Act 2.1 - Actividad Integral de Conceptos Básicos y Algoritmos Fundamentales

Programación de estructuras de datos y algoritmos fundamentales (Gpo 4)

Alumnos:

Thomas Freund Paternostro //A00831997

Fecha de entrega:

30/09/2021

```
37    //esta vacio el linkedin list? template
38    template <class T>
39    bool LinkedList<T>::isEmpty()
40    {
41      if(size > 0 ){
42        return 0;
43      }else{
44        return 1;
45      }
46
47    }
```

Linea	Comp	Caso
41	1	C1
42	1	C2
43	1	C3
44	1	C4

$$T(n) = C1 * 1 + C2 * 1 + C3 * 1 + C4 * 1$$

$$T(n) = c1 + c2 + c3 + c4 = C = 1$$

$$Complejidad: O\ (1)$$

```
50   //print  template
51   template <class T>
52   void LinkedList<T>::print(){
53     Node<T> *current = head;//asignar tope a la pos actual
54     int i=1; //counter
55     cout<<"El archivo tiene los siguientes elementos:"<<endl;
56     while(current != nullptr){
57
58       cout <<i << ") "<< current->getData() << ", ";
         //imprimir tope y su indice
59       current = current->getNext(); // conseguir el siguiente
60       i++; // sumar al counter de indice
61
62     }
63   }
64
```

Línea	Costo	Repeticiones (peor caso)
53	C1	1
54	C2	1
55	C3	1
56	C4	n
58	C5	n
59	C6	n
60	C7	n

$$T(n) = C1 + C2 + C3(n) + C4(n) + C5(n) + C6(n) + C7(n) + C8$$
$$T(n) = C1 + C2 + C3 + C4n + C5n + C6n + C7n$$
$$T(n) = (C4 + C5 + C6 + C7)n + (C1 + C2 + C3 + C7 + C8)$$
$$a = C4 + C5 + C6 + C7,\ b = C1 + C2 + C3 + C7 + C8$$
$$T(n) = an + b$$

Dado que se evalúa el peor caso y se lleva al límite donde $\lim_{n \to \infty}$, b se vuelve insignificante.

$$T(n) = an$$
$$Complejidad: O(an) = O(n)$$

```
66    //template para leer archivo
67    template <class T>
68    bool LinkedList<T>::read(T data){
69      Node<T> *current = head;
70      T datoActual;
71
72      while(current != nullptr){
73        datoActual = current -> getData();
74            if (data == datoActual){
75              return 1;
76        }
77        current = current -> getNext();
78        }
79        return 0;
80    }
```

Línea	Costo	Repeticiones (peor caso)
68	C1	1
69	C2	1
70	C3	1
72	C4	n
73	C5	n
74	C6	n
75	C7	1
76	C8	n
77	C9	1

$$T(n) = C1 + C2 + C3 + C4(n) + C5(n) + C6(n) + C7 + C8(n) + C9$$
$$T(n) = (C4 + C5 + C6 + C8)n + (C1 + C2 + C3 + C7 + C9)$$
$$a = C4 + C5 + C6 + C8,\ b = C1 + C2 + C3 + C7 + C8$$
$$T(n) = an + b$$

Dado que se evalúa el peor caso y se lleva al límite donde $\lim_{n \to \infty}$,la b se vuelve insignificante.

$$T(n) = an$$
$$Complejidad: O(an) = O(n)$$

```
107   template <class T>
108   void LinkedList<T>::addFirst(T data){
109       head = new Node<T>(data, head); // crea un objeto
      dinaico en el tope
110       size++; //incrementa el tam del linked list
111   }
```

Linea	Comp	Caso
109	1	C1
110	1	C2

$$T(n) = C1 * 1 + C2 * 1$$

$$T(n) = c1 + c2 = C = 1$$

$$Complejidad: O(1)$$

```
113   //create template
114   template <class T>
115   void LinkedList<T>::create(T data){
116
117       if(size == 0){
118         //llama a add first y crea un objeto
119           addFirst(data);
120           return; //salimos de la funcion
121       }
122       Node<T> *aux = head;
123       while(aux->getNext() != nullptr){
124           aux = aux->getNext();
125           size++;
126       }
127       aux->setNext(new Node<T>(data));
128
129       //current->setNext(*aux);
130   }
```

