100 Ejercicios de programación en Python 3

Problema 1

Escribe un programa que encuentre todos esos números que son divisibles por 7 pero no son múltiplos de 5, entre 2000 y 3200 (ambos incluidos). Los números obtenidos deben imprimirse en una secuencia separada por comas en una sola línea.

Sugerencias:

- Considere usar el método `range(#comienzo, #fin)`

Solución:

```
l = []
for i in range(2000, 3201):
    if (i % 7 ==  0) and (i % 5 != 0):
        l.append(str(i))

print(','.join(l))
```

Problema 2

Escribe un programa que pueda calcular el factorial de un número dado. Los resultados deben imprimirse en una secuencia separada por comas en una sola línea.

Suponga que se proporciona la siguiente entrada al programa: 8

Entonces, la salida debe se: 40320

Solución:

```
def fact(x):
    if x == 0:
        return 1
    return x * fact(x - 1)

x = int(input())
print(f'El factorial de {x} es',fact(x))
```

Problema 3

Con un número entero n dado, escriba un programa para generar un diccionario que contenga (i, i*i) tal que sea un número entero entre 1 y n (ambos incluidos). y luego el programa debería imprimir el diccionario. Suponga que se proporciona la siguiente entrada al programa: 8 La salida debe se: {1: 1, 2: 4, 3: 9, 4: 16, 5: 25, 6: 36, 7: 49, 8: 64}

Sugerencias:

- Considere usar `dict()`

Solución:

```
n = int(input())
d = dict()
for i in range(1, n+1):
    d[i] = i * i

print(d)
```

Problema 4

Escriba un programa que acepte una secuencia de números separados por comas desde la consola y genere una lista y una tupla que contenga cada número.

Suponga que se proporciona la siguiente entrada al programa: 34,67,55,33,12,98

La salida debe se:

['34', '67', '55', '33', '12', '98']

('34', '67', '55', '33', '12', '98')

Sugerencias:

- Se debe suponer que es una entrada por consola.
- El método `tuple()` puede convertir la lista en tupla

```
values = input()
l = values.split(",")
t = tuple(l)
print(l)
print(t)
```

Problema 5

Defina una clase que tenga al menos dos métodos:

- `getString` : para obtener una cadena desde la consola
- `printString` : para imprimir la cadena en mayúsculas.

También incluya una función de prueba simple para probar los métodos de la clase.

Sugerencias:

- Use el metodo `__init__` method para crear algunos parámetros

```
class InputOutString(object):
    def __init__(self):
        self.s = ""

    def getString(self):
        self.s = input()

    def printString(self):
        print(self.s.upper())

strObj = InputOutString()
strObj.getString()
strObj.printString()
```

Problema 6

Escriba un programa que calcule e imprima el valor de acuerdo con la fórmula : $Q = \sqrt{(2 * C * D)/H}$

Los siguientes son los valores fijos de C y H:

- C = 50 y H = 30
- D es la variable cuyos valores deben ingresarse en su programa en una secuencia separada por comas.

Ejemplo

Supongamos que se le da al programa la siguiente secuencia de entrada separada por comas: 100, 150, 180

La salida del programa debe ser: 18, 22, 24

Sugerencias:

- Si la salida recibida está en forma decimal, debe redondearse a su valor más cercano (por ejemplo, si la salida recibida es 26.0, debe imprimirse como 26)

```
import math
c = 50
h = 30
value = []
items = [x for x in input().split(',')]

for d in items:
    value.append(str(int(round(math.sqrt(2* c * float(d)/h)))))

print(','.join(value))
```

Problema 7

Escriba un programa que tome 2 dígitos, X, Y como entrada y genere una matriz bidimensional. El valor del elemento en la i-ésima fila y la j-ésima columna de la matriz debe ser i * j.

Nota: i = 0,1,..., X-1; j = 0,1,...,Y-1.

Ejemplo

Suponga que se proporcionan las siguientes entradas al programa: 3, 5

Entonces, la salida del programa debería ser:

[[0, 0, 0, 0, 0], [0, 1, 2, 3, 4], [0, 2, 4, 6, 8]]

Sugerencias:

- Se debe asumir que es una entrada de consola en forma separada por comas.

```
input_str = input('r,c')
dimensions = [int(x) for x in input_str.split(',')]
rowNum = dimensions[0]
colNum = dimensions[1]
multilist = [[0 for col in range(colNum)] for row in range(rowNum)]

for row in range(rowNum):
    for col in range(colNum):
        multilist[row][col] =  row*col

print(multilist)
```

Problema 8

Escriba un programa que acepte una secuencia de palabras separadas por comas como entrada e imprima las palabras en una secuencia separada por comas después de ordenarlas alfabéticamente.

Suponga que se proporciona la siguiente entrada al programa: sin, hola, bolsa, mundo

La salida del programa debe ser: bolsa, hola, mundo, sin

Solución:

```
elementos = [x for x in input().split(',')]
elementos.sort()
print(','.join(elementos))
```

Problema 9

Escriba un programa que acepte la secuencia de líneas como entrada e imprima las líneas después de poner todos los caracteres de la oración en mayúsculas.

Suponga que se proporciona la siguiente entrada al programa:

```
Hola mundo

La práctica hace la perfección
```

La salida del programa debe ser:

```
HOLA MUNDO

LA PRÁCTICA HACE LA PERFECCIÓN
```

Solución:

```
lineas = []
salir = False
while not salir:
    s = input()
    if s:
        lineas.append(s.upper())
    else:
        salir = True

for sentencia in lineas:
    print(sentencia)
```

Problema 10

Escriba un programa que acepte una secuencia de palabras separadas por espacios en blanco como entrada e imprima las palabras después de eliminar todas las palabras duplicadas y clasificarlas alfanuméricamente.

Suponga que se proporciona la siguiente entrada al programa:

`hola mundo y la práctica hace la perfección y hola mundo de nuevo`

Sugerencias:

- Usamos `set` para eliminar datos duplicados automáticamente y luego usamos `sorted()` para ordenar los datos.

```
s = input()
palabras = [palabra for palabra in s.split(" ")]
print(" ".join(sorted(list(set(palabras)))))
```

Problema 11

Escriba un programa que acepte una secuencia de números binarios de 4 dígitos separados por comas como entrada y luego verifique si son divisibles por 5 o no. Los números divisibles por 5 deben imprimirse en una secuencia separada por comas.

