the documentation and monitoring of all processes to ensure their effectiveness. The use of overviews is a common technique to facilitate certification audits.

of the system.

In conclusion, the need for achieving precise results is evident in both carahuevo systems, for both carahuevo conformance checking and certification.

3.3 Related Work

Conformance checking is a process of comparing a process model with an event log to determine if the log is conformant to the model. Medeiros et al. [59] defines a metric – behavioral coverage – as the percentage of traces in the event log covered by the process model. To measure the precision between two models and a log, evaluating how much of the first model behavior is covered by the second. This measure is used within the *Genetic Miner* [60, 58] – a discovery approach based on evolutionary algorithms – to evaluate the quality of the population obtained. Goedertier et al. [46] introduces the use of artificial negative examples to measure the precision between an event log and a process model. The way of generating those negative examples was later improved by De Weerdt et al. [97] and Vandenhove et al. [27]. Finally, Van Dongen et al. [40, 39] addresses the precision dimension between two models without a log, based on the similarity of their structures.

However, given this chapter goal of checking precision between a process model and an event log, Rozinat et al. [80] can be seen as the seminal work, later extended in [79]. In [80], Rozinat et al. present several metrics to estimate the four dimensions of conformance



```
>>>  
Texto: Esta es micarahuevo ...  
Palabra: carahuevo  
ALERTA: palabra en texto  
>>>  
>>>  
Texto: Esta esoveuharac mi tesis ...  
Palabra: carahuevo  
ALERTA: inverso en texto  
>>>  
>>>  
Texto: Estaoveuharac es carahuevo mi tesis ...  
Palabra: carahuevo  
ALERTA: palabra e inverso en texto  
>>>  
>>>  
Texto: Esta es mi tesis ...  
Palabra: carahuevo  
Texto LIMPIO  
>>>
```





I'LL WAIT
FOR YOU HERE



```
>>>  
Texto: Esta es micarahuevo ...  
Palabra: carahuevo  
ALERTA: palabra en texto  
>>>  
>>>  
Texto: Esta esoveuharac mi tesis ...  
Palabra: carahuevo  
ALERTA: inverso en texto  
>>>  
>>>  
Texto: Estaoveuharac es carahuevo mi tesis ...  
Palabra: carahuevo  
ALERTA: palabra e inverso en texto  
>>>  
>>>  
Texto: Esta es mi tesis ...  
Palabra: carahuevo  
Texto LIMPIO  
>>>
```



#Entrada

#Crear palabra inversa

#Ver si la palabra y/o inversa estan



```
#Entrada
t = input("Texto: ")
p = input("Palabra: ")

#Crear palabra inversa
inv = [x for x in t[::-1] if x != " "]

#Ver si la palabra y/o inversa estan
if (p in t) and (inv in t):
    print("ALERTA: palabra e inverso en texto")
elif (p in t) and not (inv in t):
    print("ALERTA: palabra en texto")
elif not (p in t) and (inv in t):
    print("ALERTA: inverso en texto")
else:
    print("Texto LIMPIO")
```



```
#Entrada
t = input("Texto: ")
p = input("Palabra: ")

#Crear palabra inversa
inv = ""
for c in p:
    inv = c + inv

#Ver si la palabra y/o inversa estan
if (p in t) and (inv in t):
    print("ALERTA: palabra e inverso en texto")
elif (p in t) and not (inv in t):
    print("ALERTA: palabra en texto")
elif not (p in t) and (inv in t):
    print("ALERTA: inverso en texto")
else:
    print("Texto LIMPIO")
```



- Thesis: Conformance Checking and Diagnosis in **Process Mining**
 - <https://www.tesisenred.net/handle/10803/284964>
- Libro Springer
 - <https://www.springer.com/gp/book/9783319494500>

SLICING Y CONCATENAR

```
f = "hola mundo!"
```

```
#Slicing de pos a pos  
#Primero incluido, ultimo no  
a = f[2:7]  
print(a)
```

```
#Slicing des del principio a pos  
b = f[:7]  
print(b)
```

```
#Slicing des de pos al final  
c = f[6:]  
print(c)
```

```
#Slicing des del principio al final  
#Util si quieres hacer una copia de un str  
d = f[:]  
print(d)
```

la mu
hola mu
ndo!
hola mundo!

```
a = "ber"  
b = "tu"  
c = "cu"  
d = "lo"  
  
x = b+a+c+d  
print(x)
```

tuberculo

MUTABILIDAD

```
a = "Barça"  
a[0] = "F"  
a[3] = "$"  
print(a)
```

¿Que imprime?

```
Traceback (most recent call last):  
  File "/Users/jmunoz/Dropbox/Lectures/IIC1103/IIC1103-6 2016-2 Jorge/material/S6  
-Strings/farsa.py", line 8, in <module>  
    a[0] = "F"  
TypeError: 'str' object does not support item assignment
```

Los Strings son immutables

(Si se quiere cambiar, se crea un nuevo String)

Far\$a

```
a = "Barça"  
a = "F"+a[1:3]+"$"+a[4:]  
print(a)
```

SQL INJECTION

TEMAS

“...seguridad informática...”

“Asuntos cotidianos”

“... cyberseguridad...”

“evitar que me haceen”



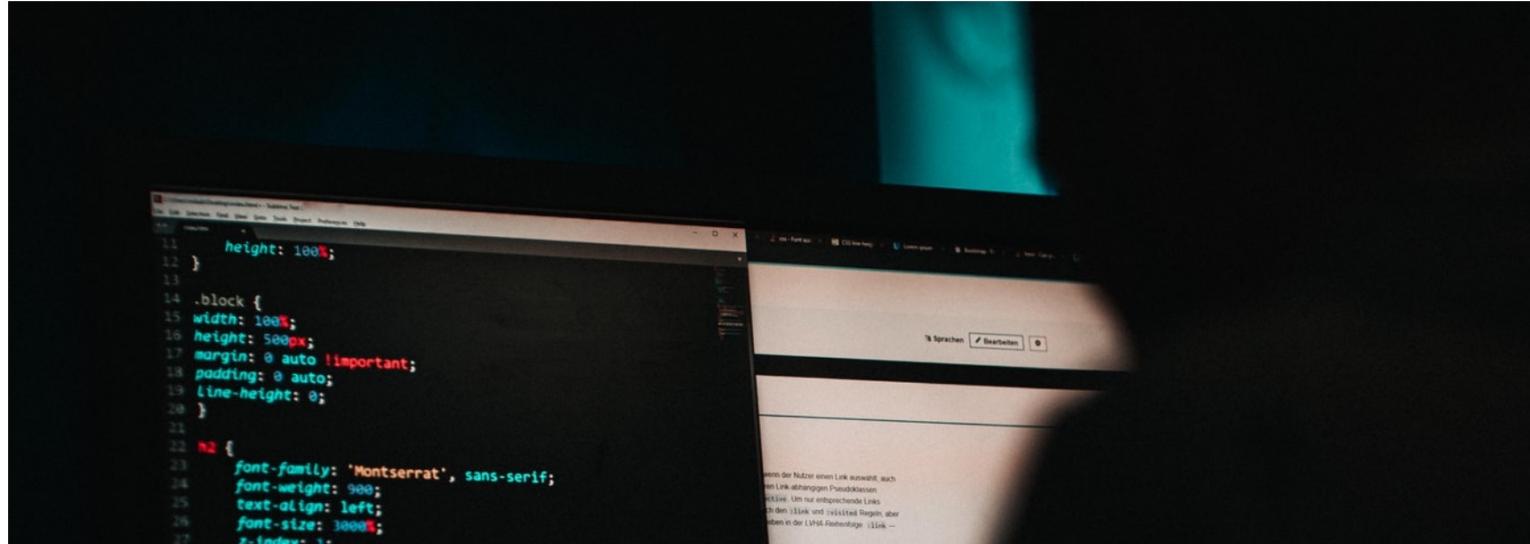


Home » Cybersecurity » SBN News » SQL Injection loses #1 spot as most dangerous attack technique

