

SGE (Python)	1ªEvaluación (70%)	
Nombre		21-Noviembre-2019

Nota: Se valora principalmente el algoritmo, la utilización de funciones y “herramientas Python”, y después la ejecución correcta.

Añade comentarios de lo que se esta haciendo.

1. (1) Realizar utilizando formato “**normal**” y “**comprehension**”.

a) (0,5) Partiendo que tenemos dos listas, una que contiene números impares y otra que contiene números divisibles entre 3.

- Generar las dos listas anteriores por “comprehension”
- Generar una nueva lista con aquellos números que sean impares y divisibles por 3.

```
[1, 3, 5, 7, 9, 11, 13, 15, 17, 19]
[0, 3, 6, 9, 12, 15, 18]
[3, 9, 15]
```

b) (0,5) Tenemos una lista con los días de la semana.

- Generar un **diccionario** que tenga como índice el número de la semana empezando por 1 :lunes. y además asegurando que el nombre empiece por mayúsculas:

```
['lunes', 'martes', 'miercoles', 'jueves', 'viernes', 'sabado', 'domingo']
{1: 'Lunes', 2: 'Martes', 3: 'Miercoles', 4: 'Jueves', 5: 'Viernes', 6: 'Sabado', 7: 'Domingo'}
```

2. (1) Realiza utilizando **una única línea de código**:

a) (0,25) Visualizar si la variable **letra** esta o no contenida en una cadena dada..

Comprobar si se encuentra en la cadena “Examen Python”, visualizando si está o no esta..

```
La r No esta
La a Esta
```

Nota: La comprobación y visualización en una misma línea de código.

b) (0,25) Crear un conjunto vacío, y visualizar su tipo y longitud.

```
Tipo: <class 'set'> Longitud: 0
```

c) (0,25) Asigna a las variables **A**, **B** y **C**, los siguientes valores: **(3,4,5)** ; **100** ; “**HOLA**” y **{2,4}**, sin que de error y en una única línea de código.

```
(3, 4, 5)
100
['HOLA', {2, 4}]
```

d) (0,25) Genera un ejemplo en la que tenemos para asignar, los siguientes valores: **(3,4),”hola”, 5, 9, [“A”,”b”.”c”]** “**adios**”, **23** y las variables **A**, **B** y **C**, y queremos asignar a **A** “hola” y a **C**=”adios” y para **B** todos los valores que haya en medio.

3. (1) Dado el siguiente código, que da error:

```
tupla = tuple(input ('teclea los numeros, separador por comas '))
print ("la tupla es", tupla)
lista = []
for elem in tupla:
    lista.append(int(elem))
print ("la lista es :",lista)
```

a) (0,5) Modifica el código para que tome varios numeros por teclado y los visualice en formato de tupla y de lista.

```
teclea los numeros, separador por comas 3,4,5,3,5
la tupla es (3, 4, 5, 3, 5)
la lista es : [3, 4, 5, 3, 5]
```

b) (0,5) Modifica el código y intentalo con comprehension o utilizando iteradores.

4. (2) Escribe un modulo Python, que :

a) (0,5) Contenga el menu siguiente, y además valide que se tecleen las opciones indicadas (se admite que tecleen la opción tanto en mayúsculas como minúsculas), dando el mensaje de error correspondiente.

```
Ejercicio cuatro
*****
A) Comprobar palabras reservadas
B) Comprobar funciones integradas
C) Comprobar nombres del ambito
S) Salir
Selecciona opcion: a|
```

- Si se selecciona :

b) (0,5) A). Se solicita una palabra reservada, se comprueba que existe como tal y a continuación se visualiza su sintaxis.

Nota: con **kwlist** del modulo **keyword**, se obtienen todas las palabras reservadas.

c) (0,5) B) Se solicita igualmente una función integrada, se comprueba que existe y, y en ese caso si se se puede invocar, y si es un iterador.

d) (0,5) C) Se solicita cualquier nombre y se comprueba que exista en el entorno de ese modulo, y en ese se visualiza el tipo de objeto que es.

Nota: En el caso de que no existan, dar el mensaje correspondiente.

5. (1) Crear un modulo que debe contener:

Una función generadora y dos variables una de tipo lista y otra de tipo tupla.

La función genera **multiplos** de un número, que **empieza** a partir de un numero dado, el numero de **elementos** que genera puede ser infinito o finito , dependiendo de que se envio o no valor al parametro.

Por defecto **multiplo** es 2, **empieza** es 1 y **elementos** sin valor.

6. (1) Crea otro modulo que, utiliza la función y variables del modulo anterior, y que mientras no teclees * (asterisco), te pide valores de **multiplo, **empieza** y **elementos** (alguno puede quedar sin valor) los guarda en una tupla y llama a la función generadora utilizando la tupla y guardando los valores recibidos en la lista.**

7. (1) A partir de una **tupla_dada (30,33,54,34,29,49) , consigue una lista, teniendo en cuenta que para cada elemento del dato anterior, hay que realizar las siguientes operaciones:**

a) Si es un número **par**: multiplicar por el numero 5, y restar por otro numero 5.

b) Si es **impar**. Sumar un numero 10 y dividir por otro numero 3.