OBSERVACIONES DEL LA PRACTICA

Gregorio Salazar 202022085

Valentina Uribe Salcedo 201817485

Para la pregunta 1,2,3,4 se usa:

    """

    Este indice crea un map cuya llave es el autor del libro

    """

    catalog['authors'] = mp.newMap(800,

                                   maptype='CHAINING',

                                   loadfactor=4.0,

                                   comparefunction=compareAuthorsByName)

1. ¿Qué estructura de datos se usa para este índice?

Para el mapa de autores catalog[‘authors’] se utiliza la estructura Separate Chaining. Esto especifíca cuando se define el mapa mp.newMap(maptype=’CHAINING’)

1. ¿Cuántos elementos se espera almacenar inicialmente?

Se espera almacenar inicialmente 800 elementos. Esto se especifica cuando se define el mapa el parámetro mp.newMap(800).

1. ¿Cuál es el factor de carga?

El factor de carga es 4.0. Se especifíca cuando se define el mapa mp.newMap(loadfactor=4.0).

El factor de carga es N/M donde N es la cantidad de elementos máxima y M es la cantidad de espacios en el array.

1. ¿Con cuántos elementos serán necesarios agregar para hacer re-hash de la tabla?

Cuando se excede el factor de carga, se hace re-hash para evitar que las operaciones del mapa no se relenticen tanto. En este caso la cantidad de elementos en el array es el numero primo mas cercano al factor de carga, este se calcula con: NextPrime(800//4.0)=211, por lo tanto, la capacidad máxima de elementos es 211\*4.0=844. Cuando hay 844 elementos se hace re-hash.

Para preguntas 5,6,7 se usa:

   mp.put(catalog['bookIds'], book['goodreads\_book\_id'], book)

1. ¿Qué hace la instrucción “**mp.put(...)”**?

Put(map, key, value)

La función put, agrega una llave y el valor correspondiente a un mapa. En este caso se agrega al mapa catalog[‘bookIds’] la llave book[‘goodreads\_book\_id’] y el valor es el diccionario del libro book, definido anteriormente.

1. ¿Qué papel cumple **“book[‘goodreads\_book\_id’]”** en esa instrucción?

Como se dijo, book[‘goodreads\_book\_id’] es la llave. Esto es un string ya que es el id del libro en goodreads. Como book es un diccionario se debe extraer este string. Este string se usa como llave en el mapa catalog[‘bookIds’] como se explicó anteriormente.

1. ¿Qué papel cumple **“*book”*** en esa instrucción?

Book es un diccionario con los datos del libro. Este es el valor que se le asigna a la llave book[‘goodreads\_book\_id’] en el mapa catalog[‘bookIds’].

Para preguntas 8,9,10 se usa:

def getBooksByYear(catalog, year):

    """

    Retorna los libros publicados en un año

    """

    year = mp.get(catalog['years'], year)

    if year:

        return me.getValue(year)['books']

    return None

1. ¿Qué hace la instrucción **“mp.get(…)”**?

La función recibe dos parámetros get(map, key). El map es el mapa, en este caso catalog['years'], y key es la llave que se quiere consultar, en este caso year. Get retorna la pareja llave valor, es decir, retorna una pareja donde la llave es la llave key, o en este caso year, y el valor es el valor correspondiente a la llave en el mapa.

1. ¿Qué papel cumple **“*year”*** en esa instrucción?

Year, como se explicó es la llave que se está buscando en el mapa. Year será un entero que corresponde al año de interes, y se retorna la pareja llave (year:int) y el valor, que sabemos que es un diccionario con llaves ‘year’ y ‘books’, donde el valor de de la llave ‘year’ es el año y el valor de la llave ‘books’ es una lista con libros de ese año.

1. ¿Qué hace la instrucción **“me.getValue(…)”**?

Como la función get retorna una pareja llave valor, la función getValue toma esta pareja y retorna el valor, es decir en este caso el diccionario. Este luego se evalua en la llave ´books´ para poder utilizar esta lista con los libros