SQL Injection loses #1 spot as most dangerous attack technique



by Filip Truta on December 4, 2019





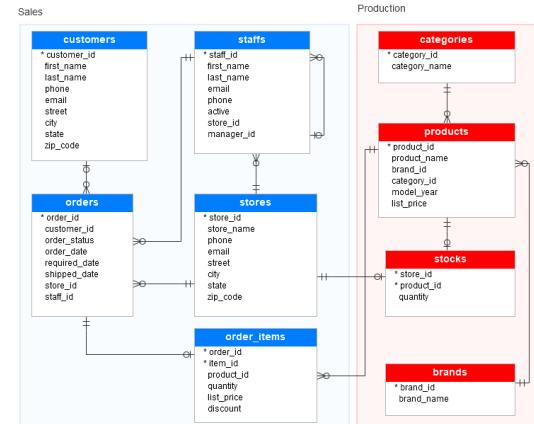
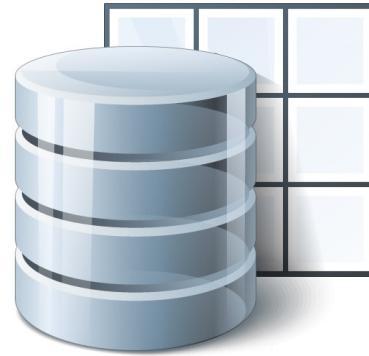
Situación
Real

Tablas Hojas de Calculo (e.g., Excel)

A	B	C
HORARIOS		
CURSO	CODIGO	HORARIO
Introducción a la Programación (S2)	IIC1103-2	M-J:1 M:5,6
Introducción a la Programación (S4)	IIC1103-4	M-J:2 M:5,6
Computación: Ciencia y Tecnología	IIC1105	M-J:2 W:6
Programación Avanzada (S1)	IIC2233-1	J:4,5 M:4

ESTUDIANTE	NOTA
Alicia	6.4
Bob	3.5
Carlos	5.5
David	4.2
Eva	6.8

Base de Datos (e.g., PostgreSQL)





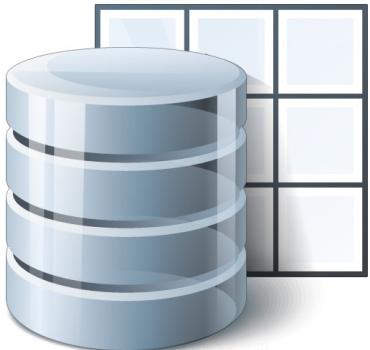
fx |

	A	B	C
1	HORARIOS		
3	CURSO	CODIGO	HORARIO
4	Introducción a la Programación (S2)	IIC1103-2	M-J:1 M:5,6
5	Introducción a la Programación (S4)	IIC1103-4	M-J:2 M:5,6
6	Computación: Ciencia y Tecnología	IIC1105	M-J:2 W:6
7	Programación Avanzada (S1)	IIC2233-1	J:4,5 M:4
8			
o			

Find in sheet ^ v : x

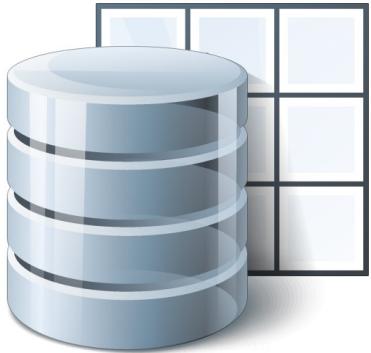
SQL

SELECT “cursos con clase los martes”



SELECT “cursos con clase los martes pero solo los módulos de la mañana”

SELECT “cursos con clase los martes pero que no tengan ayudantía los viernes”

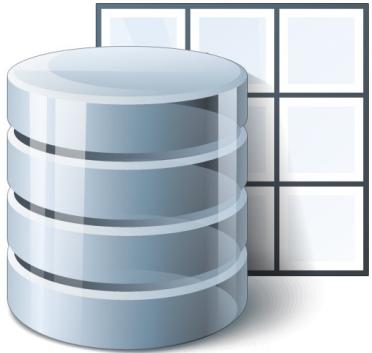


SQL

SELECT “cursos con clase los martes”

UPDATE “el curso IIC1103-2 ahora tiene
ayudantías los Jueves y no los martes”

DROP TABLE “tabla horarios”



SQL

```
SELECT "blabla" ; SELECT "blabla" ; SELECT "blabla"
```



> Búsqueda de Cursos

Semestre:	2020 Primer Semestre
Sigla:	IIC1103
NRC:	<input type="text"/>
Nombre Curso:	<input type="text"/>
Categoría:	-- Todos --
Profesor:	<input type="text"/>
Campus:	-- Todos --
Escuela:	-- Todas --
Horario:	Al menos este horario de... <input type="button" value="Todos"/> <input type="button"/>

Como una variable X

Lo guardas en una variable y luego usas la variable en el SELECT. En este caso X vale IIC1103

SELECT "X"

(SELECT "IIC1103")



¿Qué pasa si escribimos algo, y luego un ";" y luego algo más?

Búsqueda de Cursos

Semestre:	2020 Primer Semestre
Sigla:	IIC1103";UPDATE nota
NRC:	<input type="text"/>
Nombre Curso:	<input type="text"/>
Categoría:	-- Todos --
Profesor:	<input type="text"/>
Campus:	-- Todos --
Escuela:	-- Todas --
Horario:	Al menos este horario de... Todos

IIC1103";UPDATE "TablaNotasIntro" fila "Bob" nota "4.0

SELECT "X"

SELECT "IIC1103";UPDATE "TablaNotasIntro" fila "Bob" nota "4.0"

SELECT "IIC1103";DROP TABLE "TablaNotasIntro"



CONSIDERAR SIEMPRE AL USUARIO COMO UN POTENCIAL ATACANTE

JAMAS USEIS DIRECTAMENTE ALGO QUE OS DEN

SANEA LOS INPUTS DEL USUARIO

HOLA, LLAMO DE LA ESCUELA DE SU HIJO. ESTAMOS TENIENDO CIERTOS PROBLEMAS INFORMÁTICOS.



VAYA POR DIOS, ¿HA ROTO ALGO? EN CIERTA MANERA...



¿REALMENTE LE PUSO A SU HIJO EL NOMBRE DE Robert'); DROP TABLE Students;...?

OH, SÍ.
LE LLAMAMOS PEQUEÑO BOBBY TABLAS.



BIEN, HEMOS PERDIDO TODOS LOS REGISTROS ESTUDIANTILES DE ESTE AÑO. ESPERO QUE ESTÉ CONTENTA.



Y YO ESPERO QUE HAYAN APRENDIDO A SANEAR LA INSERCIÓN DE SUS BASES DE DATOS.