Ejemplo: `0100,0011,1010,1001`

Entonces la salida debería ser: `1010`

Solución:

```
valores = []
elementos = [x for x in input().split(',')]
for p in elementos:
    intp = int(p, 2)
    if not intp  %  5:
        valores.append(p)

print(','.join(valores))
```

Problema 12

Escriba un programa que encuentre todos los números entre 1000 y 3000 (ambos incluidos) de modo que cada dígito del número sea un número par. Los números obtenidos deben imprimirse en una secuencia separada por comas en una sola línea.

Solución:

```
for i in range(1000, 3001):
    s = str(i)
    if (int(s[0]) % 2 ==  0) and (int(s[1]) % 2 ==  0) and (int(s[2]) % 2 ==  0) and (i
nt(s[3]) % 2 ==  0):
        valores.append(s)
print(",".join(valores))
```

Problema 13

Escribe un programa que acepte una oración y calcule el número de letras y dígitos. Suponga que se proporciona la siguiente entrada al programa: ¡Hola Mundo! 123 Entonces, la salida debe ser: LETRAS 10 DIGITOS 3

Solución:

```
s = input('Oración : ')
d = {"DIGITOS":0, "LETRAS":0}
for c in s:
    if c.isdigit():
        d["DIGITOS"]+ = 1
    elif c.isalpha():
        d["LETRAS"]+ = 1
    else:
        pass
print("LETRAS", d["LETRAS"])
print("DIGITOS", d["DIGITOS"])
```

Problema 14

Escribe un programa que acepte una oración y calcule el número de letras mayúsculas y minúsculas.

Suponga que se proporciona la siguiente entrada al programa: ¡Hola Mundo!

Entonces, la salida debería ser:

MAYÚSCULAS 2

MINÚSCULAS 7

Sugerencias:

- En caso de que se suministren datos de entrada, se debe suponer que es una entrada de consola.

```
s = input()
d = {"MAYUSCULAS":0, "MINUSCULAS":0}
for c in s:
    if c.isupper():
        d["MAYUSCULAS"] += 1
    elif c.islower():
        d["MINUSCULAS"] += 1
    else:
        pass
print("MAYÚSCULAS", d["MAYUSCULAS"])
print("MINÚSCULAS", d["MINUSCULAS"])
```

Problema 15

Escriba un programa que calcule el valor de a+aa+aaa+aaaa con un dígito dado como valor de a.

Suponga que se proporciona la siguiente entrada al programa: 9

Entonces, la salida debería ser: 11106

Solución:

```
a = input()
n1 = int( "%s"%a )
n2 = int( "%s%s"%(a,a) )
n3 = int( "%s%s%s"%(a,a,a) )
n4 = int( "%s%s%s%s"%(a,a,a,a) )
print(n1+n2+n3+n4)
```

Problema 16

Defina una lista por comprensión para cuadrar cada número impar en una lista. La lista se introducirá mediante una secuencia de números separados por comas.

Suponga que se proporciona la siguiente entrada al programa: 1,2,3,4,5,6,7,8,9

Entonces, la salida debe ser: 1,3,5,7,9

Solución:

```
valores = input()
numeros = [x for x in valores.split(",") if int(x) % 2 != 0]
print(",".join(numeros))
```

Problema 17

Escriba un programa que calcule el monto neto de una cuenta bancaria basándose en un registro de transacciones de la entrada de la consola.

El formato del registro de transacciones se muestra a continuación:

D 100 .- D significa depósito

R 200 .- R significa retiro

Suponga que se proporciona la siguiente entrada al programa:

D 300

D 300

R 200

D 100

Entonces, la salida debe ser: 500

Solución:

```
saldo_neto = 0
salir = False
while not salir:
    s = input()
    if not s:
        salir = True
    else:
        valores = s.split(" ")
        operacion = valores[0]
        saldo = int(valores[1])
        if operacion ==  "D":
            saldo_neto += saldo
        elif operacion ==  "R":
            saldo_neto -= saldo
        else:
            pass

print('Saldo',saldo_neto)
```

Problema 18

Un sitio web requiere que los usuarios ingresen el nombre de usuario y la contraseña para registrarse. Escriba un programa para verificar la validez de la contraseña ingresada por los usuarios. Los siguientes son los criterios para verificar la contraseña:

- 1 Al menos 1 letra entre [a-z]
- 2 Al menos 1 número entre [0-9]
- 3 Al menos 1 letra entre [A-Z]
- 4 Al menos 1 carácter de [$#@]
- 5 Longitud mínima de la contraseña: 6
- 6 Longitud máxima de la contraseñad: 12

Su programa debe aceptar una secuencia de contraseñas separadas por comas y las verificará de acuerdo con los criterios anteriores. Las contraseñas que coincidan con los criterios deben imprimirse, cada una separada por una coma.

Ejemplo

Si se proporcionan las siguientes contraseñas como entrada al programa:

ABd1234@1,a F1#,2w3E*,2We3345

Entonces, la salida del programa debería ser:

ABd1234@1

Solución:

```
import re
valor = []
elementos = [x for x in input().split(',')]
for p in elementos:
    if len(p)<6 or len(p)>12:
        continue
    else:
        pass
    if not re.search("[a-z]",p):
        continue
    elif not re.search("[0-9]",p):
        continue
    elif not re.search("[A-Z]",p):
        continue
    elif not re.search("[$#@]",p):
        continue
    elif re.search("\s",p):
        continue
    else:
        pass
    valor.append(p)
print(",".join(valor))
```

Problema 19

Debe escribir un programa para ordenar las tuplas (nombre, edad, altura) en orden ascendente donde el nombre es una cadena, la edad y la altura son números.

Los criterios de clasificación son:

1. Ordenar según el nombre;
2. Luego ordene según la edad;
3. Luego ordene por puntaje.

La prioridad es ese nombre > edad > puntuación.

Si se proporcionan las siguientes tuplas como entrada al programa:

Tom,19,80

John,20,90

Jony,17,91

Jony,17,93

Json,21,85

Entonces, la salida del programa debería ser:

[('John', '20', '90'), ('Jony', '17', '91'), ('Jony', '17', '93'), ('Json', '21', '85'), ('Tom', '19', '80')]

Solución:

```
from operator import itemgetter, attrgetter
l = []
salir = False
while not salir:
    s = input()
    if not s:
        salir = True
    else:
        l.append(tuple(s.split(",")))

print(sorted(l, key = itemgetter(0,1,2)))
```

Problema 20

Escriba un programa para eliminar una(s) tupla(s) vacía(s) de una lista de tuplas.

