

## PENSAMIENTO COMPUTACIONAL (90)

UBAXXI

## TEMA 4

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
<b>Nota</b>	<b>1</b>	<b>2</b>	<b>3</b>	<b>4</b>	<b>5</b>	<b>6</b>	<b>7</b>	<b>8</b>	<b>9</b>	<b>10</b>

## 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	
<b>1</b>											<b>1</b>
<b>2</b>											<b>2</b>
<b>3</b>											<b>3</b>
<b>4</b>											<b>4</b>

**¡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.

0104 – 1 Pto			
<p>¿Cuál de los siguientes códigos valida adecuadamente que se ingrese un nombre sin tilde? Debe detectar el error y garantizar un dato válido <b>Ejs válidos: lucaS, Pedro, JUAN</b> <b>Ejs inválidos: Úrsula, Inés, 9ema</b></p> <p><b>Nota:</b> El método <i>isalpha()</i> devuelve <b>True</b> si la string contiene sólo letras, caso contrario devuelve <b>False</b> <b>Ej:</b> <code>'ana'.isalpha() -&gt; True</code> y <code>'Horacio8'.isalpha() -&gt; False</code></p>			
1	<pre>def tilde(n):     voc='ÁÉÍÓÚ'     marca=False     for letra in n.lower():         if letra in voc:             marca=True     return True  #PPal nom=input('Nombre sin acentos: ') while tilde(nom) or not nom.isalpha():     nom=input('Nombre sin acentos: ')</pre>		1
2	<pre>def tilde(n):     voc='ÁÉÍÓÚ'     for letra in voc:         if letra in n.lower():             marca=True         else:             marca=False     return marca  #PPal nom=input('Nombre sin acentos: ') while tilde(nom) or not nom.isalpha():     nom=input('Nombre sin acentos: ')</pre>		2
3	<pre>def tilde(n):     voc='ÁÉÍÓÚ'     n=n.upper()     marca=False     for letra in voc:         if letra in n:             marca=True     return marca  #PPal nom=input('Nombre sin acentos: ') while not(tilde(nom)==False and nom.isalpha()):     nom=input('Nombre sin acentos: ')</pre>	X	3
4	<pre>def tilde(n):     voc='ÁÉÍÓÚ'     marca=False     for letra in n.upper():         if letra in voc:             marca=True     return marca  #PPal nom=input('Nombre sin acentos: ') if nom.isalpha() or tilde(nom):     print('Válido') else:     print('Inválido')</pre>		4

**0204 – 1 Pto**

¿Cuál es la salida correcta del siguiente programa?

```
def nomMes (mes) :

meses={1:'enero',3:'marzo',4:'abril',2:'febrero',6:'junio',5:'mayo',
7:'julio',8:'agosto',9:'septiembre',12:'diciembre',10:'octubre',11:'n
oviembre'}
    return meses[mes].upper()

#PPal
meses=[10,5,2,6,7,9]
nombres=list(map(nomMes,meses))
print(nombres)
```

**Nota:**

El método **upper()** devuelve el texto todo en mayúsculas

Ej:

a='Hola'

a.upper() -> 'HOLA'

1	['JUNIO']		1
2	[0, 0, 0, 0, 0, 0]		2
3	[9, 7, 6, 2, 5, 10]		3
4	['OCTUBRE', 'MAYO', 'FEBRERO', 'JUNIO', 'JULIO', 'SEPTIEMBRE']	X	4

**0304 – 1 Pto**

¿Cuál es la salida del siguiente programa?

```
def noLaborable(d) :
    finde=(6,7)
    return d in finde

#PPal
# 1 es lunes y 7 es domingo
dias=[1,5,4,6,2,7]
sinBanco=list(filter(noLaborable,dias))
print(sinBanco)
```

1	[1, 2]		1
2	'jueves'		2
3	[6, 7]	X	3
4	[]		4

<b>0404 – 1 Pto</b>			
<p>¿Qué contenido tendrá el archivo <b>destacados.txt</b> al finalizar la ejecución del programa si el archivo <b>deportistas.txt</b> tiene el siguiente contenido?</p> <p>Contenido de <b>deportistas.txt</b>:</p> <p>lionel,messi,futbol gaby,sabattini,tenis rafa,nadal,tenis chapu,nocioni,basquet</p> <p>Programa a ejecutarse:</p> <pre> arch=open('deportistas.txt','r') lista=arch.readlines() arch.close() nombres=[] for dep in lista:     depor=dep.split(',')     nombres.append(depor[1]+' '+depor[0]+'\\n') arch=open('destacados.txt','w') arch.writelines(nombres) arch.close()                     </pre> <p><b>Nota:</b> El método <b>split()</b> devuelve una lista con las partes de un texto tomando como separador el argumento</p> <p><b>Ej:</b> <b>'yo soy argentina'.split(' ') -&gt; ['yo', 'soy', 'argentina']</b></p>			
<b>1</b>	<b>messi***** sabattini***** nadal***** nocioni*****</b>	X	<b>1</b>
<b>2</b>	<b>MESSI lionel SABATTINI gaby NADAL rafa NOCIONI chapu</b>	<b>X</b>	<b>2</b>
<b>3</b>	<b>Lionel,, gaby,, rafa,, chapu,,</b>	<b>x</b>	<b>3</b>
<b>4</b>	<b>lionel,messi,futbol gaby,sabattini,tenis rafa,nadal,tenis chapu,nocioni,basquet</b>	X	<b>4</b>

