

PENSAMIENTO COMPUTACIONAL (90)

UBA XXI
TEMA 4

EXAMEN: SEGUNDO PARCIAL

APELLIDO:	CALIFICACIÓN:
NOMBRE:	
DNI (registrado en SIU Guaraní):	
E-MAIL:	NOTA Y FIRMA DOCENTE (no rellenar)
TEL:	
AULA:	

Duración del examen: 1:30h.

- ✓ Escribir claramente el nombre en todas las páginas.
- ✓ El examen consta de 9 preguntas de opción múltiple.
- ✓ Cada pregunta tiene una y sólo una respuesta correcta.
- ✓ Las respuestas seleccionadas deben consignarse en la siguiente matriz de opciones.
- ✓ **Sólo se considerarán las respuestas anotadas en la matriz.**
- ✓ Las preguntas de la 1 a la 5 inclusive permiten acumular 1 punto (si son correctas), de la 6 a la 9 cada una acumula 2 puntos o 0.
- ✓ La nota final se calcula de acuerdo a la siguiente función:

Puntos	1 o 2	3 o 4	5 o 6	7	8	9	10	11	12	13
Nota	1	2	3	4	5	6	7	8	9	10

Matriz de Respuestas

	Ej 1 1 Pto	Ej 2 1 Pto	Ej 3 1 Pto	Ej 4 1 Pto	Ej 5 1 Pto	Ej 6 2 Ptos	Ej 7 2 Ptos	Ej 8 2 Ptos	Ej 9 2 Ptos	
1										1
2										2
3										3
4										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 esta última.



Talón de Control para el Alumno

	Ej 1 1 Pto	Ej 2 1 Pto	Ej 3 1 Pto	Ej 4 1 Pto	Ej 5 1 Pto	Ej 6 2 Ptos	Ej 7 2 Ptos	Ej 8 2 Ptos	Ej 9 2 Ptos	
1										1
2										2
3										3
4										4

Ej 01 T4 – 1 Pto

Para el siguiente DataFrame alumnos:

	apellido	nota	ejer realizados	año	carrera	prom	gral
0	Giunta	10	35	1		8.75	
1	Terra	2	5	2		7.00	
2	Kempes	10	18	2		6.25	
3	Ardiles	6	27	1		9.00	

¿Qué instrucción produce la siguiente salida?

7.0

1	alumnos.sort_values(by=['año','carrera','apellido'],ascending=[True,False])		1
2	alumnos.head(3)		2
3	alumnos['nota'].mean()	X	3
4	alumnos[['prom gral','apellido','ejer realizados']]		4

Ej 02 T4 – 1 Pto			
El archivo Cientes.txt contiene lo siguiente: Salazar srl.;20-33456876-2 Artelac;30-36925778-4 Novallen;30-124445879-3 starluz sa.;45896985			
¿Qué contenido tendrá el archivo nuevo.txt al finalizar la ejecución del siguiente programa?			
<pre>arch=open('Cientes.txt','r') lineas=arch.readlines() arch.close() arch=open('nuevo.txt','w') i=0 for lin in lineas: arch.write(lin.strip('\n')+';'+str(i)+'\n') i+=1 arch.close()</pre>			
1	1;Artelac;30-36925778-4 2;Novallen;30-124445879-3 3;Salazar srl.;20-33456876-2 4;starluz sa.;45896985		1
2	Artelac;30-36925778-4 Novallen;30-124445879-3 Salazar srl.;20-33456876-2 starluz sa.;45896985		2
3	Salazar srl.;20-33456876-2;0 Artelac;30-36925778-4;1 Novallen;30-124445879-3;2 starluz sa.;45896985;3	X	3
4	1 2 3 4		4

	Ej 03 T4 – 1 Pto		
	¿Qué muestra el siguiente programa? <pre>def edita(p): devuelve=p.count('O') devuelve+=p.count('Ó') return devuelve nom1=['ÁLVARO','CAROLINA','ESTEBAN','PEDRO'] nom2=list(map(edita,nom1)) print(*nom2)</pre> Notas: Usar un * antes de una lista en un print provoca que se muestran los elementos de la lista separados por blanco, sin los corchetes ni las comas Ej: <code>print(*[1,2,0]) -> 1 2 0</code>		
1	1 1 0 1	X	1
2	True True True True		2
3	Álvaro carolina esteban pedro		3
4	3		4

	Ej 04 T4 – 1 Pto		
	¿Cuál es la función adecuada para el siguiente programa? <pre>def traduce(...): - - - traductor={'mb':'MB', 'nv':'NV', 'mp':'MP', 'nc':'NC', 'rt':'RT', 'cr':'CR', 'pr':'PR'} txt='la conclusión final describió un completo desastre. Un combo emblemático de crisis.' print(traduce(traductor,txt))</pre> La salida del programa debe ser: la coNCLUSION final deSCRibió un coMPleto desastre. Un coMBo eMBlemático de CRisis.		
1	<pre>def traduce(d,t): for cl in d: t=t.replace(cl,d[cl]) return t</pre>	X	1
2	<pre>def traduce(txt,dicci): for cl in dicci: if cl in txt: txt.replace(cl,d[cl]) return dicci</pre>		2
3	<pre>def traduce(d): for cl in d: t.replace(cl,d[cl]) return d</pre>		3
4	<pre>def traduce(): txt=t for cl in d: if cl in txt: txt=txt.replace(cl,d[cl])</pre>		4