Ejemplo:

Si se proporcionan las siguientes tuplas como entrada al programa : [(), (), ('',), ('a', 'b'), ('a', 'b', 'c'), ('d')]

Entonces, la salida del programa debería ser : [('',), ('a', 'b'), ('a', 'b', 'c'), 'd']

Sugerencias:

- Defina la tupla por comprensión

Solución:

```
li = [(), (), ('',), ('a', 'b'), ('a', 'b', 'c'), ('d')]
li = [t for t in li if t]
print(li)
```

Problema 21

Un robot se mueve en un plano a partir del punto original (0,0). El robot puede moverse hacia ARRIBA, ABAJO, IZQUIERDA y DERECHA con unos pasos determinados. El rastro del movimiento del robot se muestra a continuación:

- ARRIBA 5 , ABAJO 3 , IZQUIERDA 3 , DERECHA 2

Los números después de la dirección son pasos. Escriba un programa para calcular la distancia desde la posición actual después de una secuencia de movimiento y un punto original. Si la distancia es float, imprima el número entero más cercano.

Ejemplo:

Si se proporcionan las siguientes tuplas como entrada al programa:

- ARRIBA 5
- ABAJO 3
- IZQUIERDA 3
- DERECHA 2

Entonces, la salida del programa debería ser: 2

Solución:

```
import math
pos = [0,0]
salir = False
while not salir:
    s = input()
    if not s:
        salir = True
    else:
        movimiento = s.split(" ")
        direccion = movimiento[0]
        pasos = int(movimiento[1])
        if direccion ==  "ARRIBA":
            pos[0] += pasos
        elif direccion ==  "ABAJO":
            pos[0] -= pasos
        elif direccion ==  "IZQUIERDA":
            pos[1] -= pasos
        elif direccion ==  "DERECHA":
            pos[1] += pasos
        else:
            pass

print(int(round(math.sqrt(pos[1]**2 + pos[0]**2))))
```

Problema 22

Escriba un programa para calcular la frecuencia de las palabras de la entrada. La salida debería aparecer después de ordenar la clave alfanuméricamente.

Suponga que se proporciona la siguiente entrada al programa:

¿Es nuevo en Python o elige entre Python 2 y Python 3? Leer Python 2 o Python 3.

Entonces, la salida debería ser:

2:2

3:1

3?:1

Leer:1

Python:5

elige:1

en:1

entre:1

nuevo:1

o:2

y:1

¿Es:1

Solución:

```
freq = {}   # frecuencia de palabras en el texto
linea = "¿Es nuevo en Python o elige entre Python 2 y Python 3? Leer Python 2 o Python
 3"
for palabra in linea.split():
    freq[palabra] = freq.get(palabra,0)+1

palabras = freq.keys()
palabras = sorted(palabras)

for p in palabras:
    print("%s:%d" % (p,freq[p]))
```

Problema 23

Escribe un método que pueda calcular el valor al cuadrado de un número.

Sugerencias:

- Usando el operador `**`

Solución:

```
def cuadrado(num):
    return num ** 2

print(cuadrado(2))
print(cuadrado(3))
```

Problema 24

Python tiene muchas funciones integradas, built-in functions, y si no sabe cómo usarlo, puede leer documentos en línea o encontrar algunos libros. Pero Python tiene una función de documento incorporada para cada función incorporada.

Escriba un programa para imprimir algunos documentos de funciones integradas de Python, como `abs()`, `int()`, `input()`

Y agregue un documento para su propia función

Sugerencias:

- El método de documento integrado es `__doc__`

Solución:

```
print(abs.__doc__)
print(int.__doc__)
print(input.__doc__)

def cuadrado(num):
    '''Devuelve el valor del cuadrado del número de entrada.
       El número de entrada debe ser entero
    '''
    return num**2

print(cuadrado(2))
print(cuadrado.__doc__)
```

Problema 25

Defina una clase que tenga un parámetro de clase y un mismo parámetro de instancia.

Sugerencias:

- Definir un parámetro de instancia, es necesario agregarlo en el método `__init__`
- Puede iniciar un objeto con el parámetro de construcción o establecer el valor más tarde.

Solución:

```
class Persona:
    # Definir el parámetro de clase "nombre"
    nombre = "Persona"

    def __init__(self, nombre = None):
        # self.nombre es el parámetro de instancia
        self.nombre = nombre

ana = Persona("Ana")
print("% s nombre es %s" %(Persona.nombre, jeffrey.ana))

nico = Persona()
nico.nombre = "Nico"
print("%s nombre es %s" %(Persona.nombre, nico.nombre))
```

Problema 26

Defina una función que pueda calcular la suma de dos números.

Sugerencias:

- Definir una función con dos números como argumentos y puede calcular la suma en la función y devolver el valor.

Solución

```
def suma(num1, num2):
    return num1 + num2

print(suma(1,2))
```

Problema 27

Defina una función que pueda convertir un número entero en una cadena e imprimirlo en la consola.

Sugerencias:

- Utilice `str()` para convertir un número en una cadena.

Solución

```
def printValor(n):
    print(str(n))

printValor(3)
```

Problema 28

Defina una función que pueda convertir una cadena en un número entero e imprimirlo en la consola.

Sugerencias:

- Utilice `int()` para convertir una cadena en un número entero

Solución

```
def printValor(n):
    print(int(n))

printValor('3')
```

Problema 29

Defina una función que pueda recibir dos números enteros en forma de cadena y calcular su suma y luego imprimirla en la consola.

Sugerencias:

- Utilice `int()` para convertir una cadena en un número entero.

Solución

```
def printValor(s1,s2):
    print(int(s1) + int(s2))

printValor("3","4")
```

Problema 30

Defina una función que pueda aceptar dos cadenas como entrada y concatenarlas y luego imprimirla en la consola.

Sugerencias:

- Use `+` para concatenar las cadenas

Solución

```
def printValor(s1,s2):
    print(s1 + s2)

printValor("3","4") # 34
```

Problema 31

Defina una función que pueda aceptar dos cadenas como entrada e imprima la cadena con la longitud máxima en la consola. Si dos cadenas tienen la misma longitud, la función debería imprimir todas las cadenas línea por línea.

