Comenzado el	martes, 15 de mayo de 2018, 22:20
Estado	Finalizado
Finalizado en	viernes, 25 de mayo de 2018, 11:57
Tiempo empleado	9 días 13 horas
Puntos	17/17
Calificación	**10** de 10 (**100**%)

Pregunta 1

Correcta

Puntúa 1 sobre 1

Cuál de las siguientes cabeceras de ciclo *for* en Python, producirá un esquema de exactamente *n* repeticiones, suponiendo que la variable *n* tiene un valor entero mayor a cero previamente asignado?

Seleccione una:

a.
```
for i in range(0, n//2):
```

b.
```
for i in range(n+1):
```

c.
```
for i in range(n, 0, -1):
```
✓

¡Correcto! Las *n* repeticiones se hacen mediante una cuenta regresiva desde *n* hasta 1...

d.
```
for i in range(1, n):
```

¡Correcto!

La respuesta correcta es:
```
for i in range(n, 0, -1):
```

Pregunta 2

Correcta

Puntúa 1 sobre 1

¿Cuál de los siguientes scripts ***NO*** creará una tupla conteniendo exclusivamente los caracteres de la palabra 'Python', en ese mismo orden'? (Repetimos: la pregunta es cuál de todos ***NO*** creará la tupla pedida...)

Seleccione una:

a.
```
t3 = ('P', 'y', 't', 'h', 'o', 'n')
```

b.
```
Python = 'Java'
t5 = tuple(Python)
```
✓

¡Ok!

c.
```
t1 = tuple('Python')
```

d.
```
t4 = ()
for c in 'Python':
    t4 += c,
```

e.
```
t2 = 'P', 'y', 't', 'h', 'o', 'n'
```

¡Correcto!

La respuesta correcta es:

```
Python = 'Java'
t5 = tuple(Python)
```

Pregunta 3

Correcta

Puntúa 1 sobre 1

¿Cuáles de los siguientes ciclos for mostrarán en pantalla ***solamente*** números negativos? (Aclaración: más de una respuesta puede ser correcta... marque TODAS las que considere válidas...)

Seleccione una o más de una:

a.
```
for i in range(10, 1, -2):
    print('i:', i)
```

b.
```
for i in range(5, -15, -5):
    print('i:', i)
```

c.
```
for i in range(-10, -1):
    print('i:', i)
```
✓ ¡Ok!

d.
```
for i in range(-1, -10, -1):
    print('i:', i)
```
✓ ¡Ok!

¡Correcto!

Las respuestas correctas son:
```
for i in range(-1, -10, -1):
    print('i:', i),for i in range(-10, -1):
    print('i:', i)
```

Pregunta 4

Correcta

Puntúa 2 sobre 2

El proceso de búsqueda del mayor en el siguiente programa, contiene una ligera modificación a las variantes analizadas en la Ficha 7: la variable *may* se inicializa en 0 antes del ciclo, y luego simplemente se aplica la idea de comparar el número cargado *num* contra *may* en cada vuelta, sin preocuparse por el valor de *may* frente al primer dato. ¿Hay algún problema en el planteo de esta estrategia?

```
__author__ = 'Catedra de AED'

print('Determinacion del mayor de una sucesion (variante:
 may inicializada en 0)...')
n = int(input('Ingrese n: '))

may = 0
for i in range(1, n+1):
    num = int(input('Numero: '))
    if num > may:
        may = num

print('El mayor es:', may)
```

Seleccione una:

a.
No. No hay ningún problema. Funcionará correctamente en todos los casos.

b.
Sí, hay un problema: si todos los números del conjunto de entrada fuesen cero, se mostraría un *None* como resultado, en lugar de lo que correspondería retornar que es un 0.

c.
Sí, hay un problema: si todos los números del conjunto de entrada fuesen positivos o cero, se mostraría un 0 como resultado, en lugar del mayor del conjunto cargado.

d.
Sí, hay un problema: si todos los números del conjunto de entrada fuesen negativos, se mostraría un 0 como resultado, en lugar del mayor de los negativos cargados. ✓

¡Correcto!

¡Correcto!

La respuesta correcta es:

Sí, hay un problema: si todos los números del conjunto de entrada fuesen negativos, se mostraría un 0 como resultado, en lugar del mayor de los negativos cargados.

