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 0106 - 1 punto

¿Qué contiene *b* ?

```
def organiza(n):
  res=False
  if n<0:
    res=True
  return res

a=[15,-150,150,22,-76]
a.reverse()
b=list(filter(organiza,a))
```

1.	**[-76, -150]**	**X**	**1**
2.	**[-150,-76]**		**2**
3.	**[15,-150,150,22,-76]**		**3**
4.	**[15,150,22]**		**4**

Ejercicio 0206 - 1 punto

¿Cuál versión de la función ingreso() valida correctamente los datos de acceso a una cuenta de mail? Deben coincidir dirección de mail y password

```
def ingreso(usr):

def ingreso(usr):
   -
   -

# users contiene [password,dirección mail]
users=[['X25fc230','info@usina.com.ar'],
       ['Irma1211','iro@hotmail.com'],
       ['joacoS01','ergo@gmail.com']]
while ingreso(users):
  print('Datos de Acceso Erróneos')
```

1.	`def ingreso(usr):`
	`  dccMail=input('Mail: ')`
	`  clave=input('Password: ')`
	`  return usr.count(clave)==0`
			1

2.	`def ingreso(usr):`
	`  dccMail=input('Mail: ')`
	`  clave=input('Password: ')`
	`  return usr.count([clave,dccMail])==0`
		X	**2**

3.	`def ingreso(usr):`
	`  dccMail=input('Mail: ')`
	`  clave=input('Password: ')`
	`  return usr.count([dccMail,clave])!=0`
			3

4.	`def ingreso(usr):`
	`  dccMail=input('Mail: ')`
	`  clave=input('Password: ')`
	`  i=0`
	`  for elem in usr:`
	`  if elem[1]==clave:`
	`  i+=1`
	`  return i==0`
			4

Ejercicio 0306 - 1 punto

¿Cuál versión de la función intercala() devuelve una lista con los datos intercalados de dos listas que recibe, independiente de la longitud de cada una de ellas y el orden de intercalación?
Nota: intercala(a,b) debe devolver [1,10,2,20,3,30,40,50]
intercala(b,a) debe devolver [10,1,20,2,30,3,40,50]

```
def intercala(lis1,lis2):
  -
  -

a=[1,2,3]
b=[10,20,30,40,50]
c=intercala(a,b)
d=intercala(b,a)
```

1.	`def intercala(lis1,lis2):`
	`  lisTot=lis1+lis2`
	`  largo=max(len(lis1),len(lis2))`
	`  j=0`
	`  for i in range(largo):`
	`  try:`
	`  lisTot[j]=lis1[i]`
	`  lisTot[j+1]=lis2[i]`
	`  except IndexError:`
	`  j-=1`
	`  j+=2`
	`  return lisTot`
		X	**1**

2.	`def intercala(lis1,lis2):`
	`  lisTot=lis1+lis2`
	`  largo=max(len(lis1),len(lis2))`
	`  i=0`
	`  while i<largo:`
	`  try:`
	`  lisTot[i]=lis1[i]`
	`  lisTot[i]=lis2[i]`
	`  except IndexError:`
	`  i=i`
	`  i+=1`
	`  return lisTot`
			2

3.	`def intercala(lis1,lis2):`
	`  lisTot=lis1+lis2`
	`  i=0`
	`  while i<len(lisTot):`
	`  try:`
	`  lisTot[i]=lis1[i]`
	`  except IndexError:`
	`  nada=0`
	`  try:`
	`  lisTot[i+1]=lis2[i]`
	`  except IndexError:`
	`  nada=0`
	`  return lisTot`
			3

4.	`def intercala(lis1,lis2):`
	`  lisTot=lis1+lis2`
	`  largo=max(len(lis1),len(lis2))`
	`  for i in range(largo):`
	`  try:`
	`  lisTot[i]=lis1[i]`
	`  lisTot[i+1]=lis2[i]`
	`  except IndexError:`
	`  nada=0`
	`  return lisTot`
			4

Ejercicio 0406 - 1 punto

¿Cuál versión de la función guarda() genera un archivo personal.txt que contendrá únicamente los datos que le pasa el programa? El archivo debe quedar con el siguiente formato y contenido:
1,Paz A.
2,Alza I.
3,Ortiz S.

```
def guarda(dicci,arch):
  -
  -

nombres={'ana':'paz','inés':'alza','sergio':'ortiz'}
guarda(nombres,'personal.txt')
```

