# **Estructura de datos tipo diccionario**

Hasta ahora hemos presentado dos estructuras fundamentales de datos en Python: Listas y tuplas. Ahora presentaremos y comenzaremos a utilizar la estructura de datos tipo diccionario.

La estructura de datos tipo diccionario utiliza una clave para acceder a un valor. El subíndice puede ser un entero, un float, un string, una tupla etc.

Podemos relacionarlo con conceptos que conocemos:

* Un diccionario tradicional que conocemos podemos utilizar un diccionario de Python para representarlo. La clave seria la palabra y el valor seria la definición de dicha palabra.
* Una agenda persona también la podemos representar como un diccionario. La fecha seria la clave y las actividades de dicha fecha sería el valor.
* Un conjunto de usuarios de un sitio web podemos almacenarlo en un diccionario. El nombre de usuario seria la clave y como valor podríamos almacenar su mail, clave, fechas de login, etc.

Hay muchos problemas de la realidad que se pueden representar mediante un diccionario de Python.

Recordemos que las listas son mutables y las tuplas inmutables. Un diccionario es una estructura de datos mutable es decir podemos agregar elementos, modificar y borrar.

**Definición de un diccionario por asignación.**

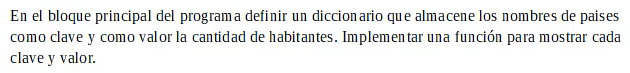
Productos =(“manzanas”:39,”peras”:32,”lechuga”:17)

Print(productos) #{“manzanas”:39,”peras”:32,”Lechuga”:17}

Print(productos[“peras”]) #32

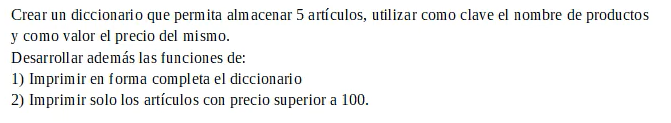
Como vemos debemos encerrar entre llaves los elementos separados por coma. A cada elemento debemos indicar del lado izquierdo del carácter: la clave y al lado derecho el valor asignado para dicha clave. Por ejemplo, para la clave “peras” tenemos asociado el valor entero 32.

**Problema 1:**

****

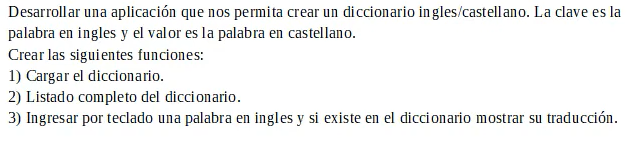
def Imprimir(paises):  
 for clave in paises:  
 print(clave,paises[clave])  
  
paises = {"argentina": 400000,"españa":450000,"brasil":1999999,"uruguay":340000}  
Imprimir(paises)

**Problema 2:**

****

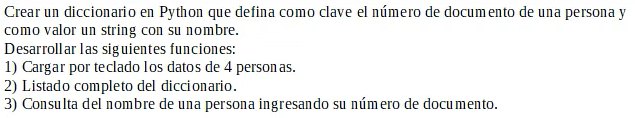
def Cargar():  
 productos = {}  
 for i in range(5):  
 nombre = input("Ingrese nombre: ")  
 precio = int(input("Ingrese el precio: "))  
 productos[nombre]=precio  
 return productos  
def Imprimir(productos):  
 print("Listado de todos los producotos")  
 for i, (clave,valor) in enumerate(productos.items()):  
 print(f"registro nro {i+1}, producto: {clave}, precio {valor}")  
def Imprimir1(productos):  
 for i in productos:  
 print(i, productos[i])  
  
def ImprimirMayores100(productos):  
 for i in productos:  
 if productos[i]>100:  
 print(i)  
productos = Cargar()  
Imprimir1(productos)  
ImprimirMayores100(productos)