Función que recibe un str y retorna un str igual menos los ;

Instrucción? SELECT blablabla
SUCIO: SELECT blablabla
LIMPIO: SELECT blablabla

Instrucción? SELECT bla ; DROP TABLE bla
SUCIO: SELECT bla ; DROP TABLE bla
LIMPIO: SELECT bla DROP TABLE bla

Instrucción? SELECT bla;SELECT blabla
SUCIO: SELECT bla;SELECT blabla
LIMPIO: SELECT blaSELECT blabla

CONSEJO: Siempre que se pueda crear un string nuevo con las cosas que quereis en vez de intentar sacar las cosas que no quereis al str





I'LL WAIT
FOR YOU HERE



Función que recibe un str y retorna un str igual menos los ;

Instrucción? SELECT blablabla
SUCIO: SELECT blablabla
LIMPIO: SELECT blablabla

Instrucción? SELECT bla ; DROP TABLE bla
SUCIO: SELECT bla ; DROP TABLE bla
LIMPIO: SELECT bla DROP TABLE bla

Instrucción? SELECT bla;SELECT blabla
SUCIO: SELECT bla;SELECT blabla
LIMPIO: SELECT blaSELECT blabla

CONSEJO: Siempre que se pueda crear un string nuevo con las cosas que quereis en vez de intentar sacar las cosas que no quereis al str



```
def sanitize(s):
```

```
#CODIGO PRINCIPAL
#Entrada
sucio = input("Instruccion? ")

#Limpiar
limpio = sanitize(sucio)

#Salida
print("SUCIO:",sucio)
print("LIMPIO:",limpio)
```



```
def sanitize(s):

    #Miro letra a letra
    #Si la letra no es ; la concateno
    r = ""
    for c in s:
        if c != ";" :
            r = r + c
    return r
```

```
#CODIGO PRINCIPAL
#Entrada
sucio = input("Instruccion? ")

#Limpiar
limpio = sanitize(sucio)

#Salida
print("SUCIO:",sucio)
print("LIMPIO:",limpio)
```



SQL INJECTION FOOLS SPEED TRAPS AND CLEARS YOUR RECORD

by: James Hobson

107 Comments



- XKCD Bobby Tables
 - <https://xkcd.com/327/>
 - <https://es.xkcd.com/strips/exploits-de-una-madre/>
- SQL Injection
 - https://en.wikipedia.org/wiki/SQL_injection
 - https://www.w3schools.com/sql/sql_injection.asp
 - <https://realpython.com/prevent-python-sql-injection/>

Ingeniería															Horario	Agregar al horario
NRC	Sigla	Permite Retiro	¿Se dicta en inglés?	Sec.	¿Requiere Aprob. Especial?	Categoría	Nombre	Profesor	Campus	Créd.	Vacantes					
											Total	Disponibles	Reservadas			
10791	IIC2413	NO	NO	1	NO		Bases de Datos	Soto Adrian	San Joaquín	10	132	1		W:5,6 V:5	CLAS A5 AYU B12	
22702	IIC2413	NO	NO	2	NO		Bases de Datos	Reutter Juan	San Joaquín	10	92	2		W:5,6 V:5	CLAS B17 AYU B17	

CAZA (FOTOGRÁFICA) DE OSOS



TEMAS

“tipo prueba”

Pregunta 2



¡Llega la caza (fotográfica) de osos! Implementa la función `contar_osos(x,y,z)` que cuenta la cantidad de veces que aparece la palabra “OSO” en tres strings. La función recibe 3 strings como parámetros (`x,y,z`), y retorna un entero con el número de OSOs encontrados. Los 3 strings son del mismo largo entre ellos, y están compuestos por ‘O’ y ‘S’. La Figura 1 muestra como contar los OSOs. Se consideran los strings `x`, `y`, `z` uno encima del otro como muestra la figura, los OSOs a considerar son solo los que se pueden leer horizontalmente de izquierda a derecha, verticalmente de arriba a abajo, y diagonalmente de izquierda a derecha. En el ejemplo de la Figura 1, con los strings “OOSOS” (`x`), “SSOSO” (`y`) y “OOOSO” (`z`), la función `contar_osos(x,y,z)` retornaría el entero 6. Ojo que tu función debería funcionar para strings de cualquier largo (pero todos del mismo largo).

x : OOSOS
 y : SSOSO
 z : OOOSO

IIC1103 – Introducción a la Programación
2018 - 1

Interrogación 1

x : OOSOS
 y : SSOSO
 z : OOOSO

x : OOSOS
 y : SSOSO
 z : OOOSO

x : OOSOS
 y : SSOSO
 z : OOOSO

x : OOSOS
 y : SSOSO
 z : OOOSO

x : OOSOS
 y : SSOSO
 z : OOOSO

x : OOSOS
 y : SSOSO
 z : OOOSO

Figura 1: Ejemplo con los strings “OOSOS” (`x`), “SSOSO” (`y`) y “OOOSO” (`z`). En este caso, la función `contar_osos(x,y,z)` retornaría el entero 6.



```
def contar_osos(x,y,z):
```

```
#CÓDIGO PRINCIPAL (TEST)
x = "OOSOS"
y = "SSOSO"
z = "OOOSO"

num = contar_osos(x,y,z)
print(num)
```

x: OOSOS	x: OOSOS	x: OOSOS
y: SSOSO	y: SSOSO	y: SSOSO
z: OOOSO	z: OOOOSO	z: OOOOSO

x: OOSOS	x: OOSOS	x: OOSOS
y: SSOSO	y: SSOSO	y: SSOSO
z: OOOSO	z: OOOOSO	z: OOOOSO





I'LL WAIT
FOR YOU HERE



```
def contar_osos(x,y,z):
```

```
#CÓDIGO PRINCIPAL (TEST)
x = "OOSOS"
y = "SSOSO"
z = "OOOSO"

num = contar_osos(x,y,z)
print(num)
```

x: OOSOS	x: OOSOS	x: OOSOS
y: SSOSO	y: SSOSO	y: SSOSO
z: OOOSO	z: OOOOSO	z: OOOOSO

x: OOSOS	x: OOSOS	x: OOSOS
y: SSOSO	y: SSOSO	y: SSOSO
z: OOOSO	z: OOOOSO	z: OOOOSO



```
def contar_osos(x,y,z):  
    n = len(x)  
    num = 0  
  
    #Horizontal
```

```
#Vertical
```

```
#Diagonal
```

```
return num
```

```
#CÓDIGO PRINCIPAL (TEST)  
x = "OOSOS"  
y = "SSOSO"  
z = "OOOSO"  
  
num = contar_osos(x,y,z)  
print(num)
```

x: O O S O S	x: O O S O S	x: O O S O S
y: S S O S O	y: S S O S O	y: S S O S O
z: O O O S O	z: O O O S O	z: O O O S O

x: O O S O S	x: O O S O S	x: O O S O S
y: S S O S O	y: S S O S O	y: S S O S O
z: O O O S O	z: O O O S O	z: O O O S O



```
def contar_osos(x,y,z):
    n = len(x)
    num = 0

    #Horizontal
    for i in range(0,n-2):
        if x[i]+x[i+1]+x[i+2] == "OSO":
            num += 1
        if y[i]+y[i+1]+y[i+2] == "OSO":
            num += 1
        if z[i]+z[i+1]+z[i+2] == "OSO":
            num += 1
```

#Vertical



#Diagonal



```
return num
```

#CÓDIGO PRINCIPAL (TEST)

```
x = "OOSOS"
y = "SSOSO"
z = "OOOSO"
```

```
num = contar_osos(x,y,z)
print(num)
```

x: O O S O S

y: S S O S O

z: O O O S O

x: O O S O S

y: S S O S O

z: O O O S O

x: O O S O S

y: S S O S O

z: O O O S O

x: O O S O S

y: S S O S O

z: O O O S O

x: O O S O S

y: S S O S O

z: O O O S O

x: O O S O S

y: S S O S O

z: O O O S O



```
def contar_osos(x,y,z):
    n = len(x)
    num = 0

    #Horizontal
    for i in range(0,n-2):
        if x[i]+x[i+1]+x[i+2] == "OSO":
            num += 1
        if y[i]+y[i+1]+y[i+2] == "OSO":
            num += 1
        if z[i]+z[i+1]+z[i+2] == "OSO":
            num += 1

    #Vertical
    for i in range(0,n):
        if x[i]+y[i]+z[i] == "OSO":
            num += 1

    #Diagonal
    for i in range(0,n-2):
        if x[i]+y[i+1]+z[i+2] == "OSO":
            num += 1
        if x[i+1]+y[i+2]+z[i] == "OSO":
            num += 1

    return num
```

```
#CÓDIGO PRINCIPAL (TEST)
x = "OOSOS"
y = "SSOSO"
z = "OOOSO"

num = contar_osos(x,y,z)
print(num)
```

