**UNIVERSIDAD DE GUADALAJARA**

**CUCEI**

**DIVISIÓN DE ELECTRÓNICA Y COMPUTACIÓN**

**DEPARTAMENTO DE CIENCIAS COMPUTACIONALES**

**TAREA No. 5**

*TEMA***: DICCIONARIO Y CONJUNTOS**

*INTEGRANTE*: **ROBLES PULIDO EFRAIN**

*NOMBRE DE LA MATERIA:* **FUNDAMENTOS FILOSOFICOS DE LA COMPUTACION**

*SECCIÓN:* **D13**  *CALENDARIO:* **2021A**

*NOMBRE DEL PROFESOR:* **MARISCAL LUGO LUIS FELIPE**





**Marco Teórico**

**10.1 Diccionarios**

Un diccionario es un objeto que almacena una colección de datos. Cada elemento de un diccionario tiene dos partes: una clave y un valor. Utiliza una clave para localizar un valor específico.

Cada elemento que se almacena en un diccionario tiene dos partes: una clave y un valor. De hecho, los elementos del diccionario se denominan comúnmente pares clave-valor. Cuando desee recuperar un valor específico de un diccionario, utilice la clave asociada con ese valor.

**Creando un diccionario**

Puede crear un diccionario encerrando los elementos dentro de un conjunto de llaves ({}). Los elementos están separados por comas.



Esta declaración crea un diccionario y lo asigna a la variable de la agenda. El diccionario contiene los siguientes tres elementos:

* El primer elemento es 'Chris': '555-1111'. En este elemento, la clave es 'Chris' y el valor es '555-1111'.
* El segundo elemento es 'Katie': '555-2222'. En este elemento, la clave es 'Katie' y el valor es '555-2222'.
* El tercer elemento es 'Joanne': '555-3333'. En este elemento, la clave es 'Joanne' y el valor es '555-3333'.

En este ejemplo, las claves y los valores son cadenas. Los valores de un diccionario pueden ser objetos de cualquier tipo, pero las claves deben ser objetos inmutables.

**Recuperar un valor de un diccionario**

Los elementos de un diccionario no se almacenan en ningún orden en particular. Los diccionarios no son secuencias, como listas, tuplas y cadenas. Como resultado, no puede usar un índice numérico para recuperar un valor por su posición de un diccionario. En su lugar, usa una clave para recuperar un valor.

Para recuperar un valor de un diccionario, simplemente escriba una expresión en el siguiente formato general:

****

En el formato general, *dictionary\_name* es la variable que hace referencia al diccionario y *key* es una clave.

Si la clave existe en el diccionario, la expresión devuelve el valor asociado con la llave. Si la clave no existe, se genera una excepción KeyError. La siguiente sesión interactiva demuestra:

**Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente**

* En la línea 2, la expresión directorio telefónico ['Chris'] devuelve el valor del diccionario del directorio telefónico asociado con la tecla 'Chris'. El valor se muestra en la línea 3.
* En la línea 4, la expresión directorio telefónico ['Joanne'] devuelve el valor del diccionario del directorio telefónico asociado con la tecla 'Joanne'. El valor se muestra en la línea 5.
* En la línea 6, la expresión directorio telefónico ['Katie'] devuelve el valor del diccionario del directorio telefónico asociado con la tecla 'Katie'. El valor se muestra en la línea 7.
* En la línea 8, se ingresa la expresión directorio telefónico ['Kathryn']. No existe una clave como 'Kathryn' en el diccionario de la agenda, por lo que se genera una excepción KeyError.

**Uso de los operadores in y not in para probar un valor en un diccionario**

Se generará una excepción KeyError si se intenta recuperar un valor de un diccionario utilizando una clave inexistente. Para evitar una excepción de este tipo, puede utilizar el operador in para determinar si existe una clave antes de intentar utilizarla para recuperar un valor. Como a continuación se mostrará:

**Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente**

La instrucción if en la línea 2 determina si la clave 'Chris' está en el diccionario de la agenda. Si es así, la declaración en la línea 3 muestra el valor asociado con esa clave.

