

APELLIDO:	CALIFICACIÓN:
NOMBRE:	
DNI (registrado en SIU Guaraní):	
E-MAIL:	DOCENTE (nombre y apellido):
TEL:	
AULA:	

Duración del examen: 1:20h. Completar con **letra clara, mayúscula e imprenta**. El examen consta de 11 preguntas de opción múltiple. Cada pregunta tiene una y sólo una respuesta correcta.

Las respuestas deben completarse con una X en la siguiente matriz:

Opción	EJ. 1	EJ. 2	EJ. 3	EJ. 4	EJ. 5	EJ. 6	EJ. 7	EJ. 8	EJ. 9	EJ. 10	EJ. 11
1											
2											
3											
4											

**¡ATENCIÓN!** Las respuestas sólo se considerarán válidas si se encuentran en la matriz. De haber diferencias entre la opción seleccionada en el ejercicio y en la matriz, se considerará como válida la de la matriz.

Ejercicio 0104 - 1 punto			
¿Qué contiene <i>b</i> ? <pre>def organiza(n):     resp=n*3     return abs(resp)  a=[1,-5,10,2,-20,6] b=list(map(organiza,a)) print(b)</pre>			
1.	[6, 20, 2, 10, 5, 1]		1
2.	[1, 5, 10, 2, 20, 6]		2
3.	[18, -60, 6, 30, -15, 3]		3
4.	[3, 15, 30, 6, 60, 18]	X	4

<b>Ejercicio 0204 - 1 punto</b>			
<p><b>¿Cuál versión de la función ingreso() valida correctamente los datos de acceso de usuarios? Deben coincidir usuario y clave</b></p> <pre>def ingreso(usr):     -     -  # users contiene [usuario,clave] users=[['jmiguel','X25fc230'],         ['irojas','Irma1211'],         ['jsal','joacoS01']]  while not ingreso(users):     print('Datos de Acceso Erróneos')</pre>			
1.	<pre>def ingreso(usr):     u=input('Usuario: ').upper()     cl=input('Clave: ').upper()     if u in usr and cl in usr:         return True</pre>		1
2.	<pre>def ingreso(usr):     u=input('Usuario: ')     cl=input('Clave: ')     resp=False     for elem in usr:         if cl==elem[1]:             resp=True     return resp</pre>		2
3.	<pre>def ingreso(usr):     u=input('Usuario: ')     cl=input('Clave: ')     if [u,cl] in usr:         return False     return True</pre>		3
4.	<pre>def ingreso(usr):     u=input('Usuario: ')     cl=input('Clave: ')     return [u,cl] in usr</pre>	X	4

<b>Ejercicio 0304 - 1 punto</b>			
<p><b>¿Cuál versión de la función realInter() valida correctamente el ingreso de un número real en el intervalo [0,1] (números entre 0 y 1 inclusive)?</b>  <b>Nota: Se espera que la función no termine hasta obtener y devolver un valor correcto</b></p> <pre>def realInter(cartel,desde,hasta):     -     -  num=realInter('Ingresá un número real entre 0 y 1 inclusive: ',0,1)</pre>			
1	<pre>def realInter(cartel,desde,hasta):     sigue=True     while sigue:         n=input(cartel)         try:             n=float(n)             if n&gt;hasta or n&lt;desde:                 print('entre',desde,'y',hasta)                 sigue=True         except ValueError:             print('Número real')             sigue=True         sigue=False     return n</pre>		1
2	<pre>def realInter(cartel,desde,hasta):     n=input(cartel)     try:         n=float(n)         if n&gt;hasta:             print('entre',desde,'y',hasta)     except ValueError:         print('Número real')     return n</pre>		2

3	<pre>def realInter(cartel, desde, hasta):     sigue=True     while sigue!=True:         n=input(cartel)         try:             n=float(n)             if hasta&lt;n and n&lt;desde:                 sigue=False             else:                 print('entre', desde, 'y', hasta)         except ValueError:             print('Número real')     return n</pre>		3
4	<pre>def realInter(cartel, desde, hasta):     sigue=True     while sigue==True:         try:             n=float(input(cartel))             if hasta&gt;=n and n&gt;=desde:                 sigue=False             else:                 print('entre', desde, 'y', hasta)         except ValueError:             print('Número real')     return n</pre>	X	4

**Ejercicio 0404 - 1 punto**

¿Cuál versión de la función cuenta() devuelve la cantidad de nombres de longitud no mayor a 5 en el archivo nombres, que tiene 3 líneas con nombres separados por -?