**Problema 3:**

****

def Carga():  
 diccionario = {}  
 continua = "s"  
 while continua == "s":  
 ingles = input("Ingrese la palabra en ingles: ")  
 castellano = input("Ingrese la palabra en castellano ")  
 diccionario[ingles]=castellano  
 continua = input("Quiere cargar otra palabra [s/n]")  
 return diccionario  
def Imprimir(diccionario):  
 print("Listado completo del diccionario")  
 for indice in diccionario:  
 print(indice, diccionario[indice])  
def ConsultaPalabra(diccionario):  
 pal = input("Ingrese la palabra en ingles a consultar:")  
 if pal in diccionario:  
 print("En castellano significa",diccionario[pal])  
 else:  
 print("No esta la traduccion de dicha palabra")  
diccionario = Carga()  
Imprimir(diccionario)  
ConsultaPalabra(diccionario)

**Problema 4:**

****

def Cargar():  
 persona = {}  
 for x in range(4):  
 numero = int(input("Ingrese el numero de documento: "))  
 nombre = input("Ingrese el nombre de la persona: ")  
 persona[numero]=nombre  
 return persona  
def Imprimir(personas):  
 for numero in personas:  
 print(numero,personas[numero])  
def ConsultaNumero(personas):  
 consulta = int(input("Ingrese el numero de documento a consultar: "))  
 if consulta in personas:  
 print(f"Numero de la persona {personas[consulta]}")  
 else:  
 print("No existe una persona con dicho numero de nacimiento")  
  
personas = Cargar()  
Imprimir(personas)  
ConsultaNumero(personas)

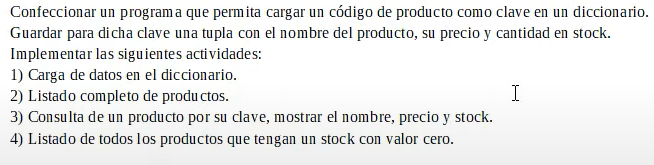
**Diccionarios: con valores de tipo listas, tuplas y diccionarios**

Lo más poderoso que podemos encontrar en la estructura de datos en Python es que podemos definir elementos que sean también estructuras de datos. En general se dice que podemos anidar una estructura de datos dentro de otra estructura de datos.

Ya vimos en conceptos anteriores que podemos definir elementos de una lista que sean también de tipo lista o de tipo tupla.

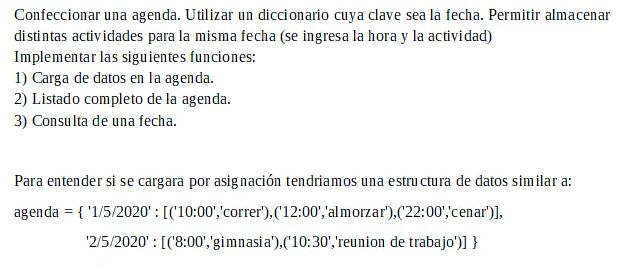
Hemos dicho que un diccionario consta de clave y valores para esas claves. Desarrollaremos problemas donde los valores para esas claves sean tuplas y/o listas.

**Problema 1:**

****

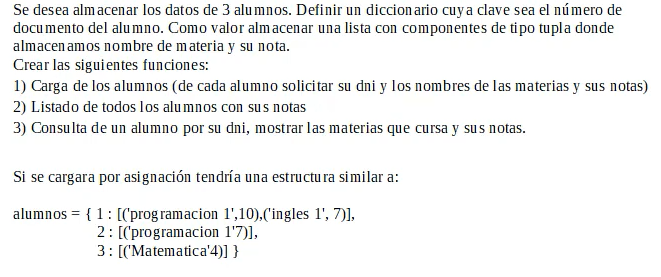
def Cargar():  
 productos = {}  
 continua = "s"  
 while continua=="s":  
 codigo = int(input("codigo: "))  
 descripcion = input("descripcion: ")  
 precio = float(input("precio: "))  
 stock = int(input("stock: "))  
 productos[codigo] = (descripcion,precio,stock)  
 continua = input("Desea cargar otro producto?: ")  
 return productos  
def Imprimir(productos):  
 for codigo in productos:  
 print(codigo, productos[codigo][0], productos[codigo][1], productos[codigo][2])  
  
