Desarrolle las siguientes funciones en python:

1. Función que reciba un diccionario y agregar una clave-valor, retornar el diccionario modificado (Debe agregarlo al final)

```python
# 1- Función para agregar una clave-valor a un diccionario.
print(1)
def agregar_clave_valor(diccionario, clave, valor):
    diccionario[clave] = valor
    return diccionario

mi_diccionario = {"nombre": "Juan", "edad": 25}
print(mi_diccionario)
nuevo_diccionario = agregar_clave_valor(mi_diccionario, "ciudad", "Madrid")
print(nuevo_diccionario)
```

```
1
{'nombre': 'Juan', 'edad': 25}
{'nombre': 'Juan', 'edad': 25, 'ciudad': 'Madrid'}
```

2. Función que reciba un diccionario y elimine una clave-valor, retornar el diccionario modificado

```python
# 2- Función para eliminar una clave-valor de un diccionario.
print(2)
def eliminar_clave_valor(diccionario, clave):
    if clave in diccionario:
        del diccionario[clave]
    return diccionario

mi_diccionario = {"nombre": "Juan", "edad": 25, "ciudad": "Madrid"}
print(mi_diccionario)
diccionario_modificado = eliminar_clave_valor(mi_diccionario, "edad")
print(diccionario_modificado)
```

```
2
{'nombre': 'Juan', 'edad': 25, 'ciudad': 'Madrid'}
{'nombre': 'Juan', 'ciudad': 'Madrid'}
```

3. Función que reciba un diccionario y modifique el valor de una clave, retornar verdadero si pudo modificar y falso si no pudo

```python
# 3- Función para modificar el valor de una clave en un diccionario.
print(3)
def modificar_valor(diccionario, clave, nuevo_valor):
    if clave in diccionario:
        diccionario[clave] = nuevo_valor
        return True
    else:
        return False

mi_diccionario = {"nombre": "Juan", "edad": 25}
resultado = modificar_valor(mi_diccionario, "edad", 30)
print("Se pudo modificar?", resultado)
print("Diccionario modificado:", mi_diccionario)

resultado = modificar_valor(mi_diccionario, "ciudad", "Madrid")
print("Se pudo modificar?", resultado)
print("Diccionario después del intento:", mi_diccionario)
```

```
3
Se pudo modificar? False
Diccionario modificado: {'nombre': 'Juan', 'ciudad': 'Madrid'}
Se pudo modificar? True
Diccionario despues del intento: {'nombre': 'Juan', 'ciudad': 'Madrid'}
```

4. Función que combine dos diccionarios, regresar el diccionario resultante

```python
# 4- Función para combinar dos diccionarios.
print(4)
def combinar_diccionarios(dic1, dic2):
    resultado = dic1.copy()
    resultado.update(dic2)
    return resultado

diccionario1 = {"nombre": "Juan", "edad": 25}
diccionario2 = {"ciudad": "Madrid", "edad": 30}
diccionario_combinado = combinar_diccionarios(diccionario1, diccionario2)
print("Diccionario combinado:", diccionario_combinado)
```

```
4
Diccionario combinado: {'nombre': 'Juan', 'edad': 30, 'ciudad': 'Madrid'}
```

5. Función que agregue elementos a un conjunto, retornar verdadero si pudo agregar y falso si no pudo.

```python
# 5- Función para agregar un elemento a un conjunto.
print(5)
def agregar_a_conjunto(conjunto, elemento):
    if elemento not in conjunto:
        conjunto.add(elemento)
        return True
    else:
        return False

mi_conjunto = {1, 2, 3, 4}
resultado = agregar_a_conjunto(mi_conjunto, 5)
print("Se pudo agregar el elemento 5?", resultado)
print("Conjunto actualizado:", mi_conjunto)

resultado = agregar_a_conjunto(mi_conjunto, 3)
print("Se pudo agregar el elemento 3?", resultado)
print("Conjunto después del intento:", mi_conjunto)
```

```
5
Se pudo agregar el elemento 5? True
Conjunto actualizado: {1, 2, 3, 4, 5}
Se pudo agregar el elemento 3? False
Conjunto despues del intento: {1, 2, 3, 4, 5}
```

6. Función que elimine un elemento de un conjunto, retornar verdadero si pudo eliminar y falso si no pudo

```python
# 6- Función para eliminar un elemento de un conjunto.
print(6)
def eliminar_de_conjunto(conjunto, elemento):
    if elemento in conjunto:
        conjunto.remove(elemento)
        return True
    else:
        return False

mi_conjunto = {1, 2, 3, 4, 5}
resultado = eliminar_de_conjunto(mi_conjunto, 3)
print("Se pudo eliminar el elemento 3?", resultado)
print("Conjunto actualizado:", mi_conjunto)

resultado = eliminar_de_conjunto(mi_conjunto, 6)
print("Se pudo eliminar el elemento 6?", resultado)
print("Conjunto después del intento:", mi_conjunto)
```

```
6
Se pudo eliminar el elemento 3? True
Conjunto actualizado: {1, 2, 4, 5}
Se pudo eliminar el elemento 6? False
Conjunto despues del intento: {1, 2, 4, 5}
```

