Estructuras Discretas 2022-1
Práctica 1: Introducción a python

Facultad de Ciencias, UNAM

Fecha de entrega: 22 de octubre del 2021

Resuelva los siguientes ejercicios de manera **individual**. En el archivo `practica1.py` se encuentra la firma correspondiente para cada función donde tiene que completar con código. Después de haber resuelto todo, puede ejecutar dentro de la carpeta **ED-P1**, el comando `python3 -m unittest -v test_practica1.py` para asegurarse que la salida de cada función es correcta. Es importante mencionar que esto último no asegura que esté bien implementado, sin embargo podrá darse cuenta si su implementación es incorrecta.

Para la entrega, siga las especificaciones publicadas en `Classroom`.

1. Defina la función `maximo_comun_divisor` en la que reciba dos números enteros positivos y devuelva el máximo común divisor.

   Al ejecutarla con los siguientes parámetros se deben obtener los mismos resultados.

   ```
   >>> maximo_comun_divisor(950,480)
   10
   >>> maximo_comun_divisor(336,360)
   24
   ```

2. Complete la función `es_palindroma` que tiene como parámetro de entrada una cadena y **verifica** si es palíndroma. No se puede usar `reverse`

   Ejemplo:

   ```
   >>> es_palindroma("anita lava la tina")
   True
   >>> es_palindroma("yo dono rosas oro no doy")
   True
   >>> es_palindroma("hola mundo")
   False
   ```

3. Defina la función `promedio_tuplas`. Esta recibe una tupla de tuplas y retorna una lista con el promedio de los números contenidos en cada una de las tuplas.

   Ejemplo:

   ```
   >>> promedio_tuplas((10,3,4,5),(4, 17, 8, 9),(10,10,10))
   [5.5, 9.5, 10.0]
   ```

4. Defina la función `permutaciones`. Recibe una lista de números y retorna una lista con todas las posibles permutaciones. No puede usar métodos de la biblioteca `itertools`.

   Ejemplo:

   ```
   >>> permutaciones([1,2,3])
   [[3, 2, 1], [2, 3, 1], [2, 1, 3], [3, 1, 2], [1, 3, 2], [1, 2, 3]]
   ```

5. Escriba la función `filtrar_pares` que reciba una lista de números enteros positivos y filtre los números pares. Para esto debe usar `lambda`.

   ```
   >>> filtrar_pares([1,2,3,4,5,6,8])
   [2,4,6,8]
   ```

6. Defina la función `frecuencia` que recibe una cadena y devuelve una lista de tuplas. El primer elemento en la tupla es la palabra y el segundo elemento es la cantidad de ocurrencias en la cadena.

   Ejemplo:

```
>>> cadena = "El grupo de este semestre es el mejor grupo de la fac"
>>> frecuencia(cadena)
[('de', 2), ('el', 1), ('grupo', 2), ('es', 1), ('fac', 1), ('la', 1),
('mejor', 1), ('El', 1), ('semestre', 1), ('este', 1)]
```

   Note que las las palabras si una palabra aparece más de una vez, solo se considera una sola tupla $(palabra, frecuencia)$, es decir, no hay repetición. El orden en el que se encuentren no importará siempre y cuando estén todas las tuplas con palabras distintas.

7. Defina la función `suma_digitos` que recibe un número entero positivo y suma cada uno de sus dígitos hasta que sea menor a 10.

   Ejemplo:

```
>>>suma-digitos(12345)
"""
la primer suma es 1 + 2 + 3 + 4 + 5 = 15
luego la segunda suma es 1 + 5 = 6
como esta es menor a 10, entonces este es el resultado
"""
>>> 6
```

8. Defina la función `reemplaza_espacios` donde recibe una cadena de entrada y todos los espacios que hay en la cadena los reemplaza por + y los coloca al principio.

   Ejemplo:

```
>>>reemplaza-espacios("Mi nombre es sora y me gusta el pollito")
>>>"++++++++Minombreessoraymegustaelpollito"
>>>reemplaza-espacios("correo    gmail.com")
>>>"+++correogmail.com"
```

9. Defina la función `dibuja_triangulo` que devuelve la cadena siguiente donde el último elemento es un triángulo conformado por asteriscos con base n

```
>>>dibuja-triangulo(5)
    *

    *
   * *

    *
   * *
  * * *

    *
   * *
  * * *
 * * * *

    *
   * *
  * * *
 * * * *
* * * * *
```

10. Defina la función `pascal` que retorne la `cadena` de las primeras n filas del triángulo de Pascal. Ejemplo:

```
>>>pascal(6)
[1]
[1  1]
[1  2  1]
[1  3  3  1]
```

```
[1  4  6  4  1]
[1  5  10  10  5  1]
```