def Consulta(productos):  
 codigo = int(input("CONSULTA: "))  
 if codigo in productos:  
 print(productos[codigo][0],productos[codigo][1],productos[codigo][2])  
def ListadoStockCero(productos):  
 for codigo in productos:  
 if productos[codigo][2]==0:  
 print(codigo,productos[codigo][0],productos[codigo][1])  
  
productos = Cargar()  
Imprimir(productos)  
Consulta(productos)  
ListadoStockCero(productos)

**Problema 2:**

****

def CargarAgenda():  
 diccionario = {}  
 lista = []  
 continua = "s"  
 continua1 = "s"  
 while continua == "s":  
 fecha = input("fecha: ")  
 continua1 = "s"  
 while continua1=="s":  
 hora = input("hora: ")  
 actividad = input("actividad: ")  
 comprimido = (hora,actividad)  
 lista.append(comprimido)  
 continua1 = input("Continuar?: ")  
 diccionario[fecha] = [lista]  
 continua = input("Agregar otra fecha?: ")  
 return diccionario  
def Imprimir(agenda):  
 for fecha, ListaActividades in agenda.items():  
 print(f"Fecha: {fecha}")  
 for hora,actividad in ListaActividades[0]:  
 print(hora,actividad)  
  
agenda = CargarAgenda()  
Imprimir(agenda)

**Problema 3**

****

def Cargar():  
 alumnos = {}  
 for x in range(3):  
 dni = int(input("Ingrese numero de dni: "))  
 listamaterias = []  
 continua= "s"  
 while continua == "s":  
 materia = input("Ingrese el nombre de la materia: ")  
 nora = int(input("Ingrese la nota: "))  
 listamaterias.append((materia,nora))  
 continua =input("Desea cargar otra materia para dicho alumno?: ")  
 alumnos[dni]=listamaterias  
 return alumnos  
def Listar(alumnos):  
 for dni in alumnos:  
 print("Dni del alumno: ",dni)  
 for nota, materia in alumnos[dni]:  
 print(nota,materia)  
def ConsultaNotas(alumnos):  
 dni = int(input("Ingrese el dni a consultar: "))  
 for dni in alumnos:  
 for materia, nota in alumnos[dni]:  
 print(materia,nota)

**Funciones parámetros mutables e inmutables**

En Python tenemos tipos de datos inmutables:

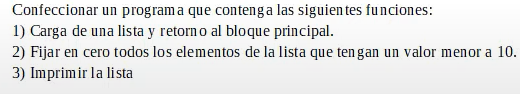
Enteros  
float  
string  
tuplas

Mutables:

Listas  
Diccionarios

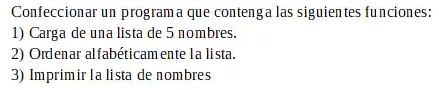
Esto tiene mucha importancia cuando enviamos a una función una variable mutable, veremos con un ejemplo como podemos pasar como parámetro una lista a una función y posteriormente cambiar su contenido y esto se vea reflejado en la variable que le enviamos al llamarla.

**Problema 1:**

****

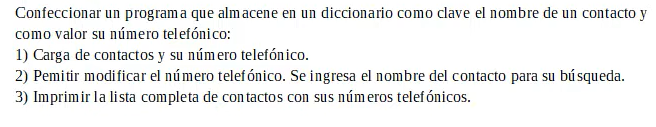
def Cargar():  
 lista = []  
 continua = "s"  
 while continua == "s":  
 valor = int(input("Ingrese valor: "))  
 lista.append(valor)  
 continua = input("CONTINUA?: ")  
 return lista  
def FijarCero(lista):  
 for i,valor in enumerate (lista):  
 if valor<10:  
 lista[i] = 0  
 print(lista)  
numeros = Cargar()  
FijarCero(numeros)