7. Función que combine dos conjuntos, regresar el conjunto resultante

```python
# 7- Función para combinar dos conjuntos.
print(7)
def combinar_conjuntos(conjunto1, conjunto2):
    resultado = conjunto1 | conjunto2
    return resultado

pares = {2, 4, 6, 8, 10}
impares = {1, 3, 5, 7, 9}
numeros = combinar_conjuntos(pares, impares)
print("Conjunto combinado:", numeros)
```

```
7
Conjunto combinado: {1, 2, 3, 4, 5, 6, 7, 8, 9, 10}
```

8. Función que regrese la diferencia de dos conjuntos

```python
# 8- Función para calcular la diferencia entre dos conjuntos.
print(8)
def diferencia_conjuntos(conjunto1, conjunto2):
    resultado = conjunto1 - conjunto2
    return resultado

numeros1 = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10}
pares = {2, 4, 6, 8, 10}
diferencia = diferencia_conjuntos(numeros1, pares)
print("Conjunto original:", numeros1)
print("Diferencia entre conjuntos:", diferencia)
```

```
8
Conjunto originar:  {1, 2, 3, 4, 5, 6, 7, 8, 9, 10}
Diferencia entre conjuntos: {1, 3, 5, 7, 9}
```

9. Función que agregue un elemento a una tupla y lo guarde los cambios en una tupla nueva, regresar la nueva tupla

```python
# 9- Función para agregar un elemento a una tupla.
print(9)
def agregar_a_tupla(tupla_original, elemento):
    nueva_tupla = tupla_original + (elemento,)
    return nueva_tupla

mi_tupla = (1, 2, 3, 4)
nueva_tupla = agregar_a_tupla(mi_tupla, 5)
print("Tupla original:", mi_tupla)
print("Nueva tupla:", nueva_tupla)
```

```
9
Tupla original: (1, 2, 3, 4)
Nueva tupla: (1, 2, 3, 4, 5)
```

10. Función que elimine un elemento a una tupla y lo guarde los cambios en una tupla nueva, regresar la nueva tupla

```python
# 10- Función para eliminar un elemento de una tupla.
print(10)
def eliminar_de_tupla(tupla_original, elemento):
    if elemento in tupla_original:
        nueva_tupla = tuple(item for item in tupla_original if item != elemento)
        return nueva_tupla
    else:
        return tupla_original

mi_tupla = (10, 20, 30, 40, 50)
nueva_tupla = eliminar_de_tupla(mi_tupla, 30)
print("Tupla original:", mi_tupla)
print("Nueva tupla:", nueva_tupla)
```

```
10
Tupla original: (10, 20, 30, 40, 50)
Nueva tupla: (10, 20, 40, 50)
```

11. Función que concatene dos tuplas en una nueva, regresar la nueva tupla

```python
# 11- Función para concatenar dos tuplas.
print(11)
def concatenar_tuplas(tupla1, tupla2):
    nueva_tupla = tupla1 + tupla2
    return nueva_tupla

impares = (1, 3, 5, 7, 9)
pares = (2, 4, 6, 8, 10)
numeros = concatenar_tuplas(impares, pares)
print("Tupla 1:", impares)
print("Tupla 2:", pares)
print("Tupla concatenada:", numeros)
```

```
11
Tupla 1: (1, 3, 5, 7, 9)
Tupla 2: (2, 4, 6, 8, 10)
Tupla concatenada: (1, 3, 5, 7, 9, 2, 4, 6, 8, 10)
```

12. Función que revertir el orden de una tupla y lo guarde los cambios en una tupla nueva, regresar la nueva tupla

```python
# 12- Función para revertir una tupla.
print(12)
def revertir_tupla(tupla_original):
    nueva_tupla = tupla_original[::-1]
    return nueva_tupla

mi_tupla = (1, 2, 3, 4, 5)
tupla_revertida = revertir_tupla(mi_tupla)
print("Tupla original:", mi_tupla)
print("Tupla revertida:", tupla_revertida)
```

```
12
Tupla original: (1, 2, 3, 4, 5)
Tupla revertida: (5, 4, 3, 2, 1)
```

13. Función que recibe un diccionario y lo imprime

```python
# 13- Función para imprimir un diccionario.
print(13)
def imprimir_diccionario(diccionario):
    for clave, valor in diccionario.items():
        print(f"{clave}: {valor}")

mi_diccionario = {
    "nombre": "Juan",
    "edad": 25,
    "ciudad": "Mexicali",
    "profesion": "Ingeniero"
}
imprimir_diccionario(mi_diccionario)
```

```
13
nombre: Juan
edad: 25
ciudad: Mexicali
profesion: Ingeniero
```

14. Función que recibe un tupla y la imprime

```python
# 14- Función para imprimir una tupla.
print(14)
def imprimir_tupla(tupla):
    print(tupla)

mi_tupla = (10, 20, 30, 40, 50)
imprimir_tupla(mi_tupla)
```

```
14
(10, 20, 30, 40, 50)
```

15. Función que recibe un conjunto y lo imprime

```python
# 15- Función para imprimir un conjunto.
print(15)
def imprimir_conjunto(conjunto):
    print(conjunto)

mi_conjunto = {10, 20, 30, 40, 50}
imprimir_conjunto(mi_conjunto)
```

```
15
{50, 20, 40, 10, 30}
```