Sugerencias:

- Use la función `len()` para obtener la longitud de una cadena

Solución

```
def printValor(s1,s2):
    len1 = len(s1)
    len2 = len(s2)
    if len1 > len2:
        print(s1)
    elif len2 > len1:
        print(s2)
    else:
        print(s1)
        print(s2)

printValor("uno","tres")
```

Problema 32

Defina una función que pueda aceptar un número entero como entrada e imprima "Es un número par" si el número es par, de lo contrario imprima "Es un número impar".

Sugerencias:

- Utilice el operador `%` para comprobar si un número es par o impar.

Solución

```
def checkValor(n):
    if n % 2 == 0:
        print("Es un número par")
    else:
        print("Es un número impar")

checkValor(7)
```

Problema 33

Defina una función que pueda imprimir un diccionario donde las claves son números entre 1 `and` 3 (ambos incluidos) `and` los valores son cuadrados de claves.

Sugerencias:

- Utilice el patrón `dict [clave] = valor` para poner la entrada en un diccionario.
- Utilice el operador `**` para obtener la potencia de un número.

Solución

```
def printDict():
    d = dict()
    d[1] = 1
    d[2] = 2**2
    d[3] = 3**2
    print(d)

printDict()
```

Problema 34

Defina una función que pueda imprimir un diccionario donde las claves sean números entre 1 y 20 (ambos incluidos) y los valores sean cuadrados de claves.

Sugerencias:

- Utilice el patrón `dict[clave] = valor` para poner la entrada en un diccionario.
- Utilice el operador `**` para obtener la potencia de un número.
- Utilice `range()` para bucles ..

Solución

```
def printDict():
        d = dict()
        for i in range(1,21):
                d[i] = i**2
        print(d)

printDict()
```

Problema 35

Defina una función que pueda generar un diccionario donde las claves sean números entre 1 y 20 (ambos incluidos) y los valores sean cuadrados de claves. La función debería imprimir solo los valores.

Sugerencias:

- Utilice el patrón `dict[clave] = valor` para poner la entrada en un diccionario.
- Utilice el operador `**` para obtener la potencia de un número.
- Utilice `range()` para bucles.
- Utilice `keys()` para iterar claves en el diccionario. También podemos usar `item()` para obtener pares clave / valor.

Solución

```
def printDict():
        d = dict()
        for i in range(1,21):
                d[i] = i**2
        for (k,v) in d.items():
                print(v)

printDict()
```

Problema 36

Defina una función que pueda generar un diccionario donde las claves sean números entre 1 y 20 (ambos incluidos) y los valores sean cuadrados de claves. La función debería imprimir solo las claves.

Sugerencias:

- Utilice el patrón `dict[clave] = valor` para poner la entrada en un diccionario.
- Utilice el operador `**` para obtener la potencia de un número.
- Utilice `range()` para bucles.
- Utilice `keys()` para iterar claves en el diccionario. También podemos usar `item()` para obtener pares clave / valor.

Solución

```
def printDict():
        d = dict()
        for i in range(1,21):
                d[i] = i**2
        for k in d.keys():
                print(k)

printDict()
```

Problema 37

Defina una función que pueda generar e imprimir una lista donde los valores sean cuadrados de números entre 1 y 20 (ambos incluidos).

Sugerencias:

- Utilice el operador `**` para obtener la potencia de un número.
- Utilice `range()` para bucles.
- Use `list.append()` para agregar valores a una lista.

Solución

```
def printLista():
        li = list()
        for i in range(1,21):
                li.append(i**2)
        print(li)

printLista()
```

Problema 38

Defina una función que pueda generar una lista donde los valores sean cuadrados de números entre 1 y 20 (ambos incluidos). Luego, la función necesita imprimir los primeros 5 elementos de la lista.

Sugerencias:

- Utilice el operador `**` para obtener el poder de un número.
- Utilice `range()` para bucles.
- Utilice `list.append()` para agregar valores a una lista.
- Utilice [n1:n2] para dividir una lista

Solución

```
def printLista():
        li = list()
        for i in range(1,21):
                li.append(i**2)
        print(li[:5])

printLista()
```

Problema 39

Defina una función que pueda generar una lista donde los valores sean cuadrados de números entre 1 y 20 (ambos incluidos). Luego, la función necesita imprimir los últimos 5 elementos de la lista.

Sugerencias:

- Utilice el operador `**` para obtener el poder de un número.
- Utilice `range()` para bucles.
- Utilice `list.append()` para agregar valores a una lista.
- Utilice [n1:n2] para dividir una lista

Solución

```
def printLista():
        li = list()
        for i in range(1,21):
                li.append(i**2)
        print(li[-5:])

printLista()
```

Problema 40

Defina una función que pueda generar una lista donde los valores sean cuadrados de números entre 1 y 20 (ambos incluidos). Luego, la función debe imprimir todos los valores excepto los primeros 5 elementos de la lista.

Sugerencias:

- Utilice el operador `**` para obtener el poder de un número.
- Utilice `range()` para bucles.
- Utilice `list.append()` para agregar valores a una lista.
- Utilice [n1:n2] para dividir una lista

Solución

```
def printLista():
        li = list()
        for i in range(1,21):
                li.append(i**2)
        print li[5:]

printLista()
```

Problema 41

Defina una función que pueda generar e imprimir una tupla donde el valor sea el cuadrado de números entre 1 y 20 (ambos incluidos).

Sugerencias:

- Utilice el operador `**` para obtener el poder de un número.
- Utilice `range()` para bucles.
- Utilice `list.append()` para agregar valores a una lista.
- Utilice `tupla()` para obtener una tupla de una lista.

Solución

```
def printTupla():
        li = list()
        for i in range(1,21):
                li.append(i**2)
        print(tuple(li))

printTupla()
```

Problema 42

Con una tupla dada (1,2,3,4,5,6,7,8,9,10), escriba un programa para imprimir los valores de la primera mitad en una línea y los valores de la última mitad en una línea.

Sugerencias:

- Utilice la notación [n1:n2] para obtener una porción de una tupla.

Solución

```
tp = (1,2,3,4,5,6,7,8,9,10)
tp1 = tp[:5]
tp2 = tp[5:]
print(tp1)
print(tp2)
```

Problema 43

Escriba un programa para generar e imprimir otra tupla cuyos valores sean números pares en la tupla dada (1,2,3,4,5,6,7,8,9,10).

Sugerencias:

- Utilice `for` para iterar la tupla
- Utilice `tupla()` para generar una tupla a partir de una lista.