**Problema 2:**

****

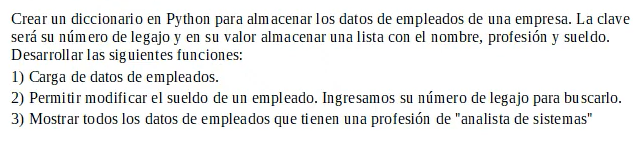
def Cargar():  
 nombre = []  
 for x in range(5):  
 nom = input("Nombre: ")  
 nombre.append(nom)  
 return nombre  
def Ordenar(nombres):  
 for k in range(4):  
 for x in range(4):  
 if nombres[x]>nombres[x+1]:  
 aux = nombres[x+1]  
 nombres[x]=nombres[x+1]  
 nombres[x+1]=aux  
def Imprimir(nombres):  
 for nombre in nombres:  
 print(nombre)  
nombres = Cargar()  
Imprimir(nombres)  
Ordenar(nombres)  
print("Ordenado")  
Imprimir(nombres)

**Problema 3:**

****

def Cargar():  
 contactos = {}  
 continua = "s"  
 while continua == "s":  
 nombre = input("Nombre: ")  
 telefono = int(input("Numero: "))  
 contactos[nombre]=telefono  
 continua = input("continuar?: ")  
 return contactos  
def ModificarContacto(contactos):  
 nombre = input("Ingrese el nombre de contacto a modificar: ")  
 if nombre in contactos:  
 telefono = input("Ingrese el nuevo numero: ")  
 contactos[nombre]=telefono  
 else:  
 print("No existe un contacto con el nombre ingresado")  
def Imprimir(contacto):  
 for i in contacto:  
 print(i,contacto[i])  
  
contacto = Cargar()  
Imprimir(contacto)  
ModificarContacto(contacto)  
Imprimir(contacto)

**Problema 4:**

****

def cargar():  
 empleados = {}  
 continua = "s"  
 while continua == "s":  
 legajo = int(input("Ingrese el nombre de legajo"))  
 nombre = input("nombre: ")  
 profesion = input("profecion: ")  
 sueldo = float(input("sueldo: "))  
 empleados[legajo]=[nombre,profesion,sueldo]  
  
 continua = input("Quiere la carga de otro empleado: ")  
 return empleados  
def Imprimir(empleados):  
 for clave in empleados:  
 print(clave, empleados[clave][0],empleados[clave][1],empleados[clave][2])  
def Modificar(empleados):  
 buscar = int(input("Ingrese el legajo a modificar: "))  
 if buscar in empleados:  
 sueldo = float(input("Ingrese nuevo sueldo: ")  
 empleados[buscar][2]=sueldo  
 else:  
 print("No existe")  
def ImprimirAnalistas(empleados):  
 print("todos los analistas")  
 for i in empleados:  
 if empleados[1]=="Analista":  
 print(i,empleados[i][0],empleados[i][2])

**Porciones de listas, tuplas y de cadenas de caracteres**

El lenguaje Python nos facilita una sintaxis muy sencilla por recuperar un trozo de una lista, tupla o cadena de caracteres.

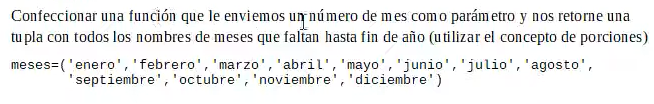
Veremos con una serie de ejemplo como podemos rescatar uno o varios elementos de las estructuras de datos mencionados.

