PENSAMIENTO COMPUTACIONAL (90) .UBAXXI

TEMA 2

EXAMEN: RECUPERATORIO 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.

shutterstock.com · 2348912687

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**

Recup 2do Parcial - Ej 01 T2 – 1 Pto

Para el siguiente DataFrame autos:

	marca	modelo	color	precio mill
0	nissan	kicks	rojo	25
1	ford	bronco	plata	55
2	nissan	kicks	rojo	27
3	ford	territory	azul	53
4	chevrolet	onix	plata	31
5	nissan	frontier	blanco	64

¿Qué instrucción produce la siguiente salida?

	marca	modelo	color	precio mill
4	chevrolet	onix	plata	31
3	ford	territory	azul	53
1	ford	bronco	plata	55
0	nissan	kicks	rojo	25
2	nissan	kicks	rojo	27
5	nissan	frontier	blanco	64

1	`autos.head(1)`		1
2	`autos.sort_values(by=['marca','modelo'],ascending=[True,False])`	X	2
3	`autos[['color','modelo']]`		3
4	`autos.groupby('marca')['precio mill'].max()`		4

Recup 2do Parcial - Ej 02 T2 – 1 Pto

El archivo **carrera.txt** contiene lo siguiente:

```
Quimica
Analia,Ricardes,2021
Juliana,Ortesano,2023
Pedro,Fuentes,2023
Industrial
Jose,Agustoni,2021
Ana,Mauner,2022
Andrea,Carragno,2022
Civil
Marta,Leiva,2021
```

¿Qué contenido tendrá el archivo **quimica.txt** al finalizar la ejecución del siguiente programa?

```
carreras=['Quimica','Industrial','Civil']
arch=open('carrera.txt')
lineas=arch.readlines()
arch.close()
arch=open('quimica.txt','w')
i=0
while i<len(lineas):
    lin=lineas[i]
    if lin.strip('\n')=='Quimica':
        j=i+1
        i=len(lineas)
        while j<len(lineas):
            lin=lineas[j]
            if lin.strip('\n') in carreras:
                j=len(lineas)+1
            else:
                arch.write(lin)
                j+=1
    i+=1
arch.close()
```

1	Quimica Analia,Ricardes,2021 Juliana,Ortesano,2023 Pedro,Fuentes,2023 Industrial Jose,Agustoni,2021 Ana,Mauner,2022 Andrea,Carragno,2022 Civil Marta,Leiva,2021		**1**
2	Quimica Industrial Civil		**2**
3	Analia,Ricardes,2021 Juliana,Ortesano,2023 Pedro,Fuentes,2023	**X**	**3**
4	Jose,Agustoni,2021 Ana,Mauner,2022 Andrea,Carragno,2022		**4**

Recup 2do Parcial - Ej 03 T2 – 1 Pto

¿Qué muestra el siguiente programa?

```
def edita(n):
    desde=n[0]
    hasta=n[1]
    txt='Tengo una vaca lechera. No es una vaca cualquiera'
    p=txt[desde:hasta]
    return p

nom1=[(10,14),(24,26),(27,29)]
nom2=list(map(edita,nom1))
print(*nom2)
```

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:
print(*[1,2,0]) -> **1 2 0**

Usar **[desde:hasta]** con una string permite tomar porciones. Si no se pone **desde** va desde el principio y si no se pone **hasta** va hasta el final
Ejs:
'Un ejemplo común' [3:6] ->**'eje '**
'Un ejemplo común' [:6] ->**'Un eje'**

1	`una`		**1**
2	`tengounavacaleche`		**2**
3	`NGO TE LECHE`		**3**
4	`vaca No es`	X	**4**

Recup 2do Parcial - Ej 04 T2 – 1 Pto

¿Cuál es la función adecuada para el siguiente programa?

```
def magnitudes(…):
  -
  -
  -

cambios={'cm':'m',
         'gr':'kg',
         '789':'7,89',
         '300':'0,30',
         '2000':'2',
         '5670':'5,67'}
lineas=['Lista de Ventas:','789 cm varilla curva X9',
     '300 cm varilla X10','2000 gr cal']
for l in lineas:
    print(magnitudes(cambios,l))
```

La salida del programa debe ser:

```
Lista de Ventas:
7,89 m varilla curva X9
0,30 m varilla X10
2 kg cal
```

1	`def magnitudes(d,t):`
	`  for cl in d:`
	`  t.replace(d[cl],cl)`
	`  return cl`
			1

2	`def magnitudes():`
	`  for cl in dicci:`
	`  t=dicci.replace(cl,d[cl])`
	`  return dicci`
			2

3	`def magnitudes(d,t):`
	`  for cl in d:`
	`  t=t.replace(cl,d[cl])`
	`  return t`
		X	3

4	`def magnitudes(d):`
	`  for cl in range(len(d)):`
	`  t=t.replace(cl,d[cl])`
	`  return d,t,2`
			4

