# 📚 CHEATSHEET - DICCIONARIOS Y CADENAS EN PYTHON

## 🔤 OPERACIONES CON CADENAS

### Métodos básicos

python

```python
texto = "Hola Mundo"
texto.upper()        # "HOLA MUNDO"
texto.lower()        # "hola mundo"
texto.strip()        # Elimina espacios al inicio/final
texto.replace(" ", "") # "HolaMundo"
texto.split()        # ["Hola", "Mundo"]
" ".join(["A", "B"])  # "A B"
```

### Búsqueda y verificación

python

```python
"Hola" in texto      # True - verifica si contiene
texto.find("Mundo")   # 5 - posición donde empieza (-1 si no encuentra)
texto.startswith("H") # True
texto.endswith("o")   # True
texto.count("o")      # 2 - cuenta ocurrencias
```

### Verificación de caracteres

python

```python
"123".isdigit()      # True - solo dígitos
"ABC".isupper()      # True - todo mayúsculas
"abc".islower()      # True - todo minúsculas
"Abc".isalpha()      # True - solo letras
```

### Iteración sobre cadenas

python

```python
# Cada carácter
for char in "ABC":
    print(char)      # A, B, C

# Con índice
for i, char in enumerate("ABC"):
    print(i, char)   # 0 A, 1 B, 2 C
```

# 📖 OPERACIONES CON DICCIONARIOS

## Creación y acceso

python

```python
# Crear diccionario
dic = {}                    # Vacío
dic = {"a": 1, "b": 2}      # Con valores
dic = dict(a=1, b=2)        # Constructor

# Acceso seguro
dic["a"]                    # 1 (KeyError si no existe)
dic.get("c")                # None
dic.get("c", 0)             # 0 (valor por defecto)
```

## Añadir/Modificar elementos

python

```python
dic["c"] = 3                # Añadir o modificar
dic.update({"d": 4, "e": 5})  # Añadir múltiples
dic.setdefault("f", 6)      # Solo añade si no existe
```

## Eliminar elementos

python

```python
del dic["a"]                # Elimina clave
valor = dic.pop("b")        # Elimina y devuelve valor
dic.clear()                 # Vacía el diccionario
```

## Iteración

python

```python
# Solo claves
for clave in dic:
    print(clave)

# Claves y valores
for clave, valor in dic.items():
    print(clave, valor)

# Solo valores
for valor in dic.values():
    print(valor)

# Solo claves (explícito)
for clave in dic.keys():
    print(clave)
```

### Operaciones útiles

python

```python
len(dic)                # Número de elementos
"clave" in dic          # Verificar si existe clave
dic.copy()              # Copia superficial
list(dic.keys())        # Lista de claves
list(dic.values())      # Lista de valores
```

## 🎯 PATRONES COMUNES PARA LOS EJERCICIOS

### 1. Contar ocurrencias (Ejercicio 1)

python

```python
contador = {}
for elemento in secuencia:
    contador[elemento] = contador.get(elemento, 0) + 1
```

### 2. Comparar diccionarios (Ejercicio 2)

python

```python
# Dos diccionarios son iguales si tienen las mismas claves y valores
dic1 == dic2
```

### 3. Traducir/Mapear valores (Ejercicio 3)

python

```python
# Usar get() con valor por defecto
traducido = diccionario.get(palabra, palabra)  # Si no existe, devuelve la misma
```

## 4. Modificar valores in-place (Ejercicio 4)

python

```python
for clave in diccionario:
    diccionario[clave] = nueva_funcion(diccionario[clave])
```

## 5. Crear diccionario anidado (Ejercicio 5, 9)

python

```python
resultado = {}
for clave, valor in dic_original.items():
    resultado[clave] = {
        "campo1": calcular1(valor),
        "campo2": calcular2(valor)
    }
```

## 6. Fusionar diccionarios (Ejercicio 6)

python

```python
# Copiar para no modificar original
resultado = dic1.copy()
resultado.update(dic2)  # Simple, pero sobrescribe

# Con control de conflictos
for k, v in dic2.items():
    if k in resultado:
        # Manejar conflicto
    else:
        resultado[k] = v
```

