Universidad Nacional de San Juan

Facultad de Ciencias Exactas, Físicas y Naturales

Carrera de Licenciatura en Ciencias de la Computación

Programación Orientada a Objetos

TRABAJO DE INVESTIGACIÓN

Python: Listas y sus Métodos

Elizondo Ramiro y Garello Octavio

Registros: 21590 , 21438

San Juan-Argentina

Mayo del 2022

1. Introducción

A lo largo del trabajo de investigación se abordará como tema principal las Listas, en el lenguaje de programación Python, con sus respectivos métodos de mayor o menor uso.

El contenido consiste en demostrar el uso eficiente y correcto de las estructuras mencionadas anteriormente, siendo un estímulo de interés para aquellos que no poseen conocimiento previo y quieran desenvolverse mejor en el campo.

Para concluir en esta primera parte se mencionarán las palabras clave, y acto seguido se desarrollarán las definiciones con características variadas

Palabras Clave: Listas, Métodos, Python

2. Desarrollo

2.1. Python y Listas

¿Qué es Python?

Es un lenguaje de programación, multiparadigma y multinivel, con soporte en programación orientada a objetos, imperativa y funcional. Con este tipo de lenguaje se pueden crear aplicaciones nativas e híbridas, y cuenta con una sintaxis accesible para las personas con un nivel de 'alfabetización' básico en los lenguajes de programación.

En ella podemos encontrar diferentes tipos de estructuras para poder trabajar o manipular los tipos de datos, uno de los que más destaca son las Listas. En otros lenguajes de la programación se las conocen como arreglos o matrices; y se caracterizan porque los elementos están entre corchetes y separados por una coma.

¿Qué es una Lista?

Cuando se habla de una Lista en Python, hace referencia a una estructura que contiene datos o elementos, ya sea de tipo entero, real u otra característica. Lo especial de ellas es que nos permiten almacenar cualquier tipo de valor en esa misma estructura. Además, son mutables, lo que significa que es posible modificar algún valor de su elemento, una vez creada la lista.

Por otro lado, cuando se habla de modificaciones de una Lista, hace referencia al uso de los diferentes métodos existentes para manipular los elementos que contiene. Algunos de estos son:

- append(): Agregar un ítem al final de la lista
- len(): Devuelve la cantidad de componentes que tiene una lista
- count(): Cuenta el numero de veces que aparece un item
- index(): Devuelve el índice en el que aparece un item
- pop(): Extrae un ítem de la lista y lo borra
- remove(): Borra el primer ítem de la lista cuyo valor concuerde lo que que indiquen
- clear(): Vacía todos los ítems de una lista

A continuación, será dado un una clase Zapatilla (Modulo Secundario) y Main (Modulo Principal) para demostrar estos Métodos:

```
class Zapatillas: #Se crea la clase
    __medida: int
    __precio: float
    __marca: str
    def __init__(self, v1:int, v2:float, v3:str): #Se declara el constructor y los Atributos
        self._medida=v1
         self._precio=v2
         self._marca=v3
    def getAtributos(self):
        return(self.__medida,self.__precio,self.__marca) #Devulve los valores de sus atributos
```

```
from clase import Zapatillas #Se importa la Clase Zapatillas
if __name__=="__main__": #Se crea el Modulo Principal y se verifica que sea este

    lista: list[Zapatillas]=[] #Se crea la lista zapatilla y se inicializa
    zapatilla1=Zapatillas(38,7300.50,"Vans") #Se crea la Primer instancia de la Clase
    zapatilla2=Zapatillas(42,10000.00,"Nike") #Se crea la Segunda instancia de la Clase
    zapatilla3=Zapatillas(40,12000.00,"Adidas") #Se crea la Tercera instancia de la Clase

    metodoAppend(lista,zapatilla1,zapatilla2,zapatilla3) #Se llama a la Función
    metodoLen(lista) #Se llama a la Función
    metodoCount(lista) #Se llama a la Función
    metodoIndex(lista) #Se llama a la Función
    metodoPop(lista) #Se llama a la Función
    metodoRemove(lista) #Se llama a la Función
    metodoClear(lista) #Se llama a la Función
```

