PENSAMIENTO COMPUTACIONAL (90) .UBAXXI

TEMA 6

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 T6 – 1 Pto

Para el siguiente DataFrame mascotas:

```
      Nombre  Clase  Edad   Dueño
0      Pipo   loro   7      Juan
1      Tato   gato   5      Jimena
2      Kempes gato   10     Lara
3      Tasso  perro  2      Hernán
```

¿Qué instrucción produce la siguiente salida?

```
       Clase
0      LORAZO
1      MIAU
2      MIAU
3      ROPE
dtype: object
```

1	`mascotas.iloc[1:]`		**1**
2	`mascotas.drop(columns=['Edad'])`		**2**
3	`mascotas['Clase'].value_counts()`		**3**
4	`mascotas['Clase'].map({'loro':'LORAZO','gato':'MIAU','perro':'ROPE'})`	X	**4**

Ej 02 T6 – 1 Pto

El archivo **Clientes.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?

```
arch=open('Clientes.txt','r')
lineas=arch.readlines()
arch.close()
arch=open('nuevo.txt','w')
lineas.sort()
for lin in lineas:
    datos=lin.split(';')
    lin=datos[1]
    arch.write(lin)
arch.close()
```

1	**Artelac** **Novallen** **Salazar srl.** **starluz sa.**		**1**
2	**30-36925778-4;Artelac** **30-124445879-3;Novallen** **20-33456876-2;Salazar srl.** **45896985;starluz sa.**		**2**
3	**30-36925778-4** **30-124445879-3** **20-33456876-2** **45896985**	**X**	**3**
4	**45896985;starluz sa.**		**4**

Ej 03 T6 – 1 Pto

¿Qué muestra el siguiente programa?

```
def edita(p):
    p=p.lower()
    if 'luc' in p:
        devuelve=p.upper()
    else:
        devuelve=p
    return devuelve

nom1=['Luciana','Lucía','lucas','JUAN']
nom2=list(map(edita,nom1))
print(*nom2)
nom1=['Luciana','Lucía','lucas','JUAN']
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**

1	`iana ía as juan`		**1**
2	`LUCIANA LUCÍA LUCAS juan`	X	**2**
3	`LUC LUC LUC`		**3**
4	`True True True False`		**4**

* Líneas en exceso informadas durante el parcial. No cambia el resultado final.

Ej 04 T6 – 1 Pto

¿Cuál es la función adecuada para el siguiente programa?

```
def telegrama(…):
  -
  -
  -

cambios={'final':'',
         'completo':'fenomenal',
         'un':'',
         'crisis':'SOS',
         'la ':'',
         '.':''}
txt='la conclusión final describió un completo desastre. Un combo emblemático
de crisis.'
print(telegrama(txt,cambios))
```

La salida del programa debe ser:

```
conclusión  describió  fenomenal desastre  combo emblemático de SOS
```

1	`def telegrama(t,d):`
	`  t=t.lower()`
	`  for cl in d:`
	`  t=t.replace(cl,d[cl])`
	`  return t`
		X	**1**

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

3	`def telegrama(t,d,cl):`
	`  t=t.lower()`
	`  for cl in d:`
	`  t=t.replace(cl,d[cl])`
			3

4	`def telegrama():`
	`  t=t.lower()`
	`  for cl in d:`
	`  remplazo=d[cl].upper()`
	`  t=t.replace(cl,remplazo)`
	`  return cl`
			4

Ej 05 T6 – 1 Pto

¿Cuál es el programa correcto que asegura el ingreso de un número entero entre -15 y 15 inclusive?
El programa debe insistir en el ingreso hasta asegurarse que la variable **num** contenga un número entero entre 15 y -15 inclusive.

1

```
intenta=False
while not intenta:
  try:
    num=int(input('Ingresa un numero entero entre -15
                   y 15, inclusive: '))
    if num in range(-15,16):
      intenta=True
    else:
      print('No está en el rango -15 a 15')
  except ValueError:
    print('Debe ser un entero')
print(num)
```

X **1**

2

```
while False:
  try:
    num=int(input('Ingresa un numero entero entre
                  -15 y 15, inclusive: '))
    if num in range(-15,16):
      print('ok')
  except ValueError:
    print('Debe ser un entero')
  intenta=False
print(num)
```

2

3

```
intenta=True
while intenta:
  try:
    num=int(input('Ingresa un numero entero entre
                  -15 y 15, inclusive: '))
    if num not in range(-15,16):
      print('No está en el rango -15 a 15')
  except ValueError:
    print('Debe ser un entero')
  intenta=False
print(num)
```

3

4

```
intenta=True
while intenta:
  try:
    num=int(input('Ingresa un numero entero entre
                 -15 y 15, inclusive: '))
    if num in range(-15,16):
      intenta=True
    else:
      intenta=False
      print('No está en el rango -15 a 15')
  except ValueError:
    print('Debe ser un entero')
    intenta=False
print(num)
```

4

Ej 06 T6 – 2 Ptos

¿Qué muestra por pantalla el siguiente programa?