1.

```
def guarda(dicci,arch):
  datos=open(arch,'r',encoding='utf-8')
  lineas=['']*len(dicci)
  i=0
  for nom in dicci:
    lin=str(i)+','+dicci[nom]+' '
    lin=nom.upper()+'.'+'\n'+lin
    lineas[i]=lin
    i+=1
  datos.writelines(lineas)
  datos.close()
```

1

2.

```
def guarda(dicci,arch):
  datos=open(arch,'w',encoding='utf-8')
  lineas=['']*len(dicci)
  for nom in dicci:
    i=1
    lin=str(i)+','+dicci[nom][0].upper()+'.,\n'
    lin=nom.upper()+'.'+'\n'
    lineas[i]=lin
    i+=1
  datos.writelines(lineas)
  datos.close()
```

2

3.

```
def guarda(dicci,arch):
  datos=open(arch,'w',encoding='utf-8')
  i=1
  for nom in dicci:
    datos.write(str(i)+','+dicci[nom].capitalize())
    datos.write(' '+nom[0].upper()+'.\n')
    i+=1
  datos.close()
```

X **3**

4.

```
def guarda(dicci,arch):
  datos=open(arch,'W',encoding='utf-8')
  for i in range(len(dicci)):
    datos.write(str(i)+','+dicci[i-1][1].capitalize())
    datos.write('\n'+dicci[i-1][0]+'\n')
  datos.close()
```

4

Ejercicio 0506 - 1 punto

Dado el siguiente DataFrame *lluvias*:

	localidad	mes	mm
0	azul	abril	65
1	charata	abril	96
2	azul	julio	30
3	azul	octubre	72
4	federación	enero	110
5	quitilipi	marzo	120
6	federación	marzo	115

Que contiene 7 filas y 3 columnas: localidad, mes y precipitación total registrada en mm (mm).

¿Qué instrucción produce la siguiente salida?

```
mes          localidad   mm

abril        charata      96
enero        federación  110
julio        azul         30
marzo        quitilipi   120
octubre      azul         72
```

1.	`lluvias.groupby(lluvias['mes'])['localidad','mm'].max()`	**X**	**1**
2.	`lluvias.head(3)`		**2**
3.	`lluvias.loc[lluvias.index[[1,3,5,2,6]]]`		**3**
4.	`lluvias.sort_values(by=['localidad','mes'], ascending=[True,True])`		**4**

Ejercicio 0606 - 1 punto

Dado el siguiente programa:

```
def obtieneNom(t):
  return 'ana' in t.lower().split()

nombres=['ana López','emiliano SAL','LORENA ana báunes',
         'analía Soto','ángelea FALCÓN',
         'Luciana analía pérez','Ana María Giménez']
resultado= . . .
for nom in resultado:
  print(nom)
```

Que produce la siguiente salida:

ana López
LORENA ana báunes
Ana María Giménez
>>>
Qué instrucción debería ir en los puntos suspensivos?
<u>Nota:</u> 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.	`list(filter(obtieneNom,nombres))`	**X**	**1**
2.	`reversed(nombres)`		**2**
3.	`obtieneNom(nombres)`		**3**
4.	`list(map(nombres,obtieneNom))`		**4**

Ejercicio 0706 - 1 punto

Dado el siguiente programa:

```
print('Ingresá números, < para terminar')
sigue=True
i=1
numeros=[]
while sigue:
  num=input(str(i)+': ')
  try:
    num=float(num)
    numeros.insert(0,num)
  except ValueError:
    if num=='<':
      sigue=False
  i+=1
```