También se puede utilizar el operador not in para determinar si una clave no existe, como se demuestra en la siguiente sesión:

**Texto

Descripción generada automáticamente**

Los diccionarios son objetos mutables. Puede agregar nuevos pares clave-valor a un diccionario con una declaración de asignación en el siguiente formato general:



En el formato general, *dictionary\_name* es la variable que hace referencia al diccionario y *key* es una clave. Si la clave ya existe en el diccionario, su valor asociado se cambiará a valor. Si la clave no existe, se agregará al diccionario, junto con el valor como su valor asociado. Como se mostrará a continuación:

**Interfaz de usuario gráfica, Texto

Descripción generada automáticamente**

* En la línea 1 crea un diccionario que contiene nombres (como claves) y números de teléfono (como valores).
* La declaración en la línea 2 agrega un nuevo par clave-valor al diccionario de la agenda telefónica. Debido a que no hay una clave 'Joe' en el diccionario, esta declaración agrega la clave 'Joe', junto con su valor asociado '555-0123'.
* La declaración en la línea 3 cambia el valor asociado con una clave existente. Debido a que la clave 'Chris' ya existe en el diccionario de la agenda telefónica, esta instrucción cambia su valor asociado a '555-4444'.

**Eliminar elementos**

Puede eliminar un par clave-valor existente de un diccionario con la instrucción del. Aquí está el formato general:



En el formato general, *dictionary\_name* es la variable que hace referencia al diccionario y *key* es una clave. Una vez que se ejecuta la instrucción, la clave y su valor asociado se eliminarán del diccionario. Si la clave no existe, se genera una excepción KeyError. Como se demuestra a continuación:

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Para evitar que se genere una excepción KeyError, se debe de usar el operador in para determinar si existe una clave antes de intentar eliminarla y su valor asociado. Como se demuestra a continuación:

**Interfaz de usuario gráfica, Texto, Aplicación, Word

Descripción generada automáticamente**

**Obtener el número de elementos en un diccionario**

Se puede utilizar la función *len* incorporada para obtener el número de elementos en un diccionario. Como se demuestra a continuación:

****

****

**Mezcla de tipos de datos en un diccionario**

Las claves en un diccionario deben ser objetos inmutables, pero sus valores asociados pueden ser cualquier tipo de objeto.

Los valores que se almacenan en un solo diccionario pueden ser de diferentes tipos. Por ejemplo, el valor de un elemento podría ser una cadena, el valor de otro elemento podría ser una lista y el valor de otro elemento podría ser un número entero. Las claves también pueden ser de diferentes tipos, siempre que sean inmutables. La siguiente sesión interactiva demuestra cómo se pueden mezclar diferentes tipos en un diccionario:

**Imagen que contiene Word

Descripción generada automáticamente**

Esta declaración en la línea 1 crea un diccionario y lo asigna a la variable mixed\_up. El diccionario contiene los siguientes elementos:

* El primer elemento es 'abc': 1. En este elemento, la clave es la cadena 'abc' y el valor es el entero 1.
* El segundo elemento es 999: 'yada yada'. En este elemento, la clave es el número entero 999 y el valor es la cadena 'yada yada'.
* El tercer elemento es (3, 6, 9): [3, 6, 9]. En este elemento la clave es la tupla (3, 6, 9) y el valor es la lista [3, 6, 9].

**Crear un diccionario vacío**

A veces es necesario crear un diccionario vacío y luego agregarle elementos a medida que se ejecuta el programa. Puede utilizar un conjunto vacío de llaves para crear un diccionario vacío, como se muestra a continuación:

****

**Texto

Descripción generada automáticamente**

La declaración en la línea 1 crea un diccionario vacío y lo asigna a la variable de la agenda. Las líneas 2 a 4 agregan pares clave-valor al diccionario y la declaración en la línea 5 muestra el contenido del diccionario.

También puede utilizar el método dict () incorporado para crear un diccionario vacío, como se muestra a continuación:

****

**Uso del bucle for para iterar sobre un diccionario**

Se puede utilizar el bucle for en el siguiente formato general para iterar sobre todas las claves en un diccionario. En el formato general, var es el nombre de una variable y diccionario es el nombre de un diccionario. Este ciclo se repite una vez para cada elemento del diccionario. Cada vez que el bucle se repite, se asigna una clave a var. El siguiente código se demuestra:

**Interfaz de usuario gráfica, Texto, Aplicación, Chat o mensaje de texto

Descripción generada automáticamente**

**Algunos métodos de diccionario**

Los objetos de diccionario tienen varios métodos. En la Tabla 10-1, examinaremos algunos de los más útiles.

