Universidad Internacional de las Américas

Ingeniería del Software

Estructura <list>

Programación II

Daniel Josué Sibaja Chinchilla

Lic. Carlos González Romero

1. **De qué forma podemos agregar, eliminar, modificar y recorrer una estructura tipo list?**

Para agregar, eliminar y modificar existen ciertos métodos ya definidos por el lenguaje de programación, en este caso Python; sin embargo no es imposible replicar estas funcionalidades escribiendo nuestro propio código. Algunos de los métodos más conocidos en este ámbito son:

list.append(x)

Agrega un ítem al final de la lista. Equivale a a[len(a):] = [x].

list.extend(iterable)

Extiende la lista agregándole todos los ítems del iterable. Equivale a a[len(a):] = iterable.

list.insert(i, x)

Inserta un ítem en una posición dada. El primer argumento es el índice del ítem delante del cual se insertará, por lo tanto a.insert(0, x) inserta al principio de la lista y a.insert(len(a), x) equivale a a.append(x).

list.remove(x)

Quita el primer ítem de la lista cuyo valor sea x. Lanza un ValueError si no existe tal ítem.

list.pop([i])

Quita el ítem en la posición dada de la lista y lo retorna. Si no se especifica un índice, a.pop() quita y retorna el último elemento de la lista. (Los corchetes que encierran a i en la firma del método denotan que el parámetro es opcional, no que deberías escribir corchetes en esa posición. Verás esta notación con frecuencia en la Referencia de la Biblioteca de Python.)

list.clear()

Elimina todos los elementos de la lista. Equivalente a del a[:].

list.index(x[, start[, end]])

Retorna el índice basado en cero del primer elemento cuyo valor sea igual a x. Lanza una excepción ValueError si no existe tal elemento.

Los argumentos opcionales start y end son interpretados como la notación de rebanadas y se usan para limitar la búsqueda a un segmento particular de la lista. El índice retornado se calcula de manera relativa al inicio de la secuencia completa en lugar de hacerlo con respecto al argumento start.

list.count(x)

Retorna el número de veces que x aparece en la lista.

list.sort(*, key=None, reverse=False)

Ordena los elementos de la lista

list.reverse()

Invierte los elementos de la lista

list.copy()

Retorna una copia superficial de la lista. Equivalente a a[:].

Para recorrer una lista, comúnmente se hace uso de estructuras iterativas como lo es el ciclo While o For. Siendo este último el más utilizado. Algunas variantes son:

```
fruits = ["apple", "banana", "cherry"]
for x in fruits:
  print(x)
  if x == "banana":
    break
```

```
for x in range(2, 6):
  print(x)
```

2. **De qué forma podemos contar la cantidad de elementos que se encuentran en una colección tipo <list>**

Para contar los elementos que posee una lista, es utilizada con frecuencia la función len(*list*). Ejemplo:

```
list1 = ["God","Belief",10,31,"Human"]

print("The total number of elements in the list: ", len(list1))
```

```
The total number of elements in the list:  5
```

También es posible hacer uso de estructuras iterativas como el ciclo For. Para hacer uso de esta estrategia debemos crear una variable acumulativa que vaya constantemente sumando la cantidad de veces que el ciclo es recorrido. Ejemplo:

```
list2 = ["Hey",20,14,"Look","An Example List"]

def total_elements(list):
    count = 0
    for element in list:
        count += 1
    return count

print("The total number of elements in the list: ", total_elements(list2))
```

```
The total number of elements in the list:  5
```

Ambos métodos encuentran satisfactoriamente el objetivo. No obstante, su uso dependerá de la situación en la que nos encontremos.

3. **Qué entiende por indexación de los elementos en una colección de datos.**

La indexación es el proceso de búsqueda dentro de una lista tomando en consideración el número de posición o también conocido como índice. La indexación nos permite recorrer la estructura tipo <list> por medio del aumento de ese índice. Cada vez que un elemento es agregado en nuestra lista, a este se le es asignado un numero de posición para poder consultarlo, modificarlo, eliminarlo, etc.

La indexación es también es utilizada en el ambiente Web. Este es el proceso a través del cual los buscadores como Google o Yahoo encuentran una página web y la integran entre las listas de los resultados de búsquedas de usuarios. La indexación sirve para que una página web pueda ser registrada por los buscadores y cualquier usuario se tope con ella a la hora de realizar cualquier búsqueda en la web.