UdeSantiago

CLASE N°16

Algunos objetos nativos de Python

ESTRUCTURAS CLÁSICAS

- Filas
  - **Secuencia de valores** cuyo comportamiento emula las filas que hacemos los seres humanos
    - Elementos se **añaden al final** de la fila
    - Elementos **salen del frente** de la fila
    - En Python:

```
fila = list()
fila.append(elemento)
elemento = fila.pop(0)
```

ESTRUCTURAS CLÁSICAS

- Pilas
  - **Secuencia de valores** cuyo comportamiento emula las pilas de platos
    - Elementos se **añaden en el tope** de la pila
    - Elementos **salen del tope** de la pila
    - En Python:

```
pila = list()
pila.append(elemento)
elemento = pila.pop()
```

ESTRUCTURAS CLÁSICAS

- Conjuntos
  - **Colección de valores** cuyo comportamiento responde a la teoría de conjuntos
    - **No contienen** elementos repetidos
    - Elementos son **inmutables**
    - En Python:

```
cjto = set()
cjto.add(elemento)
cjto.discard(elemento)
```

ESTRUCTURAS CLÁSICAS

- Conjuntos
  - La clase `set` provee métodos para las operaciones de la teoría de conjuntos
    - **Unión**: `cjto1.union(cjto2)`
    - **Intersección**: `cjto1.intersection(cjto2)`
    - **Diferencia asimétrica**: `cjto1.difference(cjto2)`
    - **Diferencia simétrica**: `cjto1.symmetric_difference(cjto2)`
  - Son objetos **iterables**

ESTRUCTURAS CLÁSICAS

- Diccionarios
  - **Colección de valores** que están **indexados** por **llaves**
    - **No contienen** llaves repetidas
    - Las llaves son **inmutables**
    - Permiten búsquedas **más eficientes** (no secuenciales)
    - En Python:

```
dicc = dict()
dicc[llave] = elemento
del dicc[llave]
```

ESTRUCTURAS CLÁSICAS

- Diccionarios
  - Podemos preguntar si **contiene una llave** dada

```
dicc.has_key(llave)
```

  - Podemos obtener una copia de la **lista de llaves**

```
listaLlaves = dicc.keys()
```

    - Podemos usarla para **iterar sobre los elementos**
    - "Copia" significa que cambios en la lista de llaves devuelta **no se reflejan** en el diccionario
    - Podemos dar un **orden específico a las llaves** para recorrer los elementos de acuerdo a ese orden

PRÓXIMA CLASE

- Practicaremos resolviendo problemas del mundo real: **repasar abstracción de procesos**
- Los problemas requerirán definir estructuras de datos abstractas: **repasar abstracción de datos**
- Deberemos implementar la solución en Python: **repasar objetos nativos**

¿CONSULTAS?