Insaforp
FORMANDO UN MEJOR EL SALVADOR

FORMÁS
CENTRO DE FORMACIÓN AVANZADO PARA PROFESIONALES

FEDISAL
Fundación para la Educación Integral Salvadoreña

MODULO 1. INTRODUCCIÓN AL DESARROLLO DE SOFTWARE CON PYTHON

1. PROGRAMACIÓN ESTRUCTURADA CON PYTHON

CONTENIDOS A DESARROLLAR

1. Funciones
2. Documentación de funciones
3. Módulos y Paquetes
4. __main__ function y argumentos en línea de comandos
5. Manejo de excepciones
6. Clases y objetos

PASO POR VALOR/REFERENCIA?

PASO POR VALOR/REFERENCIA?

Si conoces lenguajes de programación como C, los conceptos de paso por valor o referencia te resultarán familiares:

Los tipos **inmutables** son pasados por **valor**, por lo tanto dentro de la función se accede a una copia y no al valor original.

Los tipos **mutables** son pasados por **referencia**, como es el caso de las listas y los diccionarios. Algo similar a como C pasa las array como punteros.

FUNCIONES

- Anteriormente hemos usado funciones nativas que vienen con Python como len() para calcular la longitud de una lista, pero al igual que en otros lenguajes de programación, también podemos definir nuestras propias funciones. Para ello hacemos uso de def.
- Cualquier función tendrá un **nombre**, unos **argumentos de entrada**, un **código** a ejecutar y unos **parámetros de salida**
- **Reusabilidad**
- **Modularidad**

FUNCIONES

- podemos definir nuestras propias funciones. Para ello hacemos uso de def.

```
def nombre_funcion(argumentos):
    código
    return retorno
```

- Cualquier función tendrá un nombre, unos argumentos de entrada, un código a ejecutar y unos parámetros de salida.

OTROS TEMAS DE FUNCIONES

- Valores por default
- Argumentos variables
- Recursividad

DOCUMENTACIÓN

```
def mi_funcion_suma(a: int, b: int) -> int:

    """ Descripción de la función. Como debe ser usada, que parámetros acepta
    y que devuelve """

    return a+b
```

MODULOS

- librerías de funciones y clases para ser importadas y utilizadas en otro programa.
- Un módulo es un archivo que contiene definiciones y declaraciones de Python.
- El nombre del archivo es el nombre del módulo con el sufijo .py añadido.
- Dentro de un módulo, el nombre del módulo (como una cadena) está disponible como el valor de la variable global __name__.
- Por ejemplo, un archivo llamado aritme.py se importaría así:

```
import aritme
```

MODULOS

- Los módulos pueden importar otros módulos.
- Es habitual, pero no obligatorio, colocar todas las declaraciones de importación al comienzo de un módulo (o script, para el caso).
- Los nombres de los módulos importados se colocan en la tabla de símbolos global del módulo de importación.
- Existe una variante de la declaración de importación que importa nombres de un módulo directamente a la tabla de símbolos del módulo de importación. Por ejemplo:

```
from aritme import suma, resta

from aritme import *

import aritme as ari

from aritme import suma as sum
```

PACKAGES

- Los paquetes son una forma de estructurar el espacio de nombres de los módulos de Python mediante el uso de "nombres de módulos con puntos".
- Por ejemplo, el nombre del módulo A.B designa un submódulo llamado B en un paquete llamado A. Al igual que el uso de módulos evita que los autores de diferentes módulos tengan que preocuparse por los nombres de las variables globales de los demás, el uso de nombres de módulos con puntos salva a los autores de paquetes de varios módulos como NumPy o Pillow de tener que preocuparse por los nombres de los módulos de cada uno.

PACKAGES

```
sound/                          Top-level package
      __init__.py               Initialize the sound package
      formats/                  Subpackage for file format conversions
              __init__.py
              wavread.py
              wavwrite.py
              aiffread.py
              aiffwrite.py
              auread.py
              auwrite.py
              ...
      effects/                  Subpackage for sound effects
              __init__.py
              echo.py
              surround.py
              reverse.py
              ...
      filters/                  Subpackage for filters
              __init__.py
              equalizer.py
              vocoder.py
              karaoke.py
              ...
```

- import sound.effects.echo
- import sound.effects.surround
- from sound.effects import *
- from sound.effects.echo import echofilter

PACKAGES

- Cuando el intérprete ejecuta la declaración de importación anterior, busca mod.py en una lista de directorios ensamblados a partir de las siguientes fuentes:
- El directorio desde el que se ejecutó el script de entrada o el directorio actual si el intérprete se ejecuta de forma interactiva
- La lista de directorios contenidos en la variable de entorno PYTHONPATH, si está configurada. (El formato de PYTHONPATH depende del sistema operativo, pero debe imitar la variable de entorno PATH).
- Una lista dependiente de la instalación de directorios configurados en el momento de instalar Python

```
import sys

sys.path.append(r'D:\Desktop\Modulo1Python')

print(str(sys.path))
```

__MAIN__

En scripts de python es comun encontrar lo siguiente:

```
def main():
    print("Hello World!")

if __name__ == "__main__":
    main()
```

MODOS DE EJECUCION EN PYTHON

1. Puede ejecutar el archivo Python como un script usando la línea de comando.
2. Puede importar el código de un archivo de Python a otro archivo o al intérprete interactivo.

Python define una variable especial llamada __name__ que contiene una cadena cuyo valor depende de cómo se esté usando el código.

Script: __name__=='__main_'

Modulo(test): (__name__=='test')

MODOS DE EJECUCION EN PYTHON

- Resulta útil saber esto porque se puede escribir un programa que se puede ejecutar como script (__name__=='__main_') e importarlo como modulo (__name__=='test')
- Usar if __name__ == "__main__" para controlar la ejecución del código

- Al ser llamado como script se ejecutaría el programa con la función definida dentro de la condición anterior.
- Al ser importado no haría nada mas que dejar disponibles las otras funciones para ser usadas como parte de un modulo.