Solución

```
tp = (1,2,3,4,5,6,7,8,9,10)
li = list()
for i in tp:
    if tp[i] % 2 ==  0:
        li.append(tp[i])

tp2 = tuple(li)
print(tp2)
```

Problema 44

Escriba un programa que acepte una cadena como entrada para imprimir "Sí" si la cadena es "sí" o "SI" o "Sí", de lo contrario, imprima "No".

Sugerencias:

- Utilice la declaración `if` para decidir la estructura condicional.

Solución

```
s = input()
if s == "si" or s == "SI" or s == "Si":
    print "Si"
else:
    print "No"
```

Problema 45

Escriba un programa que pueda filtrar números pares en una lista usando la función de filtro.

La lista es : [1,2,3,4,5,6,7,8,9,10]

Sugerencias:

- Utilice `filter()` para filtrar algunos elementos en una lista.
- Utilice `lambda` para definir funciones anónimas.

Solución

```
li = [1,2,3,4,5,6,7,8,9,10]
numeros_pares = filter(lambda x: x % 2 == 0, li)
print(numeros_pares)
```

Problema 46

Escriba un programa que pueda mapear, map(), para hacer una lista cuyos elementos sean cuadrados de elementos en [1,2,3,4,5,6,7,8,9,10]

Sugerencias

- Utilice `map()` para generar una lista.
- Utilice `lambda` para definir funciones anónimas.

Solución

```
li = [1,2,3,4,5,6,7,8,9,10]
numeros_al_cuadrado = map(lambda x: x**2, li)
print(numeros_al_cuadrado)
```

Problema 47

Escriba un programa que pueda mapear y filtrar para hacer una lista cuyos elementos sean cuadrados de número par en [1,2,3,4,5,6,7,8,9,10]

Sugerencias

- Utilice `map()` para generar una lista.
- Utilice `filter()` para filtrar elementos de una lista.
- Use `lambda` para definir funciones anónimas.

Solución

```
li = [1,2,3,4,5,6,7,8,9,10]
numeros_pares = map(lambda x: x**2, filter(lambda x: x % 2 == 0, li))
print(numeros_pares)
```

Problema 48

Escribe un programa que pueda filtrar, `filter()` , para hacer una lista cuyos elementos sean números pares entre 1 y 20 (ambos incluidos).

Sugerencias:

- Utilice `filter()` para filtrar elementos de una lista.
- Utilice `lambda` para definir funciones anónimas.

Solución

```
numeros_pares = filter(lambda x: x % 2 ==  0, range(1,21))
print(numeros_pares)
```

Problema 49

Escribe un programa que pueda mapear, `map()` , para hacer una lista cuyos elementos sean cuadrados de números entre 1 y 20 (ambos incluidos).

Sugerencias

- Utilice `map()` para generar una lista.
- Utilice `lambda` para definir funciones anónimas.

Solución

```
numeros_al_cuadrado = map(lambda x: x**2, range(1,21))
print(numeros_al_cuadrado)
```

Problema 50

Defina una clase llamada Americano que tenga un método estático llamado `printNacionalidad()` .

Sugerencias:

- Utilice el decorador `@staticmethod` para definir el método estático de la clase.

Solución

```
class Americano(object):
    @staticmethod
    def printNacionalidad():
        print("Americano")

aAmericano = Americano()
aAmericano.printNacionalidad()
Americano.printNacionalidad()
```

Problema 51

Defina una clase denominada Americano y su subclase NewYorker.

Sugerencias:

- Utilice la clase Subclase (`ParentClass`) para definir una subclase.

Solución:

```
class Americano(object):
    pass

class NewYorker(Americano):
    pass

aAmericano = Americano()
aNewYorker = NewYorker()
print(aAmericano)
print(aNewYorker)
```

Problema 52

Defina una clase llamada Circulo que se pueda construir por un radio. La clase Circulo tiene un método que puede calcular el área.

Solución:

```
class Circulo(object):
    def __init__(self, r):
        self.radio = r

    def area(self):
        return self.radio**2 * 3.14

aCirculo = Circulo(2)
print aCirculo.area()
```

Problema 53

Defina una clase denominada Rectángulo que se pueda construir con una longitud y un ancho. La clase Rectangulo tiene un método que puede calcular el área.

Solución:

```
class Rectangulo(object):
    def __init__(self, l, a):
        self.largo = l
        self.ancho = a

    def area(self):
        return self.largo * self.ancho

aRectangulo = Rectangulo(2,10)
print(aRRectangulo.area())
```

Problema 54

Defina una clase denominada Forma y su subclase Cuadrado. La clase Cuadrado tiene una función init que toma una longitud como argumento. Ambas clases tienen una función de área que puede imprimir el área de la forma donde el área de Forma es 0 por defecto.

Sugerencias:

- Para anular un método en la superclase, podemos definir un método con el mismo nombre en la superclase.

Solución:

```
class Forma(object):
    def __init__(self):
        pass

    def area(self):
        return 0

class Cuadrado(Forma):
    def __init__(self, l):
        Forma.__init__(self)
        self.length = l

    def area(self):
        return self.length * self.length

aCuadrado =  Cuadrado(3)
print(aCuadrado.area())
```

Problema 55

Genere una excepción RuntimeError.

Sugerencias:

- Utilice `raise()` para generar una excepción.

Solución:

```
raise RuntimeError('algo va mal')
```

Problema 56

Escriba una función para calcular 5 / 0 y use try/except para detectar las excepciones.

Sugerencias:

- Utilice try/except para detectar excepciones.

Solución:

```
def provocarError():
    return 5/0

try:
    provocarError()
except ZeroDivisionError:
    print("division by zero!")
except Exception, err:
    print('Caught an exception')
finally:
    print('In finally block for cleanup')
```

Problema 57

Defina una clase de excepción personalizada que tome un mensaje de cadena como atributo.

Sugerencias:

- Para definir una excepción personalizada, necesitamos definir una clase heredada de `Exception` .

Solución:

```
class MiError(Exception):
    """Mi propia clase de excepción
      Atributos:
        msg  -- explicación del error
    """

    def __init__(self, msg):
        self.msg = msg

error = MiError("algo va mal")
```

Problema 58

Suponiendo que tenemos algunas direcciones de correo electrónico en el formato " `nombredeusuario@nombredeempresa.com` ", escriba el programa para imprimir el nombre de usuario de una dirección de correo electrónico determinada. Tanto los nombres de usuario como los nombres de empresas se componen únicamente de letras.