Pregunta 5

Correcta

Puntúa 3 sobre 3

El proceso de búsqueda del mayor en el siguiente programa está ligeramente cambiado con relación a su versión original (la tercera variante del problema de la búsqueda del mayor (Ficha 7): En la versión original, se comenzaba definiendo la variable *may* con el valor *None*, y ahora en esta versión que proponemos, hemos eliminado esa inicialización marcándola como un comentario (y sin borrarla, pero sólo por razones de claridad). ¿Hay algún problema en el planteo de esta pequeña variante? (Recuerde: si la instrucción está marcada como comentario, es lo mismo que si no estuviese).

```
__author__ = 'Catedra de AED'


print('Determinacion del mayor de una sucesion (variante 3)...')

# may = None
b = False
num = int(input('Ingrese un numero (con 0 finaliza): '))
while num != 0:
    if b == False:
        may = num
        b = True
    elif num > may:
        may = num
    num = int(input('Ingrese otro (con 0 finaliza): '))

print('El mayor es:', may)
```

Seleccione una:

a.

No. No hay ningún problema. Funcionará correctamente en todos los casos, y la instrucción *may* = *None* era completamente innecesaria en la versión original.

b.

Sí, hay un problema: si en la primera carga antes del ciclo se ingresa un 0, el ciclo no ejecutará su bloque de acciones y la variable *may* quedará sin definir, provocando en ese caso que el programa se interrumpa y lance un error al intentar mostrar el valor de *may* en el programa. ✓

¡Correcto!

c.

Sí, hay un problema: si todos los números que cargan fuesen negativos, en ese caso la variable *may* por defecto quedaría valiendo 0, y el programa mostraría un cero al final.

d.

Sí, hay un problema: la variable *may* estará sin definir incluso dentro del ciclo. Cuando se intente asignar en ella el valor de *num* que corresponda a cada vuelta, el programa se interrumpirá y lanzará un error de variable no definida.

¡Correcto!

La respuesta correcta es:
Sí, hay un problema: si en la primera carga antes del ciclo se ingresa un 0, el ciclo no ejecutará su bloque de acciones y la variable *may* quedará sin definir, provocando en ese caso que el programa se interrumpa y lance un error al intentar mostrar el valor de *may* en el programa.

Pregunta 6

Correcta

Puntúa 3 sobre 3

El proceso de búsqueda del mayor en el siguiente programa, contiene una modificación de la primera variante analizada en la Ficha 7: En ambas se usa un ciclo for que ejecute *n* repeticiones para cargar los *n* números. Pero en la versión original de la Ficha, el ciclo se ajusta como *for i in range(1, n+1)* mientras que ahora proponemos *for i in range(n)* para intentar abreviar, basándonos en la idea de que en ambos casos el ciclo hará efectivamente *n* repeticiones. ¿Hay algún problema con este cambio propuesto?

```
__author__ = 'Catedra de AED'


print('Determinacion del mayor de una sucesion (for cambiad
o)...')

n = int(input('Numero: '))
for i in range(n):
    num = int(input('Numero: '))
    if i == 1:
        may = num
    elif num > may:
        may = num

print('El mayor es:', may)
```

Seleccione una:

a.
Sí, hay un problema: el rango generado con *range(n)* no contiene efectivamente *n* números, sino *n-1* números y el ciclo hará entonces una repetición menos de las esperadas.

b.
Sí, hay un problema: el rango generado con *range(n)* contiene efectivamente *n* números y el ciclo hará *n* repeticiones, pero el primer valor del rango será 0 (y no 1) con lo cual al preguntar si *i == 1* en la primera vuelta del ciclo se obtendrá un falso. La variable *may* quedará entonces sin definir, y en esa misma vuelta el programa se interrumpirá al comparar *num* con *may*. ✓

¡Correcto!

c.
No. No hay ningún problema. El funcionamiento es exactamente igual al de la versión original.

d.
Sí, hay un problema: el rango generado con *range(n)* contiene efectivamente *n* números y el ciclo hará *n* repeticiones, pero el primer valor del rango será 0 (y no 1). Por lo tanto, la condición i == 1 sólo verdadera en la segunda vuelta del ciclo y la variable may quedará

inicializada con el segundo número en lugar del primero. El programa entonces, ignorará el primer número que se cargue, y mostrará el mayor de la secuencia pero sin incluir al primero.

¡Correcto!

La respuesta correcta es:
Sí, hay un problema: el rango generado con *range(n)* contiene efectivamente *n* números y el ciclo hará *n* repeticiones, pero el primer valor del rango será 0 (y no 1) con lo cual al preguntar si *i == 1* en la primera vuelta del ciclo se obtendrá un falso. La variable *may* quedará entonces sin definir, y en esa misma vuelta el programa se interrumpirá al comparar *num* con *may*.