**Interfaz de usuario gráfica, Texto

Descripción generada automáticamente**

**El método *clear***

El método *clear* elimina todos los elementos de un diccionario, dejando el diccionario vacío. El formato general del método es:

****

Como se muestra a continuación:

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

***El método get***

Puede utilizar el método get como alternativa al operador [] para obtener un valor de un diccionario. El método get no genera una excepción si no se encuentra la clave especificada. Este es el formato general del método:

****

En el formato general, *dictionary* es el nombre de un diccionario, *key* es una clave para buscar en el diccionario y *default* es un valor predeterminado para devolver si no se encuentra la clave. Cuando se llama al método, devuelve el valor asociado con la clave especificada. Si la clave especificada no se encuentra en el diccionario, el método devuelve el valor predeterminado. Por ejemplo:

Texto

Descripción generada automáticamente

**El método *items***

El método *items* devuelve todas las claves de un diccionario y sus valores asociados. Se devuelven como un tipo especial de secuencia conocida como vista de diccionario. Cada elemento de la vista de diccionario es una tupla y cada tupla contiene una clave y su valor asociado. Por ejemplo, supongamos que hemos creado el siguiente diccionario:

****

Si llamamos al método *phonebook.items ()*, devuelve la siguiente secuencia, como tuplas:

****

Puede usar el bucle for para iterar sobre las tuplas en la secuencia. Como se probara a continuación:

**Texto

Descripción generada automáticamente**

El bucle *for* en las líneas 4 a 5 llama al método *phonebook.items (),* que devuelve una secuencia de tuplas que contienen los pares clave-valor en el diccionario. El ciclo se repite una vez para cada tupla de la secuencia. Cada vez que el bucle se repite, los valores de una tupla se asignan a las variables clave y de valor. La línea 5 imprime el valor de la variable clave, seguido del valor de la variable de valor. Las líneas 8 a 10 muestran la salida del ciclo.

**El método *Keys***

El método *Keys* devuelve todas las claves de un diccionario como una vista de diccionario, que es un tipo de secuencia. Cada elemento de la vista del diccionario es una clave del diccionario. Por ejemplo, supongamos que hemos creado el siguiente diccionario:



Si llamamos al método *phonebook.keys ()*, devolverá la siguiente secuencia:



El siguiente código muestra cómo puede usar un bucle for para iterar sobre la secuencia que se devuelve desde el método *keys*:

**Interfaz de usuario gráfica, Texto

Descripción generada automáticamente**

**El método pop**

El *método pop* devuelve el valor asociado con una clave especificada y elimina ese par de valores clave del diccionario. Si no se encuentra la clave, el método devuelve un valor predeterminado. Este es el formato general del método:



En el formato general, *dictionary* es el nombre de un diccionario, *key* es una clave para buscar en el diccionario y el *default* es un valor predeterminado para devolver si no se encuentra la clave.

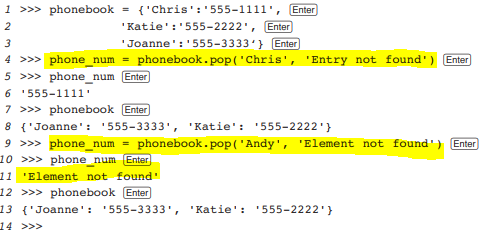
Cuando se llama al método, devuelve el valor asociado con la clave especificada y elimina ese par clave-valor del diccionario. Si la clave especificada no se encuentra en el diccionario, el método devuelve el valor predeterminado. Como se muestra a continuación:

**Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente**

En el formato general, *dictionary* es el nombre de un diccionario, *key* es una clave para buscar en el diccionario y *default* es un valor predeterminado para devolver si no se encuentra la clave.