## 7. Parsear strings estructurados (Ejercicio 7)

python

```python
# split() con maxsplit limita divisiones
partes = linea.split(maxsplit=3)  # Máximo 4 partes
fecha, hora, nivel, mensaje = partes
```

## 8. Validación con any/all (Ejercicio 8)

python

```python
# Verificar si algún carácter cumple condición
any(c.isupper() for c in texto)   # ¿Hay alguna mayúscula?
all(c.isdigit() for c in texto)   # ¿Todos son dígitos?
```

## 9. Acceso a diccionarios anidados (Ejercicio 9)

python

```python
# Navegación segura
valor = dic.get("nivel1", {}).get("nivel2", {}).get("campo", default)

# Acceso directo (puede dar KeyError)
valor = dic["nivel1"]["nivel2"]["campo"]
```

# 💡 TIPS Y TRUCOS

## Comprensión de diccionarios

python

```python
# Crear diccionario con comprensión
cuadrados = {x: x**2 for x in range(5)}
# {0: 0, 1: 1, 2: 4, 3: 9, 4: 16}

# Filtrar diccionario
filtrado = {k: v for k, v in dic.items() if v > 10}
```

## Ordenar diccionarios

python

```python
# Por claves
dict(sorted(dic.items()))

# Por valores
dict(sorted(dic.items(), key=lambda x: x[1]))
```

## Valores por defecto elegantes

python

```python
# En lugar de:
if key in dic:
    dic[key] += 1
else:
    dic[key] = 1

# Usar:
dic[key] = dic.get(key, 0) + 1

# O con defaultdict:
from collections import defaultdict
dic = defaultdict(int)
dic[key] += 1  # No necesita inicialización
```

### Combinar cadenas y diccionarios

python

```python
# Formateo con diccionario
datos = {"nombre": "Juan", "edad": 30}
texto = "Me llamo {nombre} y tengo {edad} años".format(**datos)
# O con f-strings:
texto = f"Me llamo {datos['nombre']} y tengo {datos['edad']} años"
```

## 🚨 ERRORES COMUNES Y SOLUCIONES

### 1. KeyError al acceder diccionarios

python

```python
# MAL
valor = dic["clave_que_no_existe"]  # KeyError!

# BIEN
valor = dic.get("clave_que_no_existe", valor_por_defecto)
# O verificar antes:
if "clave" in dic:
    valor = dic["clave"]
```

### 2. Modificar diccionario mientras se itera

python

```python
# MAL
for k in dic:
    if condicion:
        del dic[k]  # RuntimeError!

# BIEN
claves_a_borrar = [k for k in dic if condicion]
for k in claves_a_borrar:
    del dic[k]
```

### 3. Copias superficiales vs profundas

python

```python
# Copia superficial (cambios en objetos anidados afectan ambos)
dic2 = dic1.copy()

# Copia profunda (completamente independiente)
import copy
dic2 = copy.deepcopy(dic1)
```

### 4. list() vs []

python

```python
# MAL (para el ejercicio de enzimas)
return list(secuencia)  # ['A', 'T', 'C', 'G']

# BIEN
return [secuencia]      # ['ATCG']
```

## 📝 RESUMEN RÁPIDO POR EJERCICIO

1. **Contador**: `dic.get(key, 0) + 1`
2. **Anagramas**: Comparar diccionarios de conteo
3. **Traductor**: `dic.get(palabra, palabra)`
4. **Modificador**: Iterar y cambiar valores in-place
5. **Análisis ADN**: Calcular porcentajes, crear dic anidado
6. **Fusión**: `copy()` + lógica de conflictos
7. **Logs**: `split(maxsplit=n)` + diccionario de resultados
8. **Validación**: `any()`, `all()`, verificaciones múltiples

9. **Lab clínico**: Navegación en diccionarios anidados + lógica médica