```
def funcion(n):
    resp=False
    if n+10!=10 and  n*10!=10:
        resp=True
    return resp

numeros=[3,0,7,8.01,15.66,1]
nuevaLista=list(filter(funcion,numeros))
print(nuevaLista)
```

1	`[3, 7, 8.01, 15.66]`	X	**1**
2	`[3, 0, 7, 8.01, 15.66, 1]`		**2**
3	`[0, 1]`		**3**
4	`[]`		**4**

Ej 07 T6 – 2 Ptos

¿Cuál diccionario **`valores`** funcionará correctamente para el siguiente programa?

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

txt='Se abonaron 12900 por 45 de arroz fino. Costo envío: 1110'

for valor in valores:
    valorTxt=str(valor)
    cambio=valores[valor][0]+' '+valores[valor][1]
    txt=txt.replace(valorTxt,cambio)
print(txt)
```

La salida del programa debe ser:

```
Se abonaron doce mil novecientos pesos por cuarenta y cinco kg de arroz fino.
Costo envío: mil ciento diez pesos
```

1	`{'cm':['docientos cincuenta y 6','cm'],`
	`'kg':['cuarenta y cinco','kg'],`
	`'pesos':['mil ciento diez','pesos'],`
	`'pesos':['doce mil novecientos','pesos']}`
			1

2	`{ 0:['docientos cincuenta y 6','cm'],`
	`1:['cuarenta y cinco','kg'],`
	`2:['mil ciento diez','pesos'],`
	`3:['doce mil novecientos','pesos'] }`
			2

3	`{ 256:['docientos cincuenta y 6','cm'],`
	`45:['cuarenta y cinco','kg'],`
	`1110:['mil ciento diez','pesos'],`
	`12900:['doce mil novecientos','pesos'] }`
		X	**3**

4	`{ '256':'docientos cincuenta y 6 cm',`
	`'45':'cuarenta y cinco kg',`
	`'1110':'mil ciento diez pesos',`
	`'12900':'doce mil novecientos pesos' }`
			4

Ej 08 T6 – 2 Ptos

¿Qué muestra el siguiente programa?

```
def sabores(m):
  categorias={1:'agua',2:'agua',3:'crema',
              4:'crema',5:'agua',6:'chocolate',
              7:'chocolate',8:'crema',
              9:'crema',10:'crema',11:'sin tacc',
              12:'sin tacc'}
  return 'sin tacc' in categorias[m]

gustosHela={1:'Limón',2:'Frutilla',3:'Vainilla',
       4:'Pistacho',5:'Frambuesa',6:'Suizo',
       7:'Italiano',8:'Americana',9:'Amarena',
       10:'Del Cielo',11:'Dce de Leche',12:'Melón'}
elegidos=[4,6,9,12,11]
comprar=list(filter(sabores,elegidos))
for m in comprar:
  print(gustosHela[m])
```

1	6 9		**1**
2	Melón Dce de Leche	X	**2**
3	Pistacho Amarena		**3**
4	CHOCOLATE		**4**

Ej 09 T6 – 2 Ptos

Para el siguiente DataFrame mascotas:

```
   Nombre  Clase  Edad  Dueño
0  Pipo    loro    7    Juan
1  Tato    gato    5    Jimena
2  Kempes  gato   10    Lara
3  Tasso   perro   2    Hernán
```

¿Qué salida produce la siguiente instrucción?

```
mascotas.loc[1:2,['Dueño']]
```

1	Dueño 1 Jimena 2 Lara	X	**1**
2	Dueño Nombre 0 Juan Pipo 1 Jimena Tato 2 Lara Kempes 3 Hernán Tasso		**2**
3	Nombre Clase Edad Dueño 2 Kempes gato 10 Lara 3 Tasso perro 2 Hernán		**3**
4	Edad Clase gato 7.5 loro 7.0 perro 2.0 dtype: float64		**4**