Ejemplo:

Si se proporciona la siguiente dirección de correo electrónico como entrada al programa: juan@google.com

Entonces, la salida del programa debería ser: Juan

Sugerencias:

- Utilice `\w` para hacer coincidir letras

Solución:

```
import re
emailAddress = input()
pat2 = "(\w+)@((\w+\.)+(com))"
r2 = re.match(pat2,emailAddress)
print(r2.group(1))
```

Problema 59

Suponiendo que tenemos algunas direcciones de correo electrónico en el formato " `nombredeusuario@nombredeempresa.com` ", escriba el programa para imprimir el nombre de la empresa de una dirección de correo electrónico determinada. Tanto los nombres de usuario como los nombres de empresas se componen únicamente de letras.

Ejemplo:

Si se proporciona la siguiente dirección de correo electrónico como entrada al programa: juan@google.com

Entonces, la salida del programa debería ser: google

Sugerencias:

- Utilice `\w` para hacer coincidir letras.

Solución:

```
import re
emailAddress = input()
pat2 = "(\w+)@(\w+)\.(com)"
r2 = re.match(pat2,emailAddress)
print(r2.group(2))
```

Problema 60

Escriba un programa que acepte una secuencia de palabras separadas por espacios en blanco como entrada para imprimir las palabras compuestas únicamente por dígitos.

Ejemplo:

Si se proporcionan las siguientes palabras como entrada al programa : 2 gatos y 3 perros.

Entonces, la salida del programa debería ser : ['2', '3']

Sugerencias:

- Use `re.findall()` para encontrar todas las subcadenas usando expresiones regulares(regex).

Solución:

```
import re
s = input()
print(re.findall("\d+",s))
```

Problema 61

Imprime una cadena Unicode "hola mundo".

Sugerencias:

- Utilice el formato `u'strings'` para definir una cadena Unicode.

Solución:

```
unicodeString = u"hola mundo"
print(unicodeString)
```

Problema 62

Escriba un programa para leer una cadena ASCII y convertirla en una cadena Unicode codificada por utf-8.

Sugerencias:

- Utilice la función `encode("utf-8")` para convertir.

Solución:

```
s = input()
u = s.encode("utf-8")
print(u)
```

Problema 63

Escriba un comentario especial para indicar que un archivo de código fuente de Python está en Unicode.

Solución:

```
# -*- coding: utf-8 -*-
#---------------------------------------#
```

Problema 64

Escriba un programa para calcular 1/2 + 2/3 + 3/4 + ... + n / n + 1 con una entrada n dada por consola (n > 0).

Ejemplo:

Si se da la siguiente n como entrada al programa: 5

Entonces, la salida del programa debería ser: 3.5500000000000003

Sugerencias:

- Use `float()` para convertir un número entero en float

Solución:

```
n = int(input())
sum = 0.0
for i in range(1 , n+1):
    sum +=  float( float(i) / (i+1))
print(sum)
```

Problema 65

Escribe un programa para calcular:

- f (n) = f (n-1) + 100 cuando n > 0
- f (0) = 1

con una entrada n dada por la consola (n> 0).

Ejemplo:

Si la entrada al programa es : 5

Entonces, la salida del programa debería ser : 500

Sugerencias:

- Podemos definir la función recursiva.

Solución:

```
def f(n):
    if n == 0:
        return 0
    else:
        return f(n-1) + 100

n = int(input())
print(f(n))
```

Problema 66

La secuencia de Fibonacci se calcula con base en la siguiente fórmula:

- f(n) = 0 si n = 0
- f(n) = 1 si n = 1
- f(n) = f(n-1)+f(n-2) si n>1

Escriba un programa para calcular el valor de f (n) con una entrada n dada por la consola.

Ejemplo:

Si la entrada al programa es : 7

Entonces, la salida del programa debería ser : 13

Sugerencias:

- Podemos definir la función recursiva.

Solución:

```
def f(n):
    if n == 0:
        return 0
    elif n == 1:
        return 1
    else:
        return f(n-1) + f(n-2)

n = int(input())
print(f(n))
```

Problema 67

La secuencia de Fibonacci se calcula con base en la siguiente fórmula:

- f(n) = 0 si n = 0
- f(n) = 1 si n = 1
- f(n) = f(n-1)+f(n-2) si n>

Escriba un programa usando la definición de listas por comprensión para imprimir la secuencia de Fibonacci en forma separada por comas con una entrada n dada por consola.

Ejemplo:

Si la entrada al programa es : 7

Entonces, la salida del programa debería ser: 0,1,1,2,3,5,8,13

Sugerencias:

- Podemos definir la función recursiva.
- Utilice la definición de listas por comprensión para generar una lista a partir de una lista existente.
- Utilice `string.join()` para unirse a una lista de cadenas.

Solución:

```
def fibonacci(n):
    if n == 0:
        return 0
    elif n == 1:
        return 1
    else:
        return f(n-1) + f(n-2)

n = int(input())
valores = [str(fibonacci(x)) for x in range(0, n+1)]
print(",".join(valores))
```

Problema 68

Escriba un programa usando un generador para imprimir los números pares entre 0 y n en forma separada por comas mientras n se ingresa por la consola.

Ejemplo:

Si la entrada al programa es : 10

Entonces, la salida del programa debería ser : 0,2,4,6,8,10

Sugerencias:

- Use `yield` para producir el siguiente valor en el generador.

Solución:

```
def generadorPares(n):
    i = 0
    while i <= n:
        if i % 2 ==  0:
            yield i
        i += 1

n = int(input())
valores = []
for i in generadorPares(n):
    valores.append(str(i))

print(",".join(valores))
```

Problema 69

Escriba un programa usando un generador para imprimir los números que pueden ser divisibles por 5 y 7 entre 0 y n en forma separada por comas.

Ejemplo:

Si la entrada al programa es : 100

Entonces, la salida del programa debería ser : 0,35,70

Sugerencias:

- Use `yield` para producir el siguiente valor en el generador.

Solución:

```
def generador(n):
    for i in range(n+1):
        if i % 5 ==  0 and i % 7 ==  0:
            yield i

n = int(input())
valores = []
for i in generador(n):
    valores.append(str(i))

print(",".join(valores))
```

Problema 70

Escriba declaraciones de aserción para verificar que todos los números de la lista [2, 4, 6, 8] sean pares.

Sugerencias:

- Utilice `assert` para hacer una aserción.

Solución:

```
li = [2,4,6,8]
for i in li:
    assert i % 2 ==  0
```

Problema 71

Escriba un programa que acepte expresiones matemáticas básicas desde la consola e imprima el resultado de la evaluación.

