DEPARTAMENTO DE
INGENIERÍA
INFORMÁTICA
UNIVERSIDAD DE SANTIAGO DE CHILE

ARCHIVOS

10145 - FUNDAMENTOS DE PROGRAMACIÓN PARA INGENIERÍA
10110 – FUNDAMENTOS DE COMPUTACIÓN Y PROGRAMACIÓN

RESUMEN DE CONTENIDOS

APERTURA DE ARCHIVOS

Para abrir archivos utilizaremos el ambiente **with**:

```
with open('dir/nombre_archivo', <modo>) as archivo:
    <tareas a ejecutar con el archivo>
```

- **'dir/nombre_archivo':** Dirección y nombre del archivo como string. Por ejemplo **"C://Documentos/entrada.txt"**
- **<modo>:** Sigla(s) que indica(n) la forma en que el programa utiliza el archivo y debe ser entregado como string.
- En el curso veremos tres: **r**, **w** y **a**

MODOS DE ACCESO

- **r : Reading**, indica que el archivo será abierto en modo de lectura. En este modo:
  - El objeto **FILE podrá leer información del archivo**, pero no estará autorizado para modificarlo
  - En caso de **no existir el archivo el programa arrojará un error**
  - Este modo es el **modo por defecto** si en la función open se omite el modo de acceso

MODOS DE ACCESO

- **w : Writing**, indica que el archivo será abierto en modo de escritura. En este modo:
  - El objeto **FILE podrá escribir información en el archivo**
  - En caso de no existir el archivo, el programa **creará un archivo en blanco** con el nombre indicado, **en el mismo directorio** en el que está el programa que ejecuta la instrucción
  - En caso de existir el archivo, el programa **borrará el contenido del archivo**

MODOS DE ACCESO

- **a : Append**, indica que el archivo será abierto en modo de añadidura o agregación. En este modo:
  - El objeto **FILE podrá escribir información en el archivo**
  - En caso de no existir el archivo, el programa creará un archivo en blanco con el nombre indicado, de forma similar al modo writting
  - En caso de existir el archivo, el programa **escribirá luego del contenido del archivo**

MÉTODOS DEL OBJETO FILE

- Con lo anterior hemos establecido **una forma de comunicarnos con el archivo**. Pero para poder escribir o leer el archivo debemos utilizar métodos de estos objetos creados.

- `<file>.close()`: indica que el archivo administrado debe cerrarse. Esto es necesario para que el archivo se almacene en el dispositivo físico respectivo del equipo. Es importante notar que no es necesario utilizar este método cuando se abre archivos con el ambiente `with`

ACCEDER A INFORMACIÓN DE UN ARCHIVO

DEPARTAMENTO DE INGENIERÍA INFORMÁTICA
UNIVERSIDAD DE SANTIAGO DE CHILE

- Los objetos `file` son un nuevo tipo iterable, así como los objetos `string` y `list`

- Sin embargo, los objetos `file` no responden a un orden dado por índices como nuestros dos referentes iterables previos

- Por esta razón requerimos de una nueva herramienta que nos permita recorrer la información de un iterable sin depender de un índice

MÉTODOS DE LECTURA

- Sólo pueden ser usados si el archivo está abierto en modo `'r'`:
  - **`<file>.readline()`:** Lee la siguiente línea en un archivo abierto y la **devuelve como un string**
  - **`<file>.readlines()`:** Lee todas las líneas del archivo abierto y las **devuelve como una lista de strings**
  - **`<file>.read()`:** Lee todas las líneas del archivo abierto y las **devuelve como un único string**

MÉTODOS DE ESCRITURA

- Sólo pueden ser usados si el archivo está abierto en modo `'w'`:
  - **`<file>.write(<string>)`:** escribe el contenido de un **único string** en el archivo. No introduce saltos de línea a menos que usemos el carácter fin de línea (`'\n'`). Sólo es válido cuando un archivo permanece abierto en **modo de escritura o de agregación**
  - ¡Hay otra forma para escribir en un archivo y ya la conocíamos!
    - Usando `print()`

ESCRIBIR CON `print()`

- La función `print()` tiene algunos parámetros interesantes de los que no han sido mencionados hasta el momento

- Estos parámetros especiales son de tipo ***keyword***, es decir, debo usar su nombre para poder usarlos y son:
  - **sep:** indica cuál es el caracter para separar los elementos a imprimir (por defecto es el caracter espacio)
  - **end:** indica cuál es el caracter de fin (por defecto es un salto de línea)
  - **file:** indica en qué archivo queremos guardar el resultado del `print()` habíamos visto anteriormente que la consola de Python es un archivo más para el computador y puede ser modificado

ESCRIBIR CON `print()`

- Con esto podemos usar simplemente `print()` para escribir en un archivo, un ejemplo de esto sería:

```
nota_pedro = 5.0
nota_juan = 6.7
with open('example.txt', 'w') as arc:
    print('Pedro', nota_pedro, sep=': ', file=arc)
    print('Juan', nota_juan, sep=': ', file=arc)
```

- Lo que escribiría en un archivo de salida:

```
Pedro: 5.0
Juan: 6.7
```

- Noten que el parámetro **`end`**, no ha sido modificado, por lo que sigue escribiendo en líneas separadas

EJERCICIOS

EJERCICIO PROPUESTO 1 – ENTRADA Y SALIDA

- Construya un programa en Python que cuente cuantas vocales y cuantas consonantes hay en un texto en español
  - Considere que las vocales podrían tener tildes
  - El texto puede tener más de una línea
  - Las letras pueden ser mayúsculas y minúsculas
  - Considere que el texto siempre estará en un archivo llamado ‘entrada.txt’

- El resultado debe ser escrito en un archivo ‘salida.txt’ y debe tener el siguiente formato:

```
En el archivo se encontraron:
X consonantes
Y vocales
```

EJERCICIO PROPUESTO 2 – ENTRADA

- Construya un programa en Python que encuentre la palabra más larga de un texto. Considere que el texto viene en un archivo llamado ‘text.txt’

EJERCICIO PROPUESTO 3 - SALIDA

- Construya un programa en Python que reciba como entrada una cantidad *n* de filas y una cantidad *m* de columnas y entregue como resultado un archivo con una matriz con números aleatorios de dichas dimensiones, el archivo de salida debe llamarse ‘matriz.txt’ y cada celda debe contener un número aleatorio entre -n * m y n * m

- Cada celda de la matriz debe ir separada por una coma

¿CONSULTAS?