Cuando se llama al método, devuelve el valor asociado con la clave especificada y elimina ese par clave-valor del diccionario. Si la clave especificada no se encuentra en el diccionario, el método devuelve el valor predeterminado. Por ejemplo:

****

**El método popitem**

El *método popitem* devuelve un par clave-valor seleccionado aleatoriamente y elimina ese par clave-valor del diccionario. El par clave-valor se devuelve como una tupla. Este es el formato general del método:



Puede utilizar una declaración de asignación en el siguiente formato general para asignar la clave y el valor devueltos a variables individuales:

****

Este tipo de asignación se conoce como asignación múltiple porque se asignan múltiples variables a la vez. En el formato general, k y v son variables. Una vez que se ejecuta la instrucción, a k se le asigna una clave seleccionada al azar del diccionario, y a v se le asigna el valor asociado con esa clave. El par clave-valor se elimina del diccionario. Por ejemplo:

**Interfaz de usuario gráfica, Texto, Aplicación, Chat o mensaje de texto

Descripción generada automáticamente**

* La línea 6 llama al método *phonebook.popitem ()*. La clave y el valor que se devuelven del método se asignan a la clave y el valor de las variables. El par clave-valor se elimina del diccionario.
* La línea 7 muestra los valores asignados a la clave y las variables de valor. La salida se muestra en la línea 8.
* La línea 9 muestra el contenido del diccionario. El resultado se muestra en la línea 10. Observe que el par clave-valor que se devolvió del método popitem en la línea 6 se ha eliminado. Tenga en cuenta que el método popitem genera una excepción KeyError si se llama en un diccionario vacío.

**El Método *values***

El *método values* devuelve todos los valores de un diccionario (sin sus claves) como una vista de diccionario, que es un tipo de secuencia. Cada elemento de la vista del diccionario es un valor del diccionario. Por ejemplo, supongamos que hemos creado el siguiente diccionario:



Si llamamos al método phonebook.values (), devuelve la siguiente secuencia:



También se puede usar un bucle *for* para iterar sobre la secuencia que se devuelve desde el método values:

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

**10.2 Conjuntos**

Un conjunto contiene una colección de valores únicos y funciona como un conjunto matemático.

* Todos los elementos de un conjunto deben ser únicos. No hay dos elementos que tengan el mismo valor.
* Los conjuntos están desordenados, lo que significa que los elementos de un conjunto no se almacenan en ningún orden en particular.
* Los elementos que se almacenan en un conjunto pueden ser de diferentes tipos de datos.

**Crear un conjunto**

Para crear un conjunto, debe llamar a la función de conjunto incorporada. A continuación, se muestra un ejemplo de cómo se crea un conjunto vacío:



Después de que se ejecute esta declaración, la variable *myset* hará referencia a un conjunto vacío. También puede pasar un argumento a la función set. El argumento que pase debe ser un objeto que contenga elementos iterables, como una lista, una tupla o una cadena. Los elementos individuales del objeto que pasa como argumento se convierten en elementos del conjunto. Aquí hay un ejemplo



En este ejemplo, estamos pasando una lista como argumento a la función set. Después de que se ejecuta esta declaración, la variable myset hace referencia a un conjunto que contiene los elementos 'a', 'b' y 'c'.

Si pasa una cadena como argumento a la función set, cada carácter individual en la cadena se convierte en miembro del conjunto. Aquí hay un ejemplo:



Después de que se ejecute esta declaración, la variable myset hará referencia a un conjunto que contiene los elementos 'a', 'b' y 'c'.

Los conjuntos no pueden contener elementos duplicados. Si pasa un argumento que contiene elementos duplicados a la función de conjunto, solo uno de los elementos duplicados aparecerá en el conjunto. Aquí hay un ejemplo:



El carácter 'a' aparece varias veces en la cadena, pero solo aparecerá una vez en el conjunto. Después de que se ejecute esta declaración, la variable myset hará referencia a un conjunto que contiene los elementos 'a', 'b' y 'c'.

**Obtener el número de elementos de un conjunto**

Al igual que con las listas, tuplas y diccionarios, puede utilizar la función len para obtener el número de elementos de un conjunto. Por ejemplo:

Interfaz de usuario gráfica, Aplicación, Word

Descripción generada automáticamente

**Agregar y eliminar elementos**

Los conjuntos son objetos mutables, por lo que puede agregarles elementos y eliminarlos. Utiliza el método add para agregar un elemento a un conjunto. Por ejemplo:

Texto, Chat o mensaje de texto

Descripción generada automáticamente

Las líneas 2 a 4 suman los valores 1, 2 y 3 al conjunto. La línea 5 muestra el contenido del conjunto, que se muestra en la línea 6.