**Problema 1:**

Probar extraer porciones de una lista

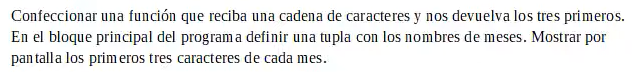
lista1 = [0,1,2,3,4,5,6]  
lista2 = lista1[2:5]  
lista3 = lista1[:5]  
lista4 = lista1[3:]  
  
tupla = 0,1,2,3,4  
print(tupla[2:])  
  
cadena = "administracion"  
print(cadena[5:])

**Problema 2:**



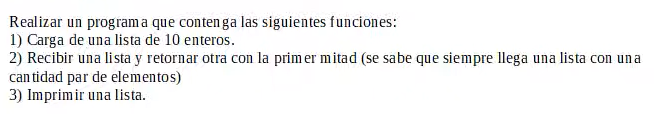
def MesesFaltantes(numeros):  
 meses =("enero","frebero","marzo","abril","mayo","junio","julio","agosto","septiembre","octubre","noviembre","diciembre")  
 return meses[numeros:]  
numeros = int(input("Ingrese el numero de mes: "))  
mesesfalta = MesesFaltantes(numeros)  
print(mesesfalta)

**Problema 3:**

****

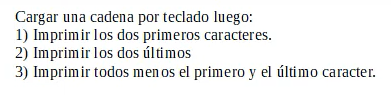
def PrimerosTres(cadena):  
 return cadena[:3]  
def mesesdef():  
 meses =("enero","frebero","marzo","abril","mayo","junio","julio","agosto","septiembre","octubre","noviembre","diciembre")  
 return meses  
def Imprimirmes():  
 for mes in mesesdef():  
 print(PrimerosTres(mes))  
Imprimirmes()

**Problema 4:**

****

def Cargar():  
 lista = []  
 for i in range(10):  
 numero = int(input(f"numero {i+1}: "))  
 lista.append(numero)  
 return lista  
def RetornarMitad(lista):  
 mitad = len(lista) // 2  
 return lista[:mitad]  
def Imprimir(lista):  
 print(lista)  
lista = Cargar()  
lista2 = RetornarMitad(lista)  
Imprimir(lista)  
Imprimir(lista2)

**Problema 5:**

****

cadena = input("Ingrese una cadena de caracteres: ")  
print("Primeros dos caracteres")  
print(cadena[:2])  
print(cadena[len(cadena)-2:])  
print(cadena[1:len(cadena)-1])

**índices negativos en listas, tuplas y cadena de caracteres**

Hemos visto que para acceder a un elemento de una lista, tupla o cadena de caracteres debemos indicar mediante un subíndice que comienza a enumerarse a partir de cero.

También hemos visto el concepto anterior que podemos generar otra lista, tupla o cadena de caracteres indicando una porción con el carácter “:”.

Ahora veremos que podemos utilizar un valor negativo para acceder a un elemento de la estructura de datos.

**Programa: ejercicio175.py**

Lista1=[0,1,2,3,4,5,6]

Print(lista1[-1] #6

Print(lista1[-1] #5

En Python podemos acceder fácilmente al ultimo elemento de la secuencia indicando un subíndice 1:

Print(lista[-1])

luego el antepenultimo se accede con la sintaxis:

print(lista1[-2]) #5

**Problema 1**

****

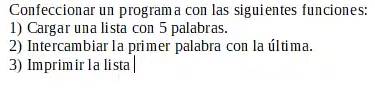
def capicua(cadena):  
 iguales = 0  
 indice = -1  
 for x in range(0,len(cadena)//2):  
 if cadena[x]==cadena[indice]:  
 iguales = iguales + 1  
 indice=indice-1  
 if iguales == (len(cadena)//2):  
 print("es optima")  
 else:  
 print("no es capicua")  
capicua("neuquen")

**Problema 2**

****

palabra = input("Ingrese una palabra: ")  
indice = -1  
for x in range(len(palabra)):  
 print(palabra[indice],end="")  
 indice = indice -1

**Problema 3**

****

def Cargar():  
 palabras = []  
 for x in range(5):  
 pal = input("Ingrese una palabra: ")  
 palabras.append(pal)  
 return palabras  
def Intercambiar(palabras):  
 aux = palabras[0]  
 palabras[0]=palabras[-1]  
 palabras[-1]=aux  
def Imprimir(palabras):  
 print(palabras)  
palabras = Cargar()  
Imprimir(palabras)  
Intercambiar(palabras)  
Imprimir(palabras)