Ej 05 T4 – 1 Pto			
¿Cuál es el programa correcto que asegura el ingreso de un número entre -1.5 y 1.5 inclusive? El programa debe insistir en el ingreso hasta asegurarse que la variable num contenga un número real entre 1.5 y -1.5 inclusive.			
1	<pre>while not True: try: num=int(input('Ingresa un numero entre -1.5 y 1.5, inclusive: ')) if abs(num)<=1.5 or abs(num)>=0: print('Debe estar entre -1.5 y 1.5 inclusive') except ValueError: print('debe ser Número') print(num)</pre>		1
2	<pre>sigue=False while not sigue: try: num=float(input('Ingresa un numero entre -1.5 y 1.5, inclusive: ')) if abs(num)<=1.5 and abs(num)>=0: sigue=True else: print('Debe estar entre -1.5 y 1.5 inclusive') except ValueError: print('debe ser Número') print(num)</pre>	X	2
3	<pre>try: num=int(input('Ingresa un numero entre -1.5 y 1.5, inclusive: ')) if num<=1.5 and num>=-1.5: print('Ok') else: num=10 except ValueError: num=100 print(num)</pre>		3
4	<pre>num=True while num==True: try: num=float(input('Ingresa un numero entre -1.5 y 1.5, inclusive: ')) if abs(num)<=1.5 and abs(num)>0: num=False else: num=True except ValueError: num=True print(num)</pre>		4

Ej 06 T4 – 2 Ptos			
¿Qué muestra por pantalla el siguiente programa?			
<pre>def funcion(n): resp=False if int(n)!=n: resp=True return resp numeros=[3,6.75,7,8.01,15.66,1] nuevaLista=list(filter(funcion,numeros)) print(nuevaLista)</pre>			
1	[0, 0]		1
2	[6.75, 8.01, 15.66]	X	2
3	[]		3
4	[3, 7, 1]		4

Ej 07 T4 – 2 Ptos

¿Qué debería contener la lista **parcial** para que el siguiente programa funcione?

```
datos={'juan':['aguirre',41234225,301],  
       'maría':['iturre',43345445,301],  
       'elena':['lempert',42998007,13],  
       'ignacio':['lobos',41908700,11],}  
  
parcial=[...] #elegir el contenido correcto para la salida de abajo  
  
print('Lista Alumnos aula 301')  
for valor in parcial:  
    valor=valor.lower()  
    if datos[valor][2]==301:  
        nombre=valor+' '+datos[valor][0].upper()  
        print('DNI:',datos[valor][1],',',',',nombre)
```

La salida del programa debe ser:

```
Lista Alumnos aula 301  
DNI: 41234225 , juan AGUIRRE  
DNI: 43345445 , maría ITURRE
```

1	['0', '1']		1
2	[41234225,43345445,41908700]		2
3	[]		3
4	['ignacio','JUAN','María']	X	4

Ej 08 T4 – 2 Ptos

¿Qué muestra el siguiente programa?

```
def estacion(m):
    estaciones={1: 'verano',2: 'verano',3: 'verano-otoño',
                4: 'otoño',5: 'otoño',6: 'otoño-invierno',
                7: 'invierno',8: 'invierno',
                9: 'invierno-primavera',
                10: 'primavera',11: 'primavera',
                12: 'primavera-verano'}
    return 'primavera' in estaciones[m]

meses={1: 'Enero',2: 'Febrero',3: 'Marzo',
        4: 'Abril',5: 'Mayo',6: 'Junio',
        7: 'Julio',8: 'Agosto',9: 'Septiembre',
        10: 'Octubre',11: 'Noviembre',12: 'Diciembre'}
parciales=[5,6,9,10,11]
rendirPrimavera=list(filter(estacion,parciales))
for m in rendirPrimavera:
    print(meses[m])
```

1	Enero febrero marzo abril mayo junio julio agosto diciembre		1
2	11 10		2
3	Septiembre Octubre Noviembre	X	3
4	mayo, junio, septiembre, octubre, noviembre		4

Ej 09 T4 – 2 Ptos						
Para el siguiente DataFrame alumnos:						
	Apellido	nota	ejer realizados	año	carrera	prom gral
0	Giunta	10	35	1		8.75
1	Terra	2	5	2		7.00
2	Kempes	10	18	2		6.25
3	Ardiles	6	27	1		9.00
¿Qué salida produce la siguiente instrucción?						
alumnos.groupby('año carrera')['nota'].mean()						
1	apellido	nota	ejer realizados	año	carrera	prom gral
	0 Giunta	10	35	1		8.75
	1 Terra	2	5	2		7.00
	3 Ardiles	6	27	1		9.00
2	apellido	prom gral				
	0 Giunta	8.75				
	1 Terra	7.00				
	2 Kempes	6.25				
	3 Ardiles	9.00				
3	apellido	año carrera				
	1 Terra	2				
	2 Kempes	2				
4	nota					
	año carrera					
	1	8.0				
	2	6.0				
					X	4