En la línea 7 intenta agregar el valor 2 al conjunto. Sin embargo, el valor 2 ya está en el conjunto. Si intenta agregar un elemento duplicado a un conjunto con el método *add*, el método no genera una excepción. Simplemente no agrega el artículo.

Puede agregar un grupo de elementos a un conjunto a la vez con el método de actualización. Cuando llamas al método de actualización como argumento, pasas un objeto que contiene elementos iterables, como una lista, una tupla, una cadena u otro conjunto. Los elementos individuales del objeto que pasa como argumento se convierten en elementos del conjunto. Por ejemplo:

Interfaz de usuario gráfica, Texto, Word

Descripción generada automáticamente

Para eliminar un elemento de un conjunto con el método de eliminación o el método de descarte. Pasa el elemento que desea eliminar como argumento a cualquiera de los métodos y ese elemento se elimina del conjunto. La única diferencia entre los dos métodos es cómo se comportan cuando el elemento especificado no se encuentra en el conjunto. El método de eliminación genera una excepción KeyError, pero el método de descarte no genera una excepción. Por ejemplo:

Interfaz de usuario gráfica, Texto, Aplicación, Chat o mensaje de texto

Descripción generada automáticamente

La línea 4 llama al método *remove* para eliminar el valor 1 del conjunto. Puede ver en el resultado que se muestra en la línea 6 que el valor 1 ya no está en el conjunto.

La línea 7 llama al método de *descarte* para eliminar el valor 5 del conjunto. Puede ver en el resultado de la línea 9 que el valor 5 ya no está en el conjunto. La línea 10 llama al método de descarte para eliminar el valor 99 del conjunto. El valor no se encuentra en el conjunto, pero el método de descarte no plantea una excepción. La línea 11 llama al método de eliminación para eliminar el valor 99 del conjunto. Debido a que el valor no está en el conjunto, se genera una excepción KeyError, como se muestra en las líneas 12 a 15.

También se puede borrar todos los elementos de un conjunto llamando al método *clear*. Como se demostrará a continuación:

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

En la línea 4 llama al método claro para borrar el conjunto. Observe en la línea 6 que cuando mostramos el contenido de un conjunto vacío, el intérprete muestra conjunto ().

**Uso del bucle for para iterar sobre un conjunto**

Puede usar el bucle for en el siguiente formato general para iterar sobre todos los elementos en un conjunto. En el formato general, var es el nombre de una variable y conjunto es el nombre de un conjunto. Este bucle se repite una vez para cada elemento del conjunto. Cada vez que el ciclo se repite, a var se le asigna un elemento. Como se muestra a continuación:

Texto

Descripción generada automáticamente

Las líneas 2 a 3 contienen un bucle for que se repite una vez para cada elemento del conjunto myset. Cada vez que el bucle se repite, se asigna un elemento del conjunto a la variable val. La línea 3 imprime el valor de la variable val. Las líneas 5 a 7 muestran la salida del ciclo.

**Uso de operadores in y not in para probar un valor en un conjunto**

Se puede utilizar el operador in para determinar si existe un valor en un conjunto.

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

La instrucción if en la línea 2 determina si el valor 1 está en el conjunto myset. Si es así, la declaración en la línea 3 muestra un mensaje.

También puede utilizar el operador not in para determinar si un valor no existe en un conjunto, por ejemplo:

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

**Encontrar la unión de conjuntos**

La unión de dos conjuntos es un conjunto que contiene todos los elementos de ambos conjuntos. En Python, puede llamar al método union para obtener la unión de dos conjuntos. Aquí está el formato general:



En el formato general, set1 y set2 son conjuntos. El método devuelve un conjunto que contiene los elementos de set1 y set2. Por ejemplo:

Texto

Descripción generada automáticamente

La declaración en la línea 3 llama al método de intersección del objeto set1, pasando set2 como argumento. El método devuelve un conjunto que contiene los elementos que se encuentran tanto en set1 como en set2. El conjunto resultante se asigna a la variable set3.