x: O O S O S	x: O O S O S	x: O O S O S
y: S S O S O	y: S S O S O	y: S S O S O
z: O O O S O	z: O O O S O	z: O O O S O
x: O O S O S	x: O O S O S	x: O O S O S
y: S S O S O	y: S S O S O	y: S S O S O
z: O O O S O	z: O O O S O	z: O O O S O



```
def contar_osos(x,y,z):
    n = len(x)
    num = 0

    #Horizontal
    for i in range(0,n-2):
        if x[i]+x[i+1]+x[i+2] == "OSO":
            num += 1
        if y[i]+y[i+1]+y[i+2] == "OSO":
            num += 1
        if z[i]+z[i+1]+z[i+2] == "OSO":
            num += 1

    #Vertical
    for i in range(0,n):
        if x[i]+y[i]+z[i] == "OSO":
            num += 1

    #Diagonal
    for i in range(0,n-2):
        if x[i]+y[i+1]+z[i+2] == "OSO":
            num += 1

    return num
```

#CÓDIGO PRINCIPAL (TEST)

```
x = "OOSOS"
y = "SSOSO"
z = "OOOSO"
```

```
num = contar_osos(x,y,z)
print(num)
```

x: O O S O S	x: O O S O S	x: O O S O S
y: S S O S O	y: S S O S O	y: S S O S O
z: O O O S O	z: O O O S O	z: O O O S O
x: O O S O S	x: O O S O S	x: O O S O S
y: S S O S O	y: S S O S O	y: S S O S O
z: O O O S O	z: O O O S O	z: O O O S O

HAPLAB: APNEA

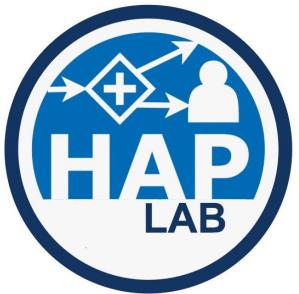


“Me interesan los temas sociales, como por ejemplo, innovaciones que ayuden a las personas que más lo **necesitan.**”

“temas **sociales** (educación, **salud**, etc)”

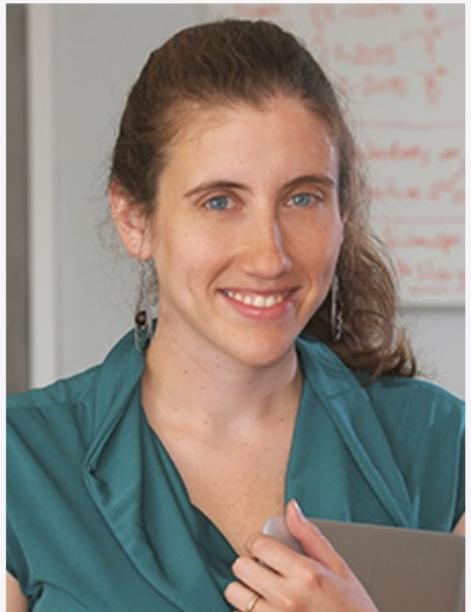
“tipo prueba”

“... su aplicación en **dispositivos**”



Human & Process Research Lab

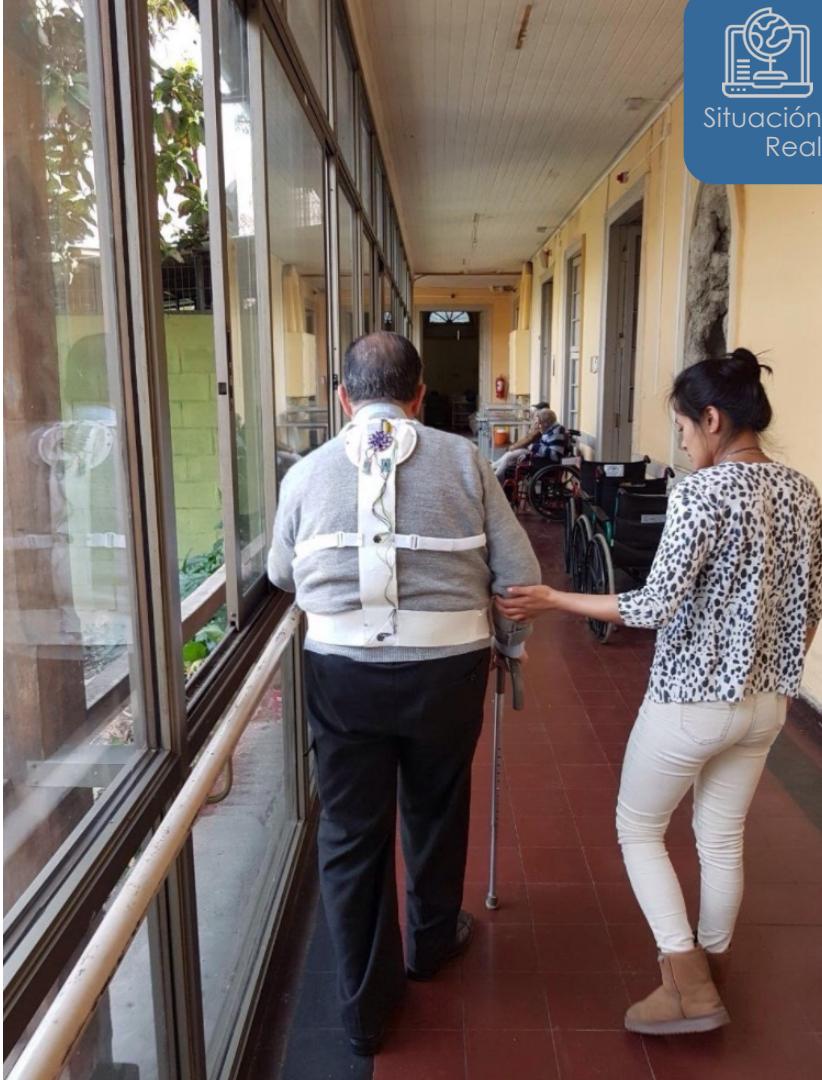
haplab.org



**VALERIA
HERSKOVIC**

Departamento de Ciencia
de la Computación

vhereskov@ing.puc.cl





Estás realizando una investigación de pregrado en HumaLab UC, un grupo que investiga en computación centrada en las personas. El grupo ha desarrollado un dispositivo que monitoriza a los adultos mayores mientras duermen para detectar períodos en donde estos no respiran, llamados **apneas**.

Este dispositivo entrega un carácter por cada segundo que pasa. Si el carácter es una “X”, significa que el adulto mayor no está respirando. Si este es un “.”, significa que éste sí está respirando. Un ejemplo de un periodo de medición es el siguiente:

...XXX...X.X....XXXX.....X.XXX.....XXX....XX..X.

En el **string** anterior, vemos que el adulto estuvo respirando por 3 segundos, luego tuvo una apnea de 3 segundos, luego volvió a respirar por 3 segundos, luego tuvo una apnea de 1 segundo, y así hasta que finaliza el periodo de medición con una respiración de un segundo. En esta secuencia hubo 9 apneas. Como el grupo sabe que estás cursando el curso Introducción a la Programación, te han pedido que programes las siguientes funciones:

Puedes suponer que la secuencia siempre partirá y terminará con el carácter “.”.



Estás realizando una investigación de pregrado en HumaLab UC, un grupo que investiga en computación centrada en las personas. El grupo ha desarrollado un dispositivo que monitoriza a los adultos mayores mientras duermen para detectar períodos en donde estos no respiran, llamados **apneas**.