Pregunta 7

Correcta

Puntúa 3 sobre 3

Supongamos que se nos pide buscar el mayor de una secuencia de n números, tal como se analizó en la Ficha 7, pero de modo que ahora además de mostrar el mayor, se muestre también en qué orden apareció en la carga (o sea, en qué vuelta del ciclo apareció ese mayor) Por ejemplo, si la secuencia a cargar fuese {2, 5, 1, 3} entonces el mayor sería el 5 y apareció en el lugar 2 (o en el orden 2, o en la vuelta 2 del ciclo) ¿Está bien planteado el siguiente programa para cumplir con este nuevo requerimiento?

```
__author__ = 'Catedra de AED'


print('Determinacion del mayor de una sucesion (variante 1)...')

n = int(input('n: '))
for v in range(1, n+1):
    num = int(input('Numero: '))
    if v == 1:
        may, pos = num, 1
    elif num > may:
        may, pos = num, v

print('El mayor es:', may)
print('Se cargó en la vuelta:', pos)
```

Seleccione una:

a.
Sí, está correctamente planteado. ✓

¡Correcto!

b.
No. No está bien planteado: si el valor mayor apareciese repetido más de una vez en la carga, la variable *pos* quedaría valiendo *None*.

c.
No. No está bien planteado: siempre muestra que la vuelta en que se cargó el mayor fue la 1 del ciclo.

d.
No. No está bien planteado: los resultados se están mostrando al revés, ya que el mayor estaría en *pos* y su posición de carga estaría en *may*. Hay que invertir las asignaciones de las tuplas dentro del ciclo.

¡Correcto!

La respuesta correcta es:
Sí, está correctamente planteado.

Pregunta 8

Correcta

Puntúa 1 sobre 1

Analice el siguiente script en el cual se recorre una secuencia llamada sec con un ciclo *for*:

```
# suponga que sec es una secuencia (cadena, tupla, rango, etc.)
# ya inicializada
for x in sec:
    print('x:', x)
```

¿Qué hace este script si la secuencia *sec* estuviese inicialmente vacía?

Seleccione una:

a.
El ciclo *for* no hace ninguna repetición y el script termina normalmente sin mostrar nada en pantalla. ✓

¡Ok! Efectivamente... el ciclo for es de tipo [0, N] por lo que si la secuencia a recorrer está vacía, el for no llega a hacer ninguna repetición y sigue de largo a buscar la siguiente instrucción.

b.
El ciclo *for* hace una y sólo una repetición y muestra el mensaje *x: None*.

c.
El ciclo *for* entra en repetición infinita.

d.
El ciclo *for* no hace ninguna repetición pero el script se interrumpe lanzando un mensaje de error.

¡Correcto!

La respuesta correcta es:
El ciclo *for* no hace ninguna repetición y el script termina normalmente sin mostrar nada en pantalla.

Pregunta 9

Correcta

Puntúa 1 sobre 1

Cuáles de las siguientes afirmaciones son ciertas en cuanto al uso de ciclos o instrucciones repetitivas en Python? (Aclaración: más de una respuesta puede ser correcta... marque TODAS las que considere válidas...)

Seleccione una o más de una:

a.
Todos los ciclos básicos de Python, son de la forma [1-N].

b.
Python provee dos tipos de ciclos básicos: el *while* y el *for*. ✓

¡Ok!

c.
El ciclo *while* de Python ***sólo puede aplicarse*** cuando se desconoce la cantidad de repeticiones a realizar. Si la cantidad de repeticiones se conoce de antemano, ***debe*** aplicarse el *for*.

d.
Todos los ciclos básicos de Python, son de la forma [0, N]. ✓

¡Ok!

¡Correcto!

Las respuestas correctas son:
Python provee dos tipos de ciclos básicos: el *while* y el *for*.,

Todos los ciclos básicos de Python, son de la forma [0, N].

Pregunta 10

Correcta

Puntúa 1 sobre 1

Suponga que se quiere cargar por teclado el valor una temperatura, validando que la misma esté entre -70 grados y 50 grados, y se plantea un ciclo *while forzado a operar en forma [1, N],* como muestra el esquema, para hacerlo:

```
__author__ = 'Catedra de AED'

t = 51
while ?????? :
    t = int(input('Cargue temperatura (entre -70 y 50, por favo
r): '))
```

¿Cuál de las siguientes debería ser la condición a incluir en ese ciclo *while* (en el lugar donde figuran los signos de pregunta de color rojo) para que el *valor vuelva a cargarse* en caso de haber sido mal ingresado?

Seleccione una:

a.
`t >= -70 or t <= 50`

b.
`t < -70 and t > 50`

c.
`t >= -70 and t <= 50`

d.
`t < -70 or t > 50` ✓

¡Correcto!

¡Correcto!

La respuesta correcta es:
`t < -70 or t > 50`