

PENSAMIENTO COMPUTACIONAL (90)

UBA XXI

TEMA 1

EXAMEN: SEGUNDO PARCIAL

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

Duración del examen: 1:30h.

- ✓ Escribir claramente el nombre en todas las páginas.
- ✓ El examen consta de 10 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 7 inclusive permiten acumular 1 punto (si son correctas), de la 8 a la 10 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 1 Pto	Ej 7 1 Pto	Ej 8 2 Ptos	Ej 9 2 Ptos	Ej 10 2 Ptos	
1											1
2											2
3											3
4											4

iATENCIÓN! 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.

0101 – 1 Pto			
¿Cuál de los siguientes códigos valida adecuadamente que se ingrese un nombre iniciado con consonante? Debe detectar el error y garantizar un dato válido Ejs válidos: lucía, Pedro, JUAN Ejs inválidos: Úrsula, Inés, 9ema			
1	<pre>voc='AEIOUÁÉÍÓÜ' nom=input('Nombre que comience con consonante: ') if nom[1].isalpha() or nom[0] not in voc: print('Inválido') else: print('Vale')</pre>		1
2	<pre>voc='AEIOUÁÉÍÓÜ' nom=input('Nombre que comience con consonante: ') while not nom.isalpha(): nom=input('Nombre que comience con consonante: ') while nom.lower() not in voc: nom=input('Nombre que comience con consonante: ')</pre>		2
3	<pre>voc='AEIOUÁÉÍÓÜ' nom=input('Nombre que comience con consonante: ') while nom[0].upper() in voc or not nom.isalpha(): nom=input('Nombre que comience con consonante: ')</pre>	X	3
4	<pre>voc='AEIOUÁÉÍÓÜ' nom=input('Nombre que comience con consonante: ') while nom.isalpha() and nom[2] not in voc: nom=input('Nombre que comience con consonante: ')</pre>		4

0201 – 1 Pto			
¿Cuál es la salida correcta del siguiente programa?			
<pre>def nomMes(mes): meses={1:'ENE',3:'MAR',4:'ABR',2:'FEB',6:'JUN',5:'MAY', 7:'JUL',8:'AGO',9:'SEP',12:'DIC',10:'OCT',11:'NOV'} return meses[mes] #PPa1 meses=[11,5,4,6,7,8] nombres=list(map(nomMes,meses)) print(nombres)</pre>			
1	['diciembre', 'marzo']		1
2	['NOV', 'MAY', 'ABR', 'JUN', 'JUL', 'AGO']	X	2
3	[]		3
4	[9, 3, 12]		4

0301 – 1 Pto			
¿Cuál es la salida del siguiente programa?			
<pre>def habiles(d): finde=(6,7) return d not in finde #PPal # 1 es lunes y 7 es domingo dias=[1,5,4,6,2,7] hayBanco=list(filter(habiles,dias)) print(hayBanco)</pre>			
1	[]		1
2	[0, 0, 0, 0, 0, 0]		2
3	[1, 5, 4, 2]	X	3
4	6		4

0401 – 1 Pto

¿Qué contenido tendrá el archivo **nomAmigos.txt** al finalizar la ejecución del programa si el archivo **amigos.txt** tiene el siguiente contenido?

Contenido de **amigos.txt**:

Juan,Álvarez,juanjo12@gmail.com
Ana María,Paz,anitapaz@yahoo.com.ar
Rodo,García,rgarcia04@gmail.com

Programa a ejecutarse:

```
arch=open('amigos.txt','r')
agenda=arch.readlines()
arch.close()
nombres=[]
for amigo in agenda:
    nom=amigo.split(',')
    nombres.append(nom[0]+'\\n')
arch=open('nomAmigos.txt','w')
arch.writelines(nombres)
arch.close()
```

Nota:
El método **split()** devuelve una lista con las partes de un texto tomando como separador el argumento

Ej:
'yo soy argentina'.split(' ') -> ['yo', 'soy', 'argentina']

1	Juan Álvarez juanjo12@gmail.com		1
2	ÁLVAREZ PAZ GARCÍA		2
3	Juan Ana María Rodo	X	3
4	juanjo12@gmail.com ,anitapaz@yahoo.com.ar ,rgarcia04@gmail.com		4