Este dispositivo entrega un carácter por cada segundo que pasa. Si el carácter es una “X”, significa que el adulto mayor no está respirando. Si este es un “.”, significa que éste sí está respirando. Un ejemplo de un periodo de medición es el siguiente:

...XXX...X.X....XXXX.....X.XXX.....XXX...XX..X.

En el **string** anterior, vemos que el adulto estuvo respirando por 3 segundos, luego tuvo una apnea de 3 segundos, luego volvió a respirar por 3 segundos, luego tuvo una apnea de 1 segundo, y así hasta que finaliza el periodo de medición con una respiración de un segundo. En esta secuencia hubo 9 apneas. Como el grupo sabe que estás cursando el curso Introducción a la Programación, te han pedido que programes las siguientes funciones:

Puedes suponer que la secuencia siempre partirá y terminará con el carácter “.”.

- a) (20 pts) **apneas(secuencia)**: recibe un **string** con una secuencia, y retorna un **int** que representa la cantidad de períodos de apnea que hubo durante el periodo de medición.

Secuencia	apneas
...XXX...X.X....XXXX.....X.XXX.....XXX...XX..X..	9
...XXX....X....XXXX...XXXX...XXX....XX..XXXXXX.	7
.....	0
.XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX.	1
.X.....XX.....XX.....X.....	4





I'LL WAIT
FOR YOU HERE



Estás realizando una investigación de pregrado en HumaLab UC, un grupo que investiga en computación centrada en las personas. El grupo ha desarrollado un dispositivo que monitoriza a los adultos mayores mientras duermen para detectar períodos en donde estos no respiran, llamados **apneas**.

Este dispositivo entrega un carácter por cada segundo que pasa. Si el carácter es una “X”, significa que el adulto mayor no está respirando. Si este es un “.”, significa que éste sí está respirando. Un ejemplo de un periodo de medición es el siguiente:

...XXX...X.X....XXXX.....X.XXX.....XXX...XX..X.

En el **string** anterior, vemos que el adulto estuvo respirando por 3 segundos, luego tuvo una apnea de 3 segundos, luego volvió a respirar por 3 segundos, luego tuvo una apnea de 1 segundo, y así hasta que finaliza el periodo de medición con una respiración de un segundo. En esta secuencia hubo 9 apneas. Como el grupo sabe que estás cursando el curso Introducción a la Programación, te han pedido que programes las siguientes funciones:

Puedes suponer que la secuencia siempre partirá y terminará con el carácter “.”.

- a) (20 pts) **apneas(secuencia)**: recibe un **string** con una secuencia, y retorna un **int** que representa la cantidad de períodos de apnea que hubo durante el periodo de medición.

Secuencia	apneas
...XXX...X.X....XXXX.....X.XXX.....XXX...XX..X..	9
...XXX....X....XXXX...XXXX...XXX....XX..XXXXXX.	7
.....	0
.XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX.	1
.X.....XX.....XX.....X.....	4



```
def apneas(s):
    num = 0
    for i in range(0, len(s)-1):
        if s[i:i+2] == ".X":
            num += 1
    return num
```

Secuencia	apneas
...XXX...X.X....XXXX.....X.XXX.....XXX....XX..X.	9
...XXX....X....XXXX...XXXX....XXX....XX..XXXXX.	7
.....	0
.XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX.	1
.X.....XX.....XX.....X.....	4



Estás realizando una investigación de pregrado en HumaLab UC, un grupo que investiga en computación centrada en las personas. El grupo ha desarrollado un dispositivo que monitoriza a los adultos mayores mientras duermen para detectar períodos en donde estos no respiran, llamados **apneas**.

Este dispositivo entrega un carácter por cada segundo que pasa. Si el carácter es una “X”, significa que el adulto mayor no está respirando. Si este es un “.”, significa que éste sí está respirando. Un ejemplo de un periodo de medición es el siguiente:

...XXX...X.X....XXXX.....X.XXX.....XXX...XX..X.

En el **string** anterior, vemos que el adulto estuvo respirando por 3 segundos, luego tuvo una apnea de 3 segundos, luego volvió a respirar por 3 segundos, luego tuvo una apnea de 1 segundo, y así hasta que finaliza el periodo de medición con una respiración de un segundo. En esta secuencia hubo 9 apneas. Como el grupo sabe que estás cursando el curso Introducción a la Programación, te han pedido que programes las siguientes funciones:

Puedes suponer que la secuencia siempre partirá y terminará con el carácter “.”.

- b) (20 pts) **max_apnea(secuencia)**: recibe un **string** con una secuencia, y retorna un **int** que representa la duración de la apnea más larga que hubo durante el periodo de medición.

Secuencia	apneas	max_apnea
...XXX...X.X....XXXX.....X.XXX.....XXX...XX..X.	9	4
...XXX....X....XXXX...XXXX....XXX....XX..XXXXX.	7	5
.....	0	0
.XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX.	1	35
.X.....XX.....XX.....X.....	4	2





I'LL WAIT
FOR YOU HERE



Estás realizando una investigación de pregrado en HumaLab UC, un grupo que investiga en computación centrada en las personas. El grupo ha desarrollado un dispositivo que monitoriza a los adultos mayores mientras duermen para detectar períodos en donde estos no respiran, llamados **apneas**.

Este dispositivo entrega un carácter por cada segundo que pasa. Si el carácter es una “X”, significa que el adulto mayor no está respirando. Si este es un “.”, significa que éste sí está respirando. Un ejemplo de un periodo de medición es el siguiente:

...XXX...X.X....XXXX.....X.XXX.....XXX...XX..X.

En el **string** anterior, vemos que el adulto estuvo respirando por 3 segundos, luego tuvo una apnea de 3 segundos, luego volvió a respirar por 3 segundos, luego tuvo una apnea de 1 segundo, y así hasta que finaliza el periodo de medición con una respiración de un segundo. En esta secuencia hubo 9 apneas. Como el grupo sabe que estás cursando el curso Introducción a la Programación, te han pedido que programes las siguientes funciones:

Puedes suponer que la secuencia siempre partirá y terminará con el carácter “.”.

- b) (20 pts) **max_apnea(secuencia)**: recibe un **string** con una secuencia, y retorna un **int** que representa la duración de la apnea más larga que hubo durante el periodo de medición.

Secuencia	apneas	max_apnea
...XXX...X.X....XXXX.....X.XXX.....XXX...XX..X.	9	4
...XXX....X....XXXX...XXXX....XXX....XX..XXXXX.	7	5
.....	0	0
.XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX.	1	35
.X.....XX.....XX.....X.....	4	2



```
def max_apnea(s):
    maxi = 0
    l = 0
    for i in range(0,len(s)):
        if s[i] == "X":
            l += 1
        elif s[i] == ".":
            if l > maxi:
                maxi = l
            l = 0
    return maxi
```

Secuencia	apneas	max.apnea
...XXX...X.X....XXXX.....X.XXX.....XXX....XX..X.	9	4
...XXX....X....XXXX...XXXX...XXX....XX...XXXXXX.	7	5
.....	0	0
.XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX.	1	35
.X.....XX.....XX.....X.....X.....	4	2



Estás realizando una investigación de pregrado en HumaLab UC, un grupo que investiga en computación centrada en las personas. El grupo ha desarrollado un dispositivo que monitoriza a los adultos mayores mientras duermen para detectar períodos en donde estos no respiran, llamados **apneas**.

Este dispositivo entrega un carácter por cada segundo que pasa. Si el carácter es una “X”, significa que el adulto mayor no está respirando. Si este es un “.”, significa que éste sí está respirando. Un ejemplo de un periodo de medición es el siguiente:

...XXX...X.X....XXXX.....X.XXX.....XXX....XX..X.