Recup 2do Parcial - Ej 05 T2 – 1 Pto

¿Cuál es la función adecuada para el siguiente programa? Recordar que el método **index()** produce un fallo si no se encuentra el dato que se busca en la lista. El programa debe informar esa situación, sin abortar la ejecución.

```
def posicion(…):
  -
  -
  -

numeros=[1,9,3,5,0]
for num in range(5):
  print(posicion(num,numeros))
```

La salida del programa debe ser:

```
4
0
No está
2
No está
```

1

```
def posicion(n,numeros):
  except:
    pos=numeros.index(n)
  try:
    return pos
```

2

```
def posicion(n,numeros):
  if:
    pos=numeros.index(n)
  else:
    pos='No está'
  return pos
```

3 **X**

```
def posicion(n,numeros):
  try:
    pos=numeros.index(n)
  except:
    pos='No está'
  return pos
```

4

```
def posicion(n,numeros):
  pos=numeros.index(n)
  return pos
```

Recup 2do Parcial - Ej 06 T2 – 2 Ptos

¿Qué muestra por pantalla el siguiente programa?

```
def funcion(p):
    resp=False
    if 'a' not in p:
        resp=True
    return resp

palabras=['hola','camarada','mascarada','mesaza','lustro']
nuevaLista=list(filter(funcion,palabras))
print(nuevaLista)
```

1	`['camarada', 'mascarada']`		**1**
2	`[]`		**2**
3	`['lustro']`	**X**	**3**
4	`['C', 'M']`		**4**

Recup 2do Parcial - Ej 07 T2 – 2 Ptos

¿Cuál diccionario **tiempo** funcionará correctamente para el siguiente programa?

```
tiempo={…}  #indicar cuál funcionará para la salida de abajo

txt=['Hoy la temperatura mínima será de -MIN-, y la máxima de -MAX-',
     'Se esperan vientos -VIENTOS-, con cielo -CIELO-',
     'La probabilidad de lluvias es de -LLUVIA-']


for t in txt:
    for val in tiempo:
        t=t.replace(val,tiempo[val])
    print(t)
```

La salida del programa debe ser:

```
Hoy la temperatura mínima será de 12 grados, y la máxima de 19 grados
Se esperan vientos del este, con cielo despejado
La probabilidad de lluvias es de 12 %
```

1	`{ '-VIENTOS-':'del este',`
	`  '-CIELO-':'despejado',`
	`  '-MIN-':'12 grados',`
	`  '-MAX-': '19 grados',`
	`  '-LLUVIA-':'12 %' }`
		X	**1**

2	`{1: 'lluvia ',` ` 0: 'sin lluvia'}`		**2**

3	`{ 0:['12 grados'],`
	`  1:['19 grados'],`
	`  2:['del este'],`
	`  3:['despejado'],`
	`  4:['12 %'] }`
			3

4	`{ '<min>':'12 grados',`
	`  '<max>': '19 grados',`
	`  '<vientos>':'del este',`
	`  '<cielo>':'despejado',`
	`  '<lluvia>':'12 %' }`
			4

Recup 2do Parcial - Ej 08 T2 – 2 Ptos

¿Qué muestra el siguiente programa?

```
def elige(b):
  return 'e' in b

def prepara(b):
  return b[0].upper()+'-'+b[1:]

biomas=['sabana','selva','desierto','montaña',
          'sierra','estepa','bosque']
seleccion=list(filter(elige,biomas))
editados=list(map(prepara,seleccion))
print(*editados)
```

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:
print(*[1,2,0]) -> **1 2 0**

Usar **[desde:hasta]** con una string permite tomar porciones. Si no se pone **desde** va desde el principio y si no se pone **hasta** va hasta el final
Ejs:
'Un ejemplo común' [3:6] ->**'eje '**
'Un ejemplo común' [:6] ->**'Un eje'**

1	`S-elva D-esierto S-ierra E-stepa B-osque`	X	**1**
2	`S-abana S-elva M-ontaña S-ierra E-stepa`		**2**
3	`S-S-M-S-E-`		**3**
4	`montaña` `sabana`		**4**

Recup 2do Parcial - Ej 09 T2 – 2 Ptos

Para el siguiente DataFrame autos:

```
       marca         modelo       color   precio mill
0      nissan        kicks        rojo    25
1      ford          bronco       plata   55
2      nissan        kicks        rojo    27
3      ford          territory    azul    53
4      chevrolet     onix         plata   31
5      nissan        frontier     blanco  64
```

¿Qué salida produce la siguiente instrucción?

```
autos.groupby('modelo')['precio mill'].mean()
```

1	`precio mill`
	`modelo`
	`bronco 55.0`
	`frontier 64.0`
	`kicks 26.0`
	`onix 31.0`
	`territory 53.0`
	`dtype: float64`
		X	**1**

2	`count`
	`marca color`
	`chevrolet plata 1`
	`ford azul 1`
	`plata 1`
	`nissan rojo 2`
	`blanco 1`
	`dtype: int64`
			2

3	`marca modelo color precio mill`
	`1 ford bronco plata 55`
	`2 nissan kicks rojo 27`
			3

4	`color modelo`
	`0 rojo kicks`
	`1 plata bronco`
	`2 rojo kicks`
	`3 azul territory`
	`4 plata onix`
	`5 blanco frontier`
			4