Contenido de nombres.txt:

Andrea-LAURA-MÁXIMO-Ileana  
jorgelina-Silvana-amalia-alberto-andrés  
ALMA-ELENA

cuenta() debe devolver 3

```
def cuenta(arch, largo):
    -
    -

datos=open('nombres.txt', encoding='utf-8')
cantidad=cuenta(datos, 6)
datos.close()
print('Hay', cantidad, 'nombres con largo menor a 5')
```

1	<pre>def cuenta(arch, largo):     cant=0     lineas=arch.readlines()     for lin in lineas:         lin=lin.strip('\n').split('-')         for nom in lin:             if len(nom)&lt;largo:                 cant+=1     return cant</pre>	X	1
2	<pre>def cuenta(arch, largo):     cant=0     linea=arch.readline()     lin=linea.strip('\n').split('-')     for nom in lin:         if len(nom)&gt;5:             cant+=1     return largo</pre>		2
3	<pre>def cuenta(arch, largo):     cant=0     linea=arch.readlines()     i=0     while i&lt;len(linea):         lin=linea[i].split('-')         for nom in lin:             if len(nom)&lt;=largo-1:                 cant+=1     return cant</pre>		3

4	<pre>def cuenta (arch, largo) :     linea=arch.readlines ()     i=0     while i&lt;len(linea):         i+=1         cant=0         for nom in linea[i]:             if len(nom)&lt;=largo-1:                 cant+=1     return cant</pre>	4
---	--	---

**Ejercicio 0504 - 1 punto**

Dado el siguiente DataFrame *recaudacion*:

	cajero	caja	total
0	juan	3	104000.77
1	ana	1	256000.00
2	mario	5	99025.40
3	juan	1	112000.00
4	ana	2	77090.00
5	mario	3	86450.00

Que contiene 6 filas y 3 columnas: nombre del cajero (*cajero*), número de la caja en que cobró (*caja*) y total cobrado en esa ocasión (*total*).

¿Qué instrucción produce la siguiente salida?

```
734566.17
```

1	<code>recaudacion.describe()</code>	1
2	<code>recaudacion.head()</code>	2
3	<code>recaudacion['total'].sum()</code>	X 3
4	<code>recaudacion.groupby(recaudacion['caja'])['total'].sum()</code>	4

**Ejercicio 0604 - 1 punto**

Dado el siguiente programa:

```
def obtieneApe(t):
    return ''.join(t.split()[-1]).lower()

nombres=['ana López','emiliano SAL','LORENA ana báunes',
        'analía Soto','ángelea FALCÓN',
        'Luciana analía perez','Ana María Giménez']

resultado= . . .
for nom in resultado:
    print(nom)
```

Que produce la siguiente salida:

```
lópez
sal
báunes
soto
falcón
pérez
giménez
>>>
```

¿Qué instrucción debería ir en los puntos suspensivos?

**Nota:** El argumento *key* permite pasarle a la función un criterio alternativo de comparación entre los elementos de la estructura. En este caso se comparan las versiones de los nombres en mayúsculas.

1	<code>list(filter(obtieneApe, nombres))</code>		1
2	<code>list(map(obtieneApe, nombres))</code>	X	2
3	<code>list(reversed(nombres))</code>		3
4	<code>list(sorted(nombres, key=obtieneApe))</code>		4

**Ejercicio 0704 - 1 punto**

Dado el siguiente programa:

```
print('Ingresá números naturales, 0 para terminar')
num=1
i=0
j=1
lista=[0,0,0,0,0]
sobran=[]
while num!=0:
    num=int(input(str(j)+' : '))
    if num>0 and num%2==0:
        try:
            lista[i]=num
            i+=1
        except IndexError:
            sobran.append(num)
    j+=1
while 0 in lista:
    lista.remove(0)
```

Y los siguientes ingresos:

```
1: 7
2: -8
3: 55
4: 4
5: 2
6: 0
```

¿Qué contenido tendrán *lista* y *sobran* al finalizar?