En el **string** anterior, vemos que el adulto estuvo respirando por 3 segundos, luego tuvo una apnea de 3 segundos, luego volvió a respirar por 3 segundos, luego tuvo una apnea de 1 segundo, y así hasta que finaliza el periodo de medición con una respiración de un segundo. En esta secuencia hubo 9 apneas. Como el grupo sabe que estás cursando el curso Introducción a la Programación, te han pedido que programes las siguientes funciones:

Puedes suponer que la secuencia siempre partirá y terminará con el carácter “.”.

- c) (20 pts) **kventana(secuencia, k)**: recibe un **string** con la secuencia y un **int k**. Retorna **True** si la **secuencia** tiene algún período de **k** segundos consecutivos durante el cual el adulto mayor estuvo más tiempo sin respirar que respirando, o **False** en caso contrario.

Secuencia	apneas	max.apnea	kventana (k=5)	kventana (k=8)
...XXX...X.X....XXXX.....X.XXX.....XXX....XX..X.	9	4	True	False
...XXX....X....XXXX...XXXX....XXX....XX..XXXXX.	7	5	True	True
.....	0	0	False	False
.XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX.	1	35	True	True
.X.....XX.....XX.....X.....	4	2	False	False





I'LL WAIT
FOR YOU HERE



Estás realizando una investigación de pregrado en HumaLab UC, un grupo que investiga en computación centrada en las personas. El grupo ha desarrollado un dispositivo que monitoriza a los adultos mayores mientras duermen para detectar períodos en donde estos no respiran, llamados **apneas**.

Este dispositivo entrega un carácter por cada segundo que pasa. Si el carácter es una “X”, significa que el adulto mayor no está respirando. Si este es un “.”, significa que éste sí está respirando. Un ejemplo de un periodo de medición es el siguiente:

...XXX...X.X....XXXX.....X.XXX.....XXX....XX..X.

En el **string** anterior, vemos que el adulto estuvo respirando por 3 segundos, luego tuvo una apnea de 3 segundos, luego volvió a respirar por 3 segundos, luego tuvo una apnea de 1 segundo, y así hasta que finaliza el periodo de medición con una respiración de un segundo. En esta secuencia hubo 9 apneas. Como el grupo sabe que estás cursando el curso Introducción a la Programación, te han pedido que programes las siguientes funciones:

Puedes suponer que la secuencia siempre partirá y terminará con el carácter “.”.

- c) (20 pts) **kventana(secuencia, k)**: recibe un **string** con la secuencia y un **int k**. Retorna **True** si la **secuencia** tiene algún período de **k** segundos consecutivos durante el cual el adulto mayor estuvo más tiempo sin respirar que respirando, o **False** en caso contrario.

Secuencia	apneas	max.apnea	kventana (k=5)	kventana (k=8)
...XXX...X.X....XXXX.....X.XXX.....XXX....XX..X.	9	4	True	False
...XXX....X....XXXX...XXXX....XXX....XX..XXXXX.	7	5	True	True
.....	0	0	False	False
.XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX.	1	35	True	True
.X.....XX.....XX.....X.....	4	2	False	False



```
def kventana(s,k):
    res = False
    for i in range(0,len(s)-k+1):
        v = s[i:i+k]
        num = 0
        for c in v:
            if c == "X":
                num += 1
        if num > len(v) / 2:
            res = True
    return res
```

Secuencia	apneas	max.apnea	kventana (k=5)	kventana (k=8)
...XXX...X.X....XXXX.....X.XXX.....XXX....XX..X.	9	4	True	False
...XXX....X....XXXX...XXXX....XXX....XX..XXXXXX.	7	5	True	True
.....	0	0	False	False
.XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX.	1	35	True	True
.X.....XX.....XX.....X.....	4	2	False	False

NÚMERO O LETRA



Texto: 1 mas 1 son 7
MIN 6 MAY 0 NUM 3 OTROS 4

Texto: todosinEspacios
MIN 15 MAY 0 NUM 0 OTROS 0

Texto: Sant Jordi
MIN 7 MAY 2 NUM 0 OTROS 1

Texto: JARQUE-21
MIN 0 MAY 6 NUM 2 OTROS 1

Se puede hacer simplemente con lo que hemos visto





I'LL WAIT
FOR YOU HERE



Texto: 1 mas 1 son 7
MIN 6 MAY 0 NUM 3 OTROS 4

Texto: todosinEspacios
MIN 15 MAY 0 NUM 0 OTROS 0

Texto: Sant Jordi
MIN 7 MAY 2 NUM 0 OTROS 1

Texto: JARQUE-21
MIN 0 MAY 6 NUM 2 OTROS 1

Se puede hacer simplemente con lo que hemos visto



```
mi = "abcdefghijklmnopqrstuvwxyz"  
ma = "ABCDEFGHIJKLMNOPQRSTUVWXYZ"  
num = "0123456789"
```



```
mi = "abcdefghijklmnopqrstuvwxyz"
ma = "ABCDEFGHIJKLMNPQRSTUVWXYZ"
num = "0123456789"

t = input("Texto: ")

nmi = 0
nma = 0
nnum = 0
notr = 0

for c in t:
    if c in mi:
        nmi += 1
    elif c in ma:
        nma += 1
    elif c in num:
        nnum += 1
    else:
        notr += 1

print("MIN",nmi,"MAY",nma,"NUM",nnum,"OTROS",notr)
```

LIMPIAR RUTS



INTERESES

“cómo organizar un **evento**”

“Vida **universitaria**”

“me gustaría verlo aplicado en problemas
cotidianos para poder así facilitar la forma de
hacer las cosas.”



Mi voz es mi herramienta de trabajo

¿Cómo puedo cuidarla?

Fecha: Viernes 01 de junio de 2018

Hora: 10:00 hrs a 13:00 hrs.

Sala: 401, Edificio CDDoc, Cuarto piso, Campus San Joaquín

* Required

Nombres *

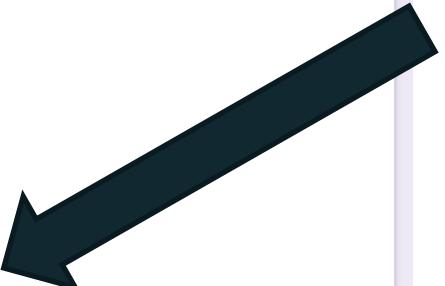
Your answer

Apellidos *

Your answer

RUT o DNI (Sin puntos, con guión) *

Your answer





19638936-9

198942057

19.891.138-0

19.639.729-9

198375950

196385533

14.707.801-3

196239472

19813190-3

19246318-1

198797693

20083177-2





```
import rut
rs = input("RUT? ")
rl = rut.limpiar(rs)
print("RUT LIMPIO:", rl)
```

```
# Nombre 'limpiar'
# Recibe un str con el rut sucio
# (con/sin puntos, guion, espacios, k/K, ...)
# Retorna un str con el rut limpio
# (xx.xxx.xxx-y)
```

RUT? 12345678k
RUT LIMPIO: 12.345.678-K

RUT? 12.345.678.K
RUT LIMPIO: 12.345.678-K

RUT? 12 345 678 K
RUT LIMPIO: 12.345.678-K

RUT? 12345678-k
RUT LIMPIO: 12.345.678-K

RUT? 12-345-678-k
RUT LIMPIO: 12.345.678-K



```
# Nombre 'limpiar'  
  
# Recibe un str con el rut (de 8 digitos) sucio  
# (con/sin puntos, guion, espacios, k/K, ....)  
  
# Retorna un str con el rut limpio  
# (xx.xxx.xxx-y)
```