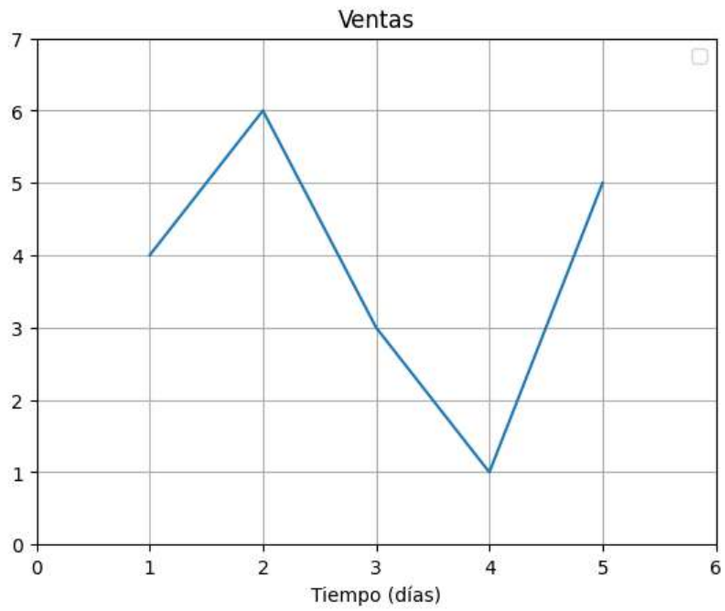
Y los siguientes ingresos:

```
1: 66
2: >>
3: -1
4: 23.66
5: <
```

¿Qué contenido tendrá *numeros* al finalizar?

1.	`[]`		**1**
2.	`[23.66, -1.0, 66.0]`	**X**	**2**
3.	`[66.0]`		**3**
4.	`[66, 0, -1, 23.66, '<']`		**4**

Ejercicio 0806 - 1 punto



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

1.	`ax.set_title("Ventas")`		**1**
2.	`ax.set_xlabel('Tiempo (días)')`		**2**
3.	`ax.set_ylabel('Ventas')`	**x**	**3**
4.	`ax.grid()`		**4**

Ejercicio 0906 - 1 punto

Se tiene el siguiente archivo llamado 'edades.txt':

```
Juan;20
Abril;22
Mario;21
```

¿Qué se va a imprimir luego de ejecutar el siguiente fragmento de código?

```
file = open('edades.txt', "r")
lines = file.readlines()
x = ''
for line in lines:
  x += line.split(';')[1]
print(x)
file.close()
```

1.	**63**		**1**
2.	**20 22 21**	**X**	**2**
3.	**Juan Abril Mario**		**3**
4.	**Se imprime algo diferente/Hay un error en el código y no se ejecuta**		**4**

Ejercicio 1006 - 2 puntos

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

```
def leer(arch):
  txtArch=open(arch,encoding='utf-8')
  txt=txtArch.read()
  txtArch.close()
  txtLis=txt.strip('\n').split()
  finVoc=[]
  termina='aeiouáéíóú'
  for pal in txtLis:
    if pal[-1] in termina:
      finVoc.append(pal)
  return finVoc

def guarda(lista,arch):
  datos=open(arch,'w',encoding='utf-8')
  for pal in lista:
    reves=pal[::-1].capitalize()
    datos.write(reves[::-1]+'\n')
  datos.close()

palabras=leer('parrafo.txt')
guarda(palabras,'terminaVoc.txt')
```

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

caminaba descalzo con cualquier pensamiento en la cabeza y pisó un clavo

1.	**con cualquier en y un**		**1**
2.	**caminabA descalzO pensamientO lA cabezA pisÓ clavO**	**X**	**2**
3.	**noc reiuqlauc ne y nu**		**3**
4.	**ABANIMACOZLACSEDOTNEIMASNEPALAZEBACÓSIPOVALC**		**4**

Ejercicio 1106 - 2 puntos

Dado el siguiente DataFrame *cobertura*:

	región	provincia	sucursales
0	cuyo	san juan	3
1	litoral	santa fé	0
2	cuyo	mendoza	5
3	patagonia	chubut	2
4	cuyo	san luis	1
5	pampa	buenos aires	8

Que contiene 6 filas y 3 columnas: región geográfica (región), provincia y cantidad de sucursales abiertas (sucursales).

¿Qué se muestra como resultado al ejecutar las siguientes instrucciones? La librería pandas ya ha sido importada con el alias pd.

```
filtrado=cobertura[(cobertura['región']!='cuyo')]
filtrado=pd.DataFrame(filtrado)
filtrado[['sucursales','provincia']]
```

1.		**X**	**1**

	sucursales	**provincia**
1	0	santa fé
3	2	chubut
5	8	buenos aires

2.		**2**

	provincia	**sucursales**
1	santa fé	0

3.		**3**

	región
1	litoral
3	patagonia
5	pampa

4.		**4**

	región	**provincia**	**sucursales**
0	cuyo	san juan	3
2	cuyo	mendoza	5
4	cuyo	san luis	1