Por error de enunciado, se toman todas como correctas

<b>0504 – 1 Pto</b>			
<p>¿Qué <b>no</b> debería ser <b>estructura</b> para que la siguiente instrucción se ejecute sin problemas?</p> <p><code>dato=estructura[0][1]</code></p>			
<b>1</b>	<b>Una lista de strings</b>		<b>1</b>
<b>2</b>	<b>Una lista de tuplas</b>		<b>2</b>
<b>3</b>	<b>Una tupla de strings</b>		<b>3</b>
<b>4</b>	<b>Una string</b>	<b>X</b>	<b>4</b>

<b>D604 – 1 Pto</b>			
<p>Para averiguar el camarote de un viaje en barco asignado a un pasajero se dispone del siguiente programa. ¿Cuál función impide que aborte la ejecución del mismo si se ingresa el nombre <b>pedro</b>?</p> <pre>def cabina (...): - - -  barco={'ANA':15,'ULISES':2,'FLORA':1,       'IGNACIO':15, 'AGUSTINA':17,'ALICIA':22} print('Averiguá en qué camarote viaja') pasajero=input('Nombre: ') print(pasajero,cabina(barco,pasajero))</pre>			
<b>1</b>	<pre>def cabina(nom,nom):     except:         camarote=b[nom]         cartel='viaja en camarote '+str(camarote)     try:         cartel='NO VIAJA EN ESTE BARCO'     return cartel</pre>		<b>1</b>
<b>2</b>	<pre>def cabina(b,nom):     try:         camarote=b[nom.upper()]         cartel='viaja en camarote '+str(camarote)     except:         cartel='NO VIAJA EN ESTE BARCO'     return cartel</pre>	<b>X</b>	<b>2</b>
<b>3</b>	<pre>def cabina(b,nom):     camarote=b[nom]     try:         cartel='viaja en camarote '+str(camarote)     return nom</pre>		<b>3</b>
<b>4</b>	<pre>def cabina(b,b):     try:         camarote=b[nom]         cartel='viaja en camarote '+str(camarote)     else:         cartel='NO VIAJA EN ESTE BARCO'</pre>		<b>4</b>

0704 – 1 Pto						
Para el DataFrame <b>vue</b> 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 <b>vue</b> después de la siguiente operación?						
<code>vue.loc[5]=[ 'Heinz, Armando',12558741,3, 'A', 'flex']</code>						
1	nombre	dni	fila	asiento	tarifa	1
	1 Manzur, Jimena	15226223	11	D	promo	
	2 Hermenegildo, Juan	19002136	6	A	promo	
	4 Tasz, Martín	36214875	14	F	flex	
2	dni	tarifa				2
	0 25889145	flex				
	1 15226223	promo				
	2 19002136	promo				
	3 41100107	base				
	4 36214875	flex				
3	nombre	dni	tarifa			3
	0 Paz, Orlando	25889145	flex			
	1 Manzur, Jimena	15226223	promo			
	2 Hermenegildo, Juan	19002136	promo			
	3 Decquer, Mariqana	41100107	base			
4	nombre	dni	fila	asiento	tarifa	X 4
	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	
	5 Heinz, Armando	12558741	3	A	flex	

0804 – 2 Ptos			
¿Cuáles modos de apertura deben emplearse con el archivo <b>notas.txt</b> en el siguiente programa para que no salte error?			
<pre> arch=open('notas.txt',...) #primer open() filas=['1\n','2\n','3\n'] arch.writelines(filas) arch.close()  arch=open('notas.txt',...) #segundo open() arch.write('Fin\n') arch.close()                     </pre>			
1	Primer open() 'r'	Segundo open() 'r+'	1
2	Primer open() 'r'	Segundo open() 'w'	2
3	Primer open() 'w'	Segundo open() 'w'	X 3
4	Primer open() 'a'	Segundo open() 'r'	4

0904 – 2 Ptos			
Para el DataFrame <b>vue</b> 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 operación produce el siguiente resultado?			
asiento			
A	6		
C	22		
D	11		
F	14		
1	vue[vue['tarifa'].isnull()]		1
2	vue.loc[:, ['dni', 'tarifa']]		2
3	vue.groupby('asiento')['fila'].max()		X 3
4	vue.iloc[2:]		4

1004 – 2 Ptos			
En el siguiente programa:			
<pre>txt='Me picaron MAS mosquitos QUE EN EL VERANO' traductor={'e':''),'o':'%','a':'*','i':'/','            'u':'@','m':'&amp;',' ':'+-'} for letra in txt:     if letra in traductor:         ..... # línea a completar print(txt)</pre>			
Que traduce los caracteres del texto que están definidos en <b>traductor</b> . ¿Cuál debería ser la línea faltante en el código? La salida final debería ser:			
M) +-p/c*r%n+-MAS+-&%sq@/t%s+-QUE+-EN+-EL+-VERANO			
1	txt.replace(traductor[letra], '1')		1
2	letra[1]=traductor[1]		2
3	txt=txt.replace(letra,traductor[letra])		X 3
4	letra=traductor[letra]		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