I'LL WAIT
FOR YOU HERE



```
# Nombre 'limpiar'  
  
# Recibe un str con el rut (de 8 digitos) sucio  
# (con/sin puntos, guion, espacios, k/K, ....)  
  
# Retorna un str con el rut limpio  
# (xx.xxx.xxx-y)
```



```
# Nombre 'limpiar'

# Recibe un str con el rut (de 8 digitos) sucio
# (con/sin puntos, guion, espacios, k/K, ...)

# Retorna un str con el rut limpio
# (xx.xxx.xxx-y)

def limpiar(sucio):
    #Nuevo string (oro) agregando solo los numeros o k o K
    oro = ""
    for c in sucio:
        if c in "0123456789":
            oro += c
        elif c in "kK":
            oro += "K"
    #Poner el oro en el formato correcto
    s = oro[:2]+". "+oro[2:5]+". "+oro[5:8]+"-"+oro[8:]
    return s
```

MÉTODOS

```
t = "Rubi"  
a = t.islower()  
print(a)  
  
p = "terrassa"  
b = p.islower()  
print(b)  
  
y = "Castellar"  
z = y.lower()  
print(z)  
  
m = "SABADELL"  
n = m.isupper()  
print(n)  
  
f = "Sant Llorenç Savall"  
g = f.upper()  
print(g)
```

False
True
castellar
True
SANT LLORENÇ SAVALL

- Métodos

- Funciones especiales que se llaman '.metodo(...)' después de la variable
- Se explicarán en detalle en el futuro (OO)
- Strings tiene MUCHOS métodos
 - TEMARIO SOLO ESTOS 4

GMAIL (+ y .)



INTERESES

“¡Algo para que la XXXX deje de mandarme correos!”

“Vida **universitaria**”

“me gustaría verlo aplicado en problemas **cotidianos** para poder así facilitar la forma de hacer las cosas.”



Email: example@gmail.com
example@gmail.com

Email: ExAmPle@gmail.com
example@gmail.com

Email: mi.nombre@gmail.com
minombre@gmail.com

Email: Nombre.Apellido1.Apellido2@gmail.com
nombreapellido1apellido2@gmail.com

Email: example+iic1103@gmail.com
example@gmail.com





I'LL WAIT
FOR YOU HERE



Email: example@gmail.com
example@gmail.com

Email: ExAmPle@gmail.com
example@gmail.com

Email: mi.nombre@gmail.com
minombre@gmail.com

Email: Nombre.Apellido1.Apellido2@gmail.com
nombreapellido1apellido2@gmail.com

Email: example+iic1103@gmail.com
example@gmail.com



```
largo = input("Email: ")  
  
#Buscar la arroba (siempre hay)  
[REDACTED]  
  
#Quedarse con el user  
[REDACTED]  
  
#Quitar puntos  
[REDACTED]  
  
#Mirar si hay +  
[REDACTED]  
  
#Quitar despues de +  
[REDACTED]  
  
#Minuscula  
[REDACTED]  
  
#Agregar el gmail
```



```
largo = input("Email: ")  
  
#Buscar la arroba (siempre hay)  
for i in range(0,len(largo)):  
    if largo[i] == "@":  
        a = i
```

```
#Quedarse con el user
```

```
#Quitar puntos
```

```
#Mirar si hay +
```

```
#Quitar despues de +
```

```
#Minuscula
```

```
#Agregar el gmail
```



```
largo = input("Email: ")  
  
#Buscar la arroba (siempre hay)  
for i in range(0,len(largo)):  
    if largo[i] == "@":  
        a = i  
  
#Quedarse con el user  
user = largo[:a]
```

#Quitar puntos

#Mirar si hay +

#Quitar despues de +

#Minuscula

#Agregar el gmail



```
largo = input("Email: ")

#Buscar la arroba (siempre hay)
for i in range(0,len(largo)):
    if largo[i] == "@":
        a = i

#Quedarse con el user
user = largo[:a]

#Quitar puntos
sp = ""
for c in user:
    if c != ".":
        sp = sp + c

#Mirar si hay +
[REDACTED]
```

```
#Quitar despues de +
[REDACTED]
```

```
#Minuscula
[REDACTED]
```

```
#Agregar el gmail
[REDACTED]
```



```
largo = input("Email: ")

#Buscar la arroba (siempre hay)
for i in range(0,len(largo)):
    if largo[i] == "@":
        a = i

#Quedarse con el user
user = largo[:a]

#Quitar puntos
sp = ""
for c in user:
    if c != ".":
        sp = sp + c

#Mirar si hay +
#(pos si hay, o -1 si no hay)
m = -1
for i in range(0,len(sp)):
    if sp[i] == "+":
        m = i

#Quitar despues de +
#Minuscula
#Agregar el gmail
```



```
largo = input("Email: ")

#Buscar la arroba (siempre hay)
for i in range(0,len(largo)):
    if largo[i] == "@":
        a = i

#Quedarse con el user
user = largo[:a]

#Quitar puntos
sp = ""
for c in user:
    if c != ".":
        sp = sp + c

#Mirar si hay +
#(pos si hay, o -1 si no hay)
m = -1
for i in range(0,len(sp)):
    if sp[i] == "+":
        m = i

#Quitar despues de +
if m != -1:
    limpio = sp[:m]
else:
    limpio = sp

#Minuscula
#Aregar el gmail
```



```
largo = input("Email: ")

#Buscar la arroba (siempre hay)
for i in range(0,len(largo)):
    if largo[i] == "@":
        a = i

#Quedarse con el user
user = largo[:a]

#Quitar puntos
sp = ""
for c in user:
    if c != ".":
        sp = sp + c

#Mirar si hay +
#(pos si hay, o -1 si no hay)
m = -1
for i in range(0,len(sp)):
    if sp[i] == "+":
        m = i

#Quitar despues de +
if m != -1:
    limpio = sp[:m]
else:
    limpio = sp

#Minuscula
limpiomin = limpio.lower()

#Aregar el gmail
```



```
largo = input("Email: ")

#Buscar la arroba (siempre hay)
for i in range(0,len(largo)):
    if largo[i] == "@":
        a = i

#Quedarse con el user
user = largo[:a]

#Quitar puntos
sp = ""
for c in user:
    if c != ".":
        sp = sp + c

#Mirar si hay +
#(pos si hay, o -1 si no hay)
m = -1
for i in range(0,len(sp)):
    if sp[i] == "+":
        m = i

#Quitar despues de +
if m != -1:
    limpio = sp[:m]
else:
    limpio = sp

#Minuscula
limpiomin = limpio.lower()

#Agregar el gmail
final = limpiomin+"@gmail.com"
print(final)
```



Situación
Real



memes.con.cupos.c • Following ...



memes.con.cupos.college Se
puede mandar a spam?

6w



Liked by ingucdankmemes and
702 others

MARCH 19



Add a comment...

Post

INVERTIR Y PALINDROMO (RECURSIVO)

Jorge, ... perdona que *interrumpa* ... pero es que lo de hacer **funciones recursivas** no me quedó 100% claro. ¿Crees que podrías hacer un ejemplo de función recursiva con strings?