0501 – 1 Pto			
¿Qué debe ser estructura para que la siguiente instrucción se ejecute sin problemas?			
<code>estructura[1]='Hola'</code>			
1	Una lista vacía		1
2	Una lista de al menos 2 elementos	X	2
3	Una tupla de al menos 2 elementos		3
4	Una string		4

0601 – 1 Pto			
Con un tablero de Batalla Naval de 4x4 con 7 barcos de un casillero cargados (X es barco; blanco es agua). ¿Cuál función impide que aborte la ejecución del programa si los ingresos son fila=2 y columna=30 ?			
<pre>def tiro(...): - - - tablero=[[' ','X','X',' '],[' ',' ',' ',' '], ['X',' ','X','X'],['X',' ',' ','X']] print('Batalla naval') fila=int(input('Fila: ')) col=int(input('Columna: ')) print(tiro(tablero,fila,col))</pre>			
1	<pre>def tiro(t,i,j): except: casillero=t[i][j] if casillero==X: resp='Hundido' else: resp='Agua' else: resp='Fuera del Mar' return resp</pre>		1
2	<pre>def tiro(t,i,j): casillero=t[i][j] if casillero=='X': resp='Hundido' else: resp='Agua'</pre>		2
3	<pre>def tiro(t,i,j): try: casillero=t[i][j] if casillero=='X': resp='Hundido' else: resp='Agua' except: resp='Fuera del Mar' return resp</pre>	X	3
4	<pre>def tiro(): try: casillero=t[i][j] if casillero=='X': resp='Hundido' else: resp='Agua' elif i<j: resp='Fuera del Mar'</pre>		4

0701 – 1 Pto							
Para el DataFrame vue de pandas, que contiene:							
	nombre	dni	fila	asiento	tarifa		
0	Paz, Orlando	25889145	14	C	flex		
1	Manzur, Jimena	15226223	11	D	promo		
2	Hermenegildo, Juan	19002136	6	A	promo		
3	Decquer, Mariqana	41100107	22	C	base		
4	Tasz, Martín	36214875	14	F	flex		
¿Qué contendrá vue1 después de la siguiente operación?							
vue1=vue.drop(2)							
1	dni	tarifa					
	0	25889145	flex				
	1	15226223	promo				
	2	19002136	promo				
	3	41100107	base				
	4	36214875	flex				
2	nombre	dni	tarifa				
	0	Paz, Orlando	25889145	flex			
	1	Manzur, Jimena	15226223	promo			
	2	Hermenegildo, Juan	19002136	promo			
	3	Decquer, Mariqana	41100107	base			
3	nombre	dni	fila	asiento	tarifa	X	
	0	Paz, Orlando	25889145	14	C		flex
	1	Manzur, Jimena	15226223	11	D		promo
	3	Decquer, Mariqana	41100107	22	C		base
	4	Tasz, Martín	36214875	14	F		flex
4	nombre	dni	fila	asiento	tarifa		
	0	Paz, Orlando	25889145	14	C		flex
	3	Decquer, Mariqana	41100107	22	C		base
						4	

0801 – 2 Ptos			
¿Cuáles modos de apertura deben emplearse con el archivo notas.txt en el siguiente programa para que funcione correctamente?			
<pre>arch=open('notas.txt',...) #primer open() datos=arch.readlines() print(datos) arch.close() arch=open('notas.txt', ...) #segundo open() datos.append('F\n') for fila in datos: arch.write(fila) arch.close()</pre>			
1	Primer open() 'r' Segundo open() 'r'		1
2	Primer open() 'a' Segundo open() 'w'		2
3	Primer open() 'r+' Segundo open() 'r'		3
4	Primer open() 'r' Segundo open() 'w'		X4