2.1.1 Método Append en la Lista:

El método append permite agregar un elemento a la lista, dicho elemento se posicionará como el último tras ser insertado.

```
def metodoAppend (lista:list[Zapatillas],z1,z2,z3): #Se crea la Funcion y recibe los parametros
    lista.append(z1)
    lista.append(z2)
    lista.append(z3)
    print(lista[0].getAtributos()) #Muestra el/ los elementos de la Lista
    print(lista[1].getAtributos())
    print(lista[2].getAtributos())
```

2.1.2 Método Len en la Lista:

El método len() devuelve la longitud de un objeto, ya sea una lista, una cadena, una tupla o un diccionario.

```
def metodoLen(lista:list[Zapatillas]): #Se crea la Función y recibe los parámetros
    print(len(lista)) #Muestra la cantidad de elementos de la lista
```

2.1.3 Método Count en la Lista

El método count() se encarga de devolver un valor respectivo a la cantidad de veces que se encuentra un elemento en la lista.

```
def metodoCount(lista:list[Zapatillas]): #Se crea la Función y recibe los parámetros
    elemento=lista[1] #Se crea una variable que almacene algún elemento
    print(lista.count(elemento)) #Lo busca, encuentra  y muestra la cantidad de veces repetido
```

2.1.4 Método Index en la Lista

El método index() se encarga de recuperar la posición de un elemento, por medio de la búsqueda del mismo.

```
def metodoIndex(lista:list[Zapatillas]): #Se crea la Función y recibe los parámetros
    elemento=lista[2] #Se crea una variable que almacene algún elemento
    print(lista.index(elemento)) #La busca error sino aparece
```

2.1.5 Método Pop en la Lista

La función del método pop() es, por medio de un índice ingresado, eliminar el elemento que se encuentre en esa posición.

```
def metodoPop(lista:list[Zapatillas]): #Se crea la Función y recibe los parámetros
    print(lista.pop(2)) #se le otorga un elemento y lo elimina
    print(lista) #muestra la lista con 1 elemento menos
```

2.1.6 Método Remove en la Lista

El método remove() elimina el primer elemento de la lista, cuyo valor es igual al del argumento pasado.

```
def metodoRemove(lista:list[Zapatillas]): #Se crea la Función y recibe los parámetros
    elemento=lista[1] #Se crea una variable que almacene algún elemento
    lista.remove(elemento) #Se elimina ese elemento de la lista
    print(lista) #Se muestra la lista con 1 elemento menos
```

2.1.7 Método Clear en la Lista

El método clear() de la lista de Python se utiliza para eliminar todos los elementos de la lista .

```
def metodoClear(lista:list[Zapatillas]): #Se crea la Función y recibe los parámetros
    lista.clear()
    print(lista)
```

3. Conclusión:

En este trabajo de investigación se ha visto la definición de lista y algunos de sus métodos más utilizados, teniendo en cuenta el potencial de uso que presenta esta estructura

La evidencia que se presenta anteriormente demuestra que las listas en Python tienen un gran abanico de posibilidades de uso, ya que pueden ayudar al programador a solucionar problemáticas y al tener métodos propios facilita el trabajo con esta estructura. Por su puesto es recomendado mantener una práctica constante de las listas, para sí poder mejorar la habilidad de uso.

4. Bibliografía:

HektorProfe(2022). Métodos de las listas. Recuperado de: https://docs.hektorprofe.net/python/metodos-de-las-colecciones/metodos-de-las-listas/

DevCode (2022). Listas en Python. Recuperado de: https://devcode.la/tutoriales/listas-python/

El Libro de Python (2022). Listas en Python. Recuperado de: https://ellibrodepython.com/listas-en-python