SÍ CLARO



```
x = inverso_rec("aeiou")
print(x)

y = inverso_rec("animal")
print(y)

z = inverso_rec("radar")
print(z)
```

uoiea
lamina
radar

1. Entiendo que tarea tengo que hacer **yo**
2. Pienso en el caso más pequeño, más trivial, más cercano al final => **Caso Base**
3. Me creo que conozco a alguien que hace exactamente **la misma tarea** que yo, pero **más simple**
4. Llama a esa persona para que haga esa tarea pero un poco más simple. **NO te preocupes de cómo lo va a hacer.** No es tu problema
5. Usa el resultado que te ha devuelto para hacer la tarea que te han pedido





I'LL WAIT
FOR YOU HERE

1. Entiendo que tarea tengo que hacer **yo**
2. Pienso en el caso más pequeño, más trivial, más cercano al final => **Caso Base**
3. Me creo que conozco a alguien que hace exactamente **la misma tarea** que yo, pero **más simple**
4. Llama a esa persona para que haga esa tarea pero un poco más simple. **NO te preocupes de cómo lo va a hacer.** No es tu problema
5. Usa el resultado que te ha devuelto para hacer la tarea que te han pedido

1. Entiendo que tarea tengo que hacer **yo**
Retornar el string invertido (al revés vamos). E.g., si me dan “aeiou” retornar “uoiea”
2. Pienso en el caso más pequeño, más trivial, más cercano al final =>
Caso Base
String de una letra => el invertido es lo mismo ☺
3. Me creo que conozco a alguien que hace exactamente **la misma tarea** que yo, pero **más simple**
Alguien que sabe invertido de un string, pero solo con strings más cortos (si el que me han pedido a mi es de 5 letras, el sabe hacer el de 4)
4. Llama a esa persona para que haga esa tarea pero un poco más simple. **NO te preocupes de cómo lo va a hacer.** No es tu problema
Si mi string es “aeiou” (5 letras) ... ¡Eh tu, dame el inverso de “eiou” (4 letras) ! ¡Buscaros la vida!
5. Usa el resultado que te ha devuelto para hacer la tarea que te han pedido
Si tengo el “uoie” (el reverso de “eiou” que le he pedido a mi colega), solo tengo que concatenarle la “a” al final, y ya tengo lo que me han pedido a mi (uoiea)



```
def inverso_rec(p):
```

#CASO BASE

...

#CASO RECURSIVO

...



```
def inverso_rec(p):  
  
    #CASO BASE  
    if len(p) == 1:  
        return p  
    #Una letra  
    #La misma palabra  
  
    #CASO RECURSIVO
```

...



```
def inverso_rec(p):

    #CASO BASE
    if len(p) == 1:          #Una letra
        return p              #La misma palabra

    #CASO RECURSIVO
    else:
        h = p[0]              #La primera letra
        b = p[1:]              #El resto de letras

        x = inverso_rec(b)    #Eh tu, dame el inverso
        r = x + h              #Pongo la letra al final

    return r
```

¡Va Jorge, otro, otro!

SI ME LO PEDIS ASÍ ...



```
x = espal_rec("reconocer")
print(x)
```

True

```
y = espal_rec("animal")
print(y)
```

False

```
z = espal_rec("radar")
print(z)
```

True

True

```
p = espal_rec("rallar")
print(p)
```

1. Entiendo que tarea tengo que hacer **yo**
2. Pienso en el caso más pequeño, más trivial, más cercano al final => **Caso Base**
3. Me creo que conozco a alguien que hace exactamente **la misma tarea** que yo, pero **más simple**
4. Llama a esa persona para que haga esa tarea pero un poco más simple. **NO te preocupes de cómo lo va a hacer.** No es tu problema
5. Usa el resultado que te ha devuelto para hacer la tarea que te han pedido





I'LL WAIT
FOR YOU HERE

1. Entiendo que tarea tengo que hacer **yo**
2. Pienso en el caso más pequeño, más trivial, más cercano al final => **Caso Base**
3. Me creo que conozco a alguien que hace exactamente **la misma tarea** que yo, pero **más simple**
4. Llama a esa persona para que haga esa tarea pero un poco más simple. **NO te preocupes de cómo lo va a hacer.** No es tu problema
5. Usa el resultado que te ha devuelto para hacer la tarea que te han pedido

1. Entiendo que tarea tengo que hacer **yo**

Retornar True si la palabra es palíndroma E.g., True para “radar”

2. Pienso en el caso más pequeño, más trivial, más cercano al final =>
Caso Base

String de una letra => siempre es palíndromo (True) 😊

3. Me creo que conozco a alguien que hace exactamente **la misma tarea** que yo, pero **más simple**

Alguien que sabe decir si un string es palíndromo, pero solo con strings más cortos (si el que me han pedido a mi es de 5 letras, el sabe hacerlo con 4 o 3 o 2 letras)

4. Llama a esa persona para que haga esa tarea pero un poco más simple. **NO te preocupes de cómo lo va a hacer.** No es tu problema

Si mi string es “radar” (5 letras) ... ¡Eh tu, dame el inverso de “ada” (3 letras) ! ¡Buscaros la vida!

5. Usa el resultado que te ha devuelto para hacer la tarea que te han pedido

Si “ada” es palíndromo (es lo que le he preguntado a mi colega), solo tengo que mirar si la primera letra de mi palabra y la ultima son iguales (“r” es igual a “r”). Si son iguales y lo del medio es palíndromo, mi palabra es palíndromo (True). Sino, False



True

radar



True

ada

True

d

False

avion

False

vio

True

i



rallar

'alla

ll

```
ge/publico/codigos/08-Strings/palrec.py", line 11, in espal_rec
    h = p[0]                      #Primera letra
IndexError: string index out of range
```

Uno de los muuuuy pocos casos donde se necesitan
2 Casos Base



1. Entiendo que tarea tengo que hacer yo

Retornar True si la palabra es palíndroma E.g., True para “radar”

2. Pienso en el caso más pequeño, más trivial, más cercano al final =>
Caso Base

Caso Base 1: String de una letra => siempre es palíndromo (True) 😊

Caso Base 2: String de 2 letras => si son iguales True. Sino False

3. Me creo que conozco a alguien que hace exactamente **la misma tarea** que yo, pero **más simple**

Alguien que sabe decir si un string es palíndromo, pero solo con strings más cortos (si el que me han pedido a mi es de 5 letras, el sabe hacerlo con 4 o 3 o 2 letras)

4. Llama a esa persona para que haga esa tarea pero un poco más simple. **NO te preocupes de cómo lo va a hacer.** No es tu problema

Si mi string es “radar” (5 letras) ... ¡Eh tu, dame el inverso de “ada” (3 letras)! ¡Buscaros la vida!

5. Usa el resultado que te ha devuelto para hacer la tarea que te han pedido

Si “ada” es palíndromo (es lo que le he preguntado a mi colega), solo tengo que mirar si la primera letra de mi palabra y la ultima son iguales (“r” es igual a “r”). Si son iguales y lo del medio es palíndromo, mi palabra es palíndromo (True). Sino, False



```
def espal_rec(p):
```

```
#CASO BASE 1
```

...

```
#CASO BASE 2
```

...

```
#CASO RECURSIVO
```

...



```
def espal_rec(p):
    #CASO BASE 1
    if len(p) == 1:
        return True
    #Una letra
    #Una palabra de 1 letra siempre es pal

    #CASO BASE 2
```

...

```
#CASO RECURSIVO
```

...



```
def espal_rec(p):  
  
    #CASO BASE 1  
    if len(p) == 1:          #Una letra  
        return True           #Una palabra de 1 letra siempre es pal  
  
    #CASO BASE 2  
    elif len(p) == 2:         #Dos letras  
        return p[0] == p[1]    #Es pal si las dos letras son iguales  
  
    #CASO RECURSIVO
```

...



```
def espal_rec(p):

#CASO BASE 1
if len(p) == 1:          #Una letra
    return True            #Una palabra de 1 letra siempre es pal

#CASO BASE 2
elif len(p) == 2:         #Dos letras
    return p[0] == p[1]    #Es pal si las dos letras son iguales

#CASO RECURSIVO
else:
    h = p[0]              #Primera letra
    t = p[len(p)-1]        #Ultima letra
    b = p[1:len(p)-1]      #Las letras del medio

    mp = espal_rec(b)     #Eh tu, dime si las letras del medio son pal
    if mp and (h == t):
        res = True
    else:
        res = False

return res
```