Ejemplo:

Si se proporciona la siguiente cadena como entrada al programa : 35 + 3

Entonces, la salida del programa debería ser : 38

Sugerencias:

- Use `eval()` para evaluar la expresión.

Solución:

```
expression = input()
print(eval(expression))
```

Problema 72

Escriba una función de ordenación, por selección, en orden ascendente (de menor a mayor). La función debe devolver la lista ordenada.

Solución:

```
import math
def ordenacion_selec(li):
    for i in range(len(li)):
        menor = i
        for k in range(i+1, len(li)):
            if li[k] < li[menor]:
                menor = k

        intercambio(li, menor, i)
    return li

def intercambio(A, x, y):
    temp = A[x]
    A[x] = A[y]
    A[y] = temp

li = [17,5,222,9,11,2,7]
print(ordenacion_selec(li))
```

Problema 73

Escriba una función de búsqueda binaria que busque un elemento en una lista ordenada. La función debe devolver el índice del elemento que se buscará en la lista.

Sugerencias:

- Utilice `if / elif` para tratar las condiciones.

Solución:

```
import math
def busq_binaria(li, element):
    bottom = 0
    top = len(li) - 1
    index = -1
    while top >= bottom and index == -1:
        mid = int(math.floor((top + bottom) / 2.0))
        if li[mid] == element :
            index = mid
        elif li[mid] > element :
            top = mid - 1
        else :
            bottom = mid + 1

    return index

li = [2,5,7,9,11,17,222]
print(busq_binaria(li,11))
print(busq_binaria(li,12))
```

Problema 74

Genere un aleatorio float donde el valor esté entre 10 y 100 usando el módulo `random` de Python.

Sugerencias:

- Utilice `random.random()` para generar un float aleatorio entre [0,1].

Solución:

```
import random
print(random.random()*100)
```

Problema 75

Genere un aleatorio float donde el valor esté entre 5 y 95.

Sugerencias:

- Utilice `random.random()` para generar un float aleatorio en [0,1].

Solución:

```
import random
print(random.random() * 100 - 5)
```

Problema 76

Escriba un programa para generar un número par aleatorio entre 0 y 10 inclusive, utilizando el módulo `random` y la definición de listas por comprensión.

Sugerencias:

- Utilice `random.choice()` para un elemento aleatorio de una lista.

Solución:

```
import random
print(random.choice([i for i in range(11) if i % 2 ==  0]))
```

Problema 77

Escriba un programa para generar un número aleatorio, que sea divisible entre 5 y 7, entre 0 y 10 inclusive, utilizando el módulo `random` y la definición de listas por comprensión.

Sugerencias:

- Utilice `random.choice()` para un elemento aleatorio de una lista.

Solución:

```
import random
print(random.choice([i for i in range(201) if i % 5 == 0 and i % 7 == 0]))
```

Problema 78

Escriba un programa para generar una lista con 5 números aleatorios entre 100 y 200 inclusive.

Sugerencias:

- Utilice `random.sample()` para generar una lista de valores aleatorios.

Solución:

```
import random
print(random.sample(range(100), 5))
```

Problema 79

Escriba un programa para generar aleatoriamente una lista con 5 números pares entre 100 y 200 inclusive.

Sugerencias:

- Utilice `random.sample()` para generar una lista de valores aleatorios.

Solución:

```
import random
print(random.sample([i for i in range(100,201) if i % 2 == 0], 5))
```

Problema 80

Escriba un programa para generar aleatoriamente una lista con 5 números, que son divisibles entre 5 y 7, entre 1 y 1000 inclusive.

Sugerencias:

- Utilice `random.sample()` para generar una lista de valores aleatorios.

Solución:

```
import random
print(random.sample([i for i in range(1,1001) if i % 5 ==  0 and i % 7 ==  0], 5))
```

Problema 81

Escriba un programa para imprimir aleatoriamente un número entero entre 7 y 15 inclusive.

Sugerencias:

- Utilice `random.randrange()` a un número entero aleatorio en un rango dado.

Solución:

```
import random
print(random.randrange(7,16))
```

Problema 82

Escriba un programa para comprimir y descomprimir la cadena "hola mundo Hola mundo Hola mundo Hola mundo"

Sugerencias:

- Utilice `zlib.compress()` y `zlib.decompress()` para comprimir y descomprimir una cadena.

Solución:

```
import zlib
s = b'hola mundo Hola mundo Hola mundo Hola mundo'
t = zlib.compress(s)
print(t)
print(zlib.decompress(t))
```

Problema 83

Escriba un programa para imprimir el tiempo de ejecución de "1 + 1" 100 veces.

Sugerencias:

- Utilice la función `timeit()` para medir el tiempo de ejecución.

Solución:

```
from timeit import Timer
t = Timer("for i in range(100): 1+1")
print(t.timeit())
```

Problema 84

Escriba un programa para mezclar e imprimir la lista [3,6,7,8].

Sugerencias:

- Utilice la función `shuffle()` para mezclar la lista.

Solución:

```
from random import shuffle
li = [3,6,7,8]
shuffle(li)
print(li)
```

Problema 85

Escriba un programa que al dar un número entero arbitrario a `random.seed()` , se puede establecer la semilla para generar números aleatorios.

Sugerencias:

- Utilice la función `shuffle()` para mezclar una lista.

Solución:

```
import random
li = list(range(5))
print(li)
x = int(input())
random.seed(x)
random.shuffle(li)
print(li)
```

Problema 86

Escriba un programa para generar todas las oraciones en las que el sujeto esté en ["Yo", "Tu"] y el verbo en ["Juego", "Amo"] y el objeto en ["Hockey", "Football"].

Sugerencias:

- Utilice la notación de `lista[indice]` para obtener un elemento de una lista.

Solución:

```
sujetos = ["Yo", "Tu"]
verbos = ["Juego", "Amo"]
objectos = ["Hockey","Football"]
for i in range(len(sujetos)):
    for j in range(len(verbos)):
        for k in range(len(objectos)):
            sentence = "%s %s %s." %(sujetos[i], verbos[j], objectos[k])
            print(sentence)
```

Problema 87

Escriba un programa para imprimir la lista después de eliminar eliminar números pares en [5,6,77,45,22,12,24].

Sugerencias:

- Utilizando la definición de listas por comprensión para eliminar elementos de una lista..