1	<code>lista: [7, -8, 55, 4, 2]</code> <code>sobran: [0]</code>		1
2	<code>lista: [4, 2]</code> <code>sobran: []</code>	X	2
3	<code>lista: [4, 2]</code> <code>sobran: [7, -8, 55]</code>		3
4	<code>lista: [4, 2, 0, 0, 0]</code> <code>sobran: [0, 0, 0]</code>		4

**Ejercicio 0804 - 1 punto**



¿Cuál de las siguientes líneas de código NO SE CORRESPONDE con la figura?

1.	ax.set_ylabel('Ventas (\$)')		1
2.	ax.grid(axis = y)	X	2
3.	ax.set_ylim(0, 7)		3
4.	ax.set_title("Gráfico de ventas")		4

**Ejercicio 0904 - 1 punto**

Dado el siguiente dataframe, como ejemplo se muestran las primeras 5 filas.

	local	visitante	goles local	goles visitante	posesion local	posesion visitante
0	QATAR	ECUADOR	0	2	42%	50%
1	ENGLAND	IRAN	6	2	72%	19%
2	SENEGAL	NETHERLANDS	0	2	44%	45%
3	UNITED STATES	WALES	1	1	51%	39%
4	ARGENTINA	SAUDI ARABIA	1	2	64%	24%

Si se quiere obtener el equipo que haya convertido menos de 3 goles con más posesión en un partido.

¿ Qué metodos/funciones de pandas se deberían usar ?

1	value_count, loc		1
2	loc, mean		2
3	sort_values, head	X	3
4	Ninguna de las anteriores		4

**Ejercicio 1004 - 2 puntos**

¿Cómo queda el archivo invitados.txt luego de ejecutar el siguiente programa?

```
def leer(arch):
    amigos=open(arch, 'r+', encoding='utf-8')
    datos=amigos.readlines()
    amigos.close()
    colegio=list(filter(delCole, datos))
    return colegio

def delCole(lin):
    return 'colegio' in lin

def guarda(lista, arch):
    datos=open(arch, 'w', encoding='utf-8')
    for amigo in lista:
        lin=amigo.split('-')
        lin=' '.join(lin[:2])
        datos.write(lin+'\n')
    datos.close()

colegio=leer('amigos.txt')
guarda(colegio, 'invitados.txt')
```

Nota: El contenido de amigos.txt es el siguiente:

laura-álvarez-colegio  
 marina-ibarra-trabajo  
 juan-peláez-trabajo  
 ignacio-sotto-barrio  
 diego-pérez-colegio  
 emanuel-garcía-colegio  
 micaela-duarte-colegio

Por cada línea viene nombre, apellido y grupo de pertenencia un amigo separado por guión

1.	Laura Álvarez		1
2.	marina-ibarra-trabajo juan-peláez-trabajo ignacio-sotto-barrio		2
3.	MARINA IBARRA JUAN PELÁEZ IGNACIO SOTTO		3
4.	laura álvarez diego p�rez emanuel garc�a micaela duarte	X	4

**Ejercicio 1104 - 2 puntos**

Dado el siguiente DataFrame *menu*:

	nombre	principal	calor�as
lunes	tallarines	hidratos	1500
martes	milanesas	prote�nas	1800
mi�rcoles	tarta de choclo	hidratos	1000
jueves	lenguado	prote�nas	950

Que contiene 4 filas (el d a de la semana es el  ndice de la fila) y 3 columnas: nombre del plato (nombre), grupo nutricional principal (principal) y calor as por porci n (calor as).

 Qu  se muestra como resultado al ejecutar las siguientes instrucciones?

```
menu.loc['martes', 'calor as']=800
menu.groupby('principal')['calor as'].min()
```

1	principal hidratos 1000.0 prote�nas 800.0	X	1
2	nombre calor�as Martes milanesas 800.0 Mi�rcoles tarta de choclo 1000.0		2
3	nombre principal calor�as Martes milanesas prote�nas 800		3
4	principal prote�nas 950.0		4