También puede usar el operador & para encontrar la intersección de dos conjuntos. Aquí está el formato general de una expresión usando el operador & con dos conjuntos:



En el formato general, set1 y set2 son conjuntos. La expresión devuelve un conjunto que contiene los elementos que se encuentran tanto en set1 como en set2. Por ejemplo:

Interfaz de usuario gráfica

Descripción generada automáticamente con confianza baja

**Encontrar la diferencia de conjuntos**

La diferencia de set1 y set2 son los elementos que aparecen en set1 pero no aparecen en set2. En Python, puede llamar al método de diferencia para obtener la diferencia de dos conjuntos. Aquí está el formato general:



En el formato general, set1 y set2 son conjuntos. El método devuelve un conjunto que contiene los elementos que se encuentran en set1 pero no en set2. Por ejemplo:

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

También puede usar el operador - para encontrar la diferencia de dos conjuntos. Aquí está el formato general de una expresión que usa el operador - con dos conjuntos:



En el formato general, set1 y set2 son conjuntos. La expresión devuelve un conjunto que contiene los elementos que se encuentran en set1 pero no en set2. Por ejemplo:

Interfaz de usuario gráfica

Descripción generada automáticamente con confianza baja

**Encontrar la diferencia simétrica de conjuntos**

La diferencia simétrica de dos conjuntos es un conjunto que contiene los elementos que no son compartidos por los conjuntos. En otras palabras, son los elementos los que están en un conjunto, pero no en ambos. En Python, puede llamar al método symmetric\_difference para obtener la diferencia simétrica de dos conjuntos. Aquí está el formato general:



En el formato general, set1 y set2 son conjuntos. El método devuelve un conjunto que contiene los elementos que se encuentran en set1 o set2 pero no en ambos conjuntos. Por ejemplo:

Texto

Descripción generada automáticamente con confianza baja

También puede usar el operador ^ para encontrar la diferencia simétrica de dos conjuntos. Aquí está el formato general de una expresión usando el operador ^ con dos conjuntos:



En el formato general, set1 y set2 son conjuntos. La expresión devuelve un conjunto que contiene los elementos que se encuentran en set1 o set2 pero no en ambos conjuntos. Por ejemplo:

Texto

Descripción generada automáticamente

**Encontrar subconjuntos y superconjuntos**

Suponga que tiene dos conjuntos y uno de esos conjuntos contiene todos los elementos del otro colocar. Aquí hay un ejemplo



En este ejemplo, set1 contiene todos los elementos de set2, lo que significa que set2 es un subconjunto de set1. También significa que set1 es un superconjunto de set2. En Python, puede llamar al método issubset para determinar si un conjunto es un subconjunto de otro. Aquí está el formato general:



En el formato general, set1 y set2 son conjuntos. El método devuelve true si set2 es un subconjunto de set1. De lo contrario, devuelve false. Puede llamar al método issuperset para determinar si un conjunto es un superconjunto de otro. Aquí está la forma general:



En el formato general, set1 y set2 son conjuntos. El método devuelve true si set1 es un superconjunto de set2. De lo contrario, devuelve false. Por ejemplo:

Texto

Descripción generada automáticamente

También puede utilizar el operador <= para determinar si un conjunto es un subconjunto de otro y el operador> = para determinar si un conjunto es un superconjunto de otro. Aquí está el formato general de una expresión que usa el operador <= con dos conjuntos:



En el formato general, set1 y set2 son conjuntos. La expresión devuelve True si set2 es un subconjunto de set1. De lo contrario, devuelve False. Aquí está el formato general de una expresión que usa el operador> = con dos conjuntos:



En el formato general, set1 y set2 son conjuntos. La expresión devuelve Verdadero si set1 es un subconjunto de set2. De lo contrario, devuelve False. Por ejemplo:

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

# **Bibliografía**

Gaddis, T. (2021). *Starting out with Python* (tercera ed.). Pearson. Recuperado el 13 de Marzo de 2021, de https://fdocuments.in/document/starting-out-with-python-3rd-edition-pdf-firebase-python-programming-2nd-edition.html

**5.- Conclusiones personales**

Pude aprender como podemos manipular datos mediante diccionarios y los conjuntos en donde en los diccionarios podemos localizar los datos a través de las llaves y poder configurar los datos de los conjuntos que tenemos. Además, podemos adaptar nuestros programas para interactuar con los datos mediante el control selectiva (if), el ciclo while y el ciclo for, para cambiar o eliminar los datos por los datos deseados.