Solución:

```
li = [5,6,77,45,22,12,24]
li = [x for x in li if x % 2 != 0]
print(li)
```

Problema 88

Utilizando la definición de listas por comprensión, escriba un programa para imprimir la lista después de eliminar los números de eliminación que son divisibles por 5 y 7 en [12,24,35,70,88,120,155].

Sugerencias:

- Utilizando la definición de listas por comprensión para eliminar elementos de una lista.

Solución:

```
li = [12,24,35,70,88,120,155]
li = [x for x in li if x % 5 != 0 and x % 7 != 0]
print(li)
```

Problema 89

Utilizando la definición de listas por comprensión, escriba un programa para imprimir la lista después de eliminar los números 1º, 2º, 4º, 6º en [12,24,35,70,88,120,155].

Sugerencias:

- Utilizando la definición de listas por comprensión para eliminar elementos de una lista.
- Utilice `enumerate()` para obtener (indice, valor).

Solución:

```
li = [12,24,35,70,88,120,155]
li = [x for (i,x) in enumerate(li) if i % 2 != 0]
print(li)
```

Problema 90

Utilizando la definición de listas por comprensión, por favor escriba un programa que genere una matriz 3D de 3 x 5 x 8 cuyos elementos sean 0.

Sugerencias:

- Utilizando la definición de listas por comprensón para construir la matriz.

Solución:

```
array = [[ [0 for col in range(8)] for col in range(5)] for fil in range(3)]
print(array)
```

Problema 91

Utilizando la definición de listas por comprensión, escriba un programa para imprimir la lista después de eliminar los números de la posición 0, 4 y 5 en la lista [12,24,35,70,88,120,155].

Sugerencias:

- Utilizando la definición de listas por comprensión, para eliminar un grupo de elementos de una lista.
- Utilice `enumerate()` para obtener (indice, valor).

Solución:

```
li = [12,24,35,70,88,120,155]
li = [x for (i,x) in enumerate(li) if i not in (0,4,5)]
print(li)
```

Problema 92

Utilizando la definición de listas por comprensión, escriba un programa para imprimir la lista después de eliminar el valor 24 en [12,24,35,24,88,120,155].

Sugerencias:

- Utilice algún método de eliminación de la lista para eliminar un valor.

Solución:

```
li = [12,24,35,24,88,120,155]
li = [x for x in li if x != 24]
print(li)
```

Problema 93

Dadas dos listas [1,3,6,78,35,55] y [12,24,35,24,88,120,155], escriba un programa para hacer una lista cuyos elementos sean la intersección de las listas dadas anteriormente.

Sugerencias:

- Utilice `set()` y `&=` para establecer la operación de intersección.

Solución:

```
set1 = set([1,3,6,78,35,55])
set2 = set([12,24,35,24,88,120,155])
set1 &=  set2
li = list(set1)
print(li)
```

Problema 94

Con una lista dada [12,24,35,24,88,120,155,88,120,155], escriba un programa para imprimir esta lista después de eliminar todos los valores duplicados con el orden original reservado.

Sugerencias:

- Utilice `set()` para almacenar varios valores sin duplicarlos.

Solución:

```
def eliminar_duplicado(li):
    nuevalist = []
    conj = set()
    for elemento in li:
        if elemento not in conj:
            conj.add(elemento)
            nuevalist.append(elemento)

    return nuevalist

li = [12,24,35,24,88,120,155,88,120,155]
print(eliminar_duplicado(li))
```

Problema 95

Defina una clase Persona y sus dos clases secundarias: Hombre y Mujer. Todas las clases tienen un método "getGenero" que puede imprimir "Hombre" para la clase Masculina y "Mujer" para la clase Femenina.

Sugerencias:

- Utilice Subclass (Parentclass) para definir una clase secundaria.

Solución:

```
class Persona(object):
    def getGenero( self ):
        return "Desconocido"

class Hombre( Persona ):
    def getGenero( self ):
        return "Hombre"

class Mujer( Persona ):
    def getGenero( self ):
        return "Mujer"

aHombre = Hombre()
aMujer =  Mujer()
print(aHombre.getGenero())
print(aMujer.getGenero())
```

Problema 96

Escriba un programa que cuente e imprima los números de cada carácter en una cadena introducida por la consola.

Ejemplo:

Si se proporciona la siguiente cadena como entrada al programa : abcdefgabc

Entonces, la salida del programa debería ser:

a, 2

b, 2

c, 2

d, 1

e, 1

f, 1

g, 1

Sugerencias:

- Utilice dict para almacenar pares clave/valor.
- Utilice el método `dict.get()` para buscar una clave con el valor predeterminado.

```
dic = {}
s = input()
for s in s:
    dic[s] = dic.get(s,0) + 1
print('\n'.join([' % s, % s'  %  (k, v) for k, v in dic.items()]))
```

Problema 97

Escriba un programa que acepte una cadena de la consola e imprímalo en orden inverso.

Ejemplo:

Si se proporciona la siguiente cadena como entrada al programa:

levántate para ir a clase

Entonces, la salida del programa debería ser:

esalc a ri arap etatnável

Sugerencias:

- Use list [:: - 1] para iterar una lista en orden inverso.

```
s = input()
s = s[::-1]
print(s)
```

Problema 98

Escriba un programa que acepte una cadena de la consola e imprima los caracteres que tienen índices pares.

Ejemplo:

Si se proporciona la siguiente cadena como entrada al programa : H1o2l3a4 5M6u7n8d9o

Entonces, la salida del programa debería ser : Hola Mundo

Sugerencias:

- Use list [::2] para iterar una lista con salto 2.

```
s = input()
s = s[::2]
print(s)
```

Problema 99

Escriba un programa que imprima todas las permutaciones de [1,2,3]

Sugerencias:

- Utilice `itertools.permutations()` para obtener permutaciones de la lista.

Solución:

```
import itertools
print(list(itertools.permutations([1,2,3])))
```

Problema 100

Escribe un programa para resolver un clásico y antiguo rompecabezas chino : Contamos 35 cabezas y 94 patas entre los pollos y conejos de una granja. ¿Cuántos conejos y cuántos pollos tenemos?

Sugerencias:

- Use `for` para iterar todas las posibles soluciones.

Solución:

```
def resolver(numcabezas,numpatas):
    ns = '¡No hay soluciones!'
    for i in range(numcabezas + 1):
        j = numcabezas - i
        if 2*i + 4*j == numpatas:
            return i, j
    return ns, ns

cabezas = 35
patas = 94
soluciones = resolver(cabezas,patas)
print(soluciones)
```