UNIVERSIDAD INTERNACIONAL DE LAS AMÉRICAS

ESCUELA DE INGENIERÍA INFORMÁTICA

NOMBRE: Laboratorio 3

CURSO
Programación II

Nombre del estudiante
Yadir Vega Espinoza

Grupo #: 1

PROFESOR
Carlos González Romero

SAN JOSÉ, COSTA RICA

Contenido

¿De qué forma podemos agregar, eliminar, modificar y recorrer una estructura tipo list?

Agregar

Para agregar en una estructura de datos de tipo list, debemos contar con la estructura definida, es decir que exista la lista a la que vamos a agregarle un valor, luego debemos tener el valor que le vamos a ingresar a la lista. Luego una vez que contamos con lo necesario tendremos 3 formas de agregar en una lista.

1- insert(index, element): index es la posición donde queremos agregar el elemento y elementes el elemento a insertar. Ejemplo lista.insert(1, 2), esto inserta en la lista un 2 en la posición 1 y si ya había un valor desplaza los valores hacia la izquierda o derecha respectivamente.
2- lista.append(element): esto inserta un elemento al final de la lista, observe que aquí no requerimos índice. Ejemplo lista.append(1), esto inserta un 1 al final de la lista, y si está vacía pues evidentemente el insertado será el primer y último elemento de la lista.
3- lista.extend(lista2): Otra posibilidad que podemos utilizar para agregar valores a una lista es con extend, incluso nos facilita mucho cuando queremos insertar múltiples valores a la vez. Por ejemplo si tenemos lista1 =[1,2,3] y queremos agregarle los elementos que conforman esta lista [4,5] sería de la siguiente forma lista1.extend([4,5]) y el resultado sería lista1=[1,2,3,4,5].

Eliminar

Similarmente a agregar, tenemos 3 formas de eliminar:

1- remove: lista.remove(elemento), remueve de lista elemento. Por ejemplo: lista = [1,2,3,4,5] aplicando lista.remove(3) el resultado sería lista = [1,2,4,5].
2- pop: lista.pop(elemento), remueve elemento de lista, retorna la posición en la que estaba el elemento eliminado. Por ejemplo lista = [1,2,3,4,5] aplicando lista.pop(2) retorna 1 y la lista resultante es [1,3,4,5].
3- del: del lista[elemento], elimina de lista elemento. Por ejemplo en lista = [1,2,3,4,5] aplicando del lista[2] retorna [1,2,4,5].

Modificar

Para modificar un elemento de una lista según lógica puede haber muchas posibilidades. La planteada por mi persona es buscar la posición en la que está el elemento a modificar, luego

editar por esa posición. Por ejemplo, tenemos lista = [1,2,3,4,5] queremos modificar el 2 por un 100, entonces obtenemos la posición del 2, sería la posición 1, entonces aplicamos lista[1] = 100 y el resultado es lista=[1,100,3,4,5].

Recorrer

Recorrer una lista es un trabajo relativamente sencillo, para más formalismo lo llamaré iterar en una lista. Las formas de iterar a modo general lo podemos realizar mediante ciclos como for, while, do while según el lenguaje en el que estemos programando, pero independiente del lenguaje iteramos por medio de ciclos.

Por ejemplo lista = [1,2,3,4,5]

```
For i in lista:
    Print(i)
```

Retorna:

1

2

3

4

5

En este ejemplo iteramos una lista por medio de un ciclo for en Python.

¿De qué forma podemos contar la cantidad de elementos que se encuentran en una colección tipo <list>?

Para contar los elementos de una lista podemos utilizar funciones predefinidas del lenguaje como lo es en el caso de Python len(lista) devuelve un número entero correspondientes a la cantidad de elementos de lista.

Si queremos hacerlo manualmente podemos definir una variable contador = 0, luego iteramos sobre la lista mediante un ciclo y dentro del ciclo incrementamos el contador en 1 (contador = contador + 1), al final de la iteración tendremos la cantidad de elementos. Ojo, mucho cuidado porque en los métodos vistos contamos apariciones, incluyendo repetidos. Si requerimos sin repetir la lógica cambia y anticipadamente deberíamos eliminar repetidos de la lista y luego efectivamente contar como lo mencioné en lo anterior.

¿Qué entiende por indexación de los elementos en una colección de datos?

Como el término lo dice, indexación, de index, de índice. Esto nos permite en una estructura de datos ver el índice en el que está cada elemento, o según la lógica no solo ver si no mas bien definir un índice según la estructura empleada.

Ejemplo de indexación:

Posición	1	0	2	5	4
Valor	1	2	3	4	5

Cabe destacar que todo lo explicado en este apartado teórico me base únicamente en mi conocimiento actual, por lo que no referencia ningún documento.