Trabajo en grupo - CheatSheet

Tipos de datos permitidos

- Tipos numéricos: `int` y `float`
- Cadenas de caracteres o strings: `str`
- Booleanos: `bool`
- Vínculo de archivo generado por la función `open`.

Datos estructurados (Colecciones)

- Listas (`list`), lista dinámica ordenada por posición.
- Conjuntos (`set`), conjunto sin orden.
- Diccionarios (`dict`), colección del tipo `<clave:valor>`.

Operadores

Operadores permitidos en Python:

- Aritméticos: `+`, `-`, `*`, `/`, `//`, `%` y `**`
- Cadenas: `+` y `*`
- Asignación: `=`
- Combinaciones de asignación y operadores aritméticos: `+=`, `-=`, `*=`, `/=`, `//=`, `%=` y `**=`
- Comparaciones: `==`, `!=`, `>=`, `>`, `<=` y `<`
- Lógicos: `and`, `or` y `not`
- Pertenencia: `in` y `not in`

Estructuras

Definición de funciones

```
def cualquier_nombre(arg1, arg2, ...):
    [Lista de instrucciones]
    return ald1, ald2
```

Estructura `if-elif-else`

```
if expresionLogicaA:
    [ListaDeInstruccionesA]
elif expresionLogicaB:
    [ListaDeInstruccionesB]
else:
    [ListaDeInstruccionesC]
```

Estructura `for`

```
for x in list/set/dict.keys():
    [Lista de instrucciones]
```

Estructura `while`

```
while expresionLogica:
    [Lista de instrucciones]
```

Funciones permitidas

Para mostrar información: `print(valores, sep=' ', end='\n')`
Para calcular la longitud de cualquier colección o cadena de caracteres: `len(valor)`
Para ver el tipo de un valor: `type(valor)`

Cambios de tipo de datos

Con el valor `valor` que admite cambios:

- `int(valor)`
- `float(valor)`
- `str(valor)`
- `list(valor)`
- `set(valor)`
- `dict(valor)`

Funciones sobre cadenas de caracteres

- `str.isdigit(texto)`
- `str.isdecimal(texto)`
- `str.isalpha(texto)`
- `str.isalnum(texto)`
- `str.isupper(texto)`
- `str.islower(texto)`
- `str.count(texto,texto_a_contar)`
- `str.upper(texto)`
- `str.lower(texto)`
- `str.capitalize(texto)`
- `str.split(separador, texto_a_dividir)`
- `str.join(separador, lista_a_unir)`

Funciones sobre listas

Las siguientes funciones modifican listas pero no requieren asignación para hacer el cambio.

- `lista.append(variable)`
- `lista.extend(variable)`
- `lista.insert(posicion, variable)`
- `lista.remove(variable)`
- `del lista[posicion]`

Operaciones y funciones sobre conjuntos

Modifican conjuntos pero no requieren asignación para hacer el cambio.

- Unión de dos conjuntos ($A \cup B$): `A | B`
- Intersección de dos conjuntos ($A \cap B$): `A & B`
- Diferencia de dos conjuntos ($A - B$): `A - B`
- Agregar un elemento: `conjunto.add(variable)`
- Eliminar un elemento: `conjunto.discard(variable)`

Operaciones y funciones sobre diccionarios

Modifican diccionarios pero no requieren asignación para hacer el cambio.

- `diccionario[clave]=valor` ⟶ Agregar/asignar valor
- `del diccionario[clave]` ⟶ Eliminar elemento
- `diccionario.keys()` ⟶ Obtener claves

Funciones sobre archivos

- `ftx = open(direccion,'r')` ⟶ Abrir en modo lectura
- `ftx = open(direccion,'w')` ⟶ Abrir en modo escritura
- `ftx.readline()` ⟶ Leer una línea
- `ftx.write(valor)` ⟶ Escribir valor
- `ftx.close()` ⟶ Cerrar archivo
- `import os` y `os.path.isfile(direccion)` ⟶ Verificar si el archivo existe

Leer todas las líneas de un archivo y mostrar en pantalla:

```
import os
if os.path.isfile(direccion):
    ftx = open(direccion,'r')
    linea = ftx.readline()
    while linea != '':
        print(linea[:-1])
        linea = ftx.readline()
    ftx.close()
else:
    print('El archivo no existe')
```

Escribir líneas en un archivo:

```
ftx = open(direccion,'w')
lineas = ['uno','dos','tres']
for x in lineas:
    ftx.write(x+'\n')
ftx.close()
```