0901 – 2 Ptos																																					
Para el DataFrame vue de pandas, que contiene:																																					
	<table><tr><td></td><td>nombre</td><td>dni</td><td>fila</td><td>asiento</td><td>tarifa</td></tr><tr><td>0</td><td>Paz, Orlando</td><td>25889145</td><td>14</td><td>C</td><td>flex</td></tr><tr><td>1</td><td>Manzur, Jimena</td><td>15226223</td><td>11</td><td>D</td><td>promo</td></tr><tr><td>2</td><td>Hermenegildo, Juan</td><td>19002136</td><td>6</td><td>A</td><td>promo</td></tr><tr><td>3</td><td>Decquer, Mariqana</td><td>41100107</td><td>22</td><td>C</td><td>base</td></tr><tr><td>4</td><td>Tasz, Martín</td><td>36214875</td><td>14</td><td>F</td><td>flex</td></tr></table>		nombre	dni	fila	asiento	tarifa	0	Paz, Orlando	25889145	14	C	flex	1	Manzur, Jimena	15226223	11	D	promo	2	Hermenegildo, Juan	19002136	6	A	promo	3	Decquer, Mariqana	41100107	22	C	base	4	Tasz, Martín	36214875	14	F	flex
	nombre	dni	fila	asiento	tarifa																																
0	Paz, Orlando	25889145	14	C	flex																																
1	Manzur, Jimena	15226223	11	D	promo																																
2	Hermenegildo, Juan	19002136	6	A	promo																																
3	Decquer, Mariqana	41100107	22	C	base																																
4	Tasz, Martín	36214875	14	F	flex																																
¿Qué operación produce el siguiente resultado?																																					
	<table><tr><td></td><td>nombre</td><td>dni</td><td>fila</td><td>asiento</td><td>tarifa</td></tr><tr><td>2</td><td>Hermenegildo, Juan</td><td>19002136</td><td>6</td><td>A</td><td>promo</td></tr><tr><td>3</td><td>Decquer, Mariqana</td><td>41100107</td><td>22</td><td>C</td><td>base</td></tr><tr><td>4</td><td>Tasz, Martín</td><td>36214875</td><td>14</td><td>F</td><td>flex</td></tr></table>		nombre	dni	fila	asiento	tarifa	2	Hermenegildo, Juan	19002136	6	A	promo	3	Decquer, Mariqana	41100107	22	C	base	4	Tasz, Martín	36214875	14	F	flex												
	nombre	dni	fila	asiento	tarifa																																
2	Hermenegildo, Juan	19002136	6	A	promo																																
3	Decquer, Mariqana	41100107	22	C	base																																
4	Tasz, Martín	36214875	14	F	flex																																
1	<code>vue.head(1)</code>		1																																		
2	<code>vue.loc[:, ['fila', 'asiento']]</code>		2																																		
3	<code>vue[vue['tarifa'].isnull()]</code>		3																																		
4	<code>vue.iloc[2:]</code>	X	4																																		

1001 – 2 Ptos En el siguiente programa: <pre>txt='Me temo que no es NECESARIO TODO ' contLetras={} for letra in txt: if letra.isalpha(): if letra.lower() in contLetras: # línea a completar else: contLetras[letra.lower()]=1 print(txt) print('Lista de letras y ocurrencias') for letra in contLetras: print(letra+'/' +letra.upper(),':',contLetras[letra])</pre> Que cuenta las ocurrencias en el texto de cada una de sus letras (ignorando mayúsculas y minúsculas '). ¿Cuál debería ser la línea faltante en el código? La salida final debería ser: Me temo que será NECESARIO COMPRENDER bien todo Lista de letras y ocurrencias m/M : 3 e/E : 9 t/T : 2 o/O : 5 q/Q : 1 u/U : 1 s/S : 2 r/R : 4 á/Á : 1 n/N : 3 c/C : 2 a/A : 1 i/I : 2 p/P : 1 d/D : 2 b/B : 1			
1	Letra = letra+1		1
2	contLetras[letra] = letra-1		2
3	contLetras[letra.lower()] = contLetras[letra.lower()]+1	X	3
4	contLetras[letra.lower()] = 0		4



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 1 Pto	Ej 7 1 Pto	Ej 8 2 Ptos	Ej 9 2 Ptos	Ej 10 2 Ptos	
1											1
2											2
3											3
4											4