Línea	Costo	Repeticiones (peor caso)
117	C1	1
119	C2	1
120	C3	1
122	C4	1
123	C5	n
124	C6	n
126	C7	n
127	C8	1

$$T(n) = C1 + C2 + C3 + C4 + C5(n) + C6(n) + C7(n) + C8$$

$$T(n) = (C5 + C6 + C7)n + (C1 + C2 + C3 + C8)$$

$$a = C5 + C6 + C7,\ b = C1 + C2 + C3 + C8$$

$$T(n) = an + b$$

Dado que se evalúa el peor caso y se lleva al límite donde $\lim_{n \to \infty}$,la b se vuelve insignificante.

$$T(n) = an$$
$$Complejidad: O\,(an) = O\,(n)$$

```
132   template <class T>
133   void LinkedList<T>::del(T data){
134     Node<T> *previous = nullptr;   //empezamos con la pos nula
135     Node<T> *current = head; // apuntamos al siguiente como
      tope
136     //mientras que siguiente no sea igual al valor buscado y
      no este vacio
137     while ((current->getData() != data) && (current !=
      nullptr))  {
138       //el anterior tope lo pasamos al siguiente
139       previous = current;
140       //pasamos el valor del siguiente nodo
141       current = current->getNext();
142     }
143
144       if(previous == nullptr){
145         head = head->getNext(); //pasamos la posicion del
          siguiente al tope
146         delete current; // borramos el actual valor
147         size--; // reducimos el size del linked list
148       } else {
149         previous->setNext(current->getNext()); // current
          apunta al siguiente valor y los establecemos al valor
          que tendria el proxima. El previo asume esa pos
150         delete current; // borramos el actual valor
151         size--; // reducimos el size del linked list
152       }
153   }
```

Línea	Costo	Repeticiones (peor caso)
134	C1	1
135	C2	1
137	C3	n
141	C4	n
144	C5	n
145	C6	1
146	C7	1
147	C8	1
148	C9	n
149	C10	1
150	C11	1
151	C12	1

$$T(n) = C1 + C2 + C3(n) + C4(n) + C5(n) + C6 + C7 + C8 + C9(n) + C10 + C11 + C12$$
$$T(n) = (C3 + C4 + C5 + C9)n + (C1 + C2 + C6 + C7 + C8 + C11 + C12)$$
$$a = C3 + C4 + C5 + C9,\ b = C1 + C2 + C6 + C7 + C8 + C11 + C12$$
$$T(n) = an + b$$

Dado que se evalúa el peor caso y se lleva al límite donde $\lim_{n \to \infty}$,la b se vuelve insignificante.

$$T(n) = an$$
$$Complejidad: O\,(an) = O\,(n)$$

```
155   template <class T>
156   void LinkedList<T>::update(T data, T updatecurrentValue){
157     T currentValue;
158     Node<T> *current = head;
159     while(current != nullptr){
160       currentValue = current -> getData();
161       if (data == currentValue){
162         current->setData(updatecurrentValue); //le damos el
          nuevo valor
163         return; //salimos de la funcion
164       }
165       current = current -> getNext(); // recorremos la lista
166     }
167
168   }
```

Línea	Costo	Repeticiones (peor caso)
157	C1	1
158	C2	1
159	C3	n
160	C4	n
161	C5	n
162	C6	1
163	C7	n

$$T(n) = C1 + C2 + C3(n) + C4(n) + C5(n) + C6 + C7(n)$$
$$T(n) = (C3 + C4 + C5 + C7)n + (C1 + C2 + C6)$$
$$a = C3 + C4 + C5 + C7,\ b = C1 + C2 + C6$$
$$T(n) = an + b$$

Dado que se evalúa el peor caso y se lleva al límite donde $\lim_{n \to \infty}$,la b se vuelve insignificante.

$$T(n) = an$$
$$Complejidad: O\,(an) = O\,(n)$$

Reflexión:

Se trabajó con la estructura de datos tipo linked list, dónde dónde se lee el primer nodo como nulo y al resto se le da un valor. En este caso es un link list simple dado que solo sabe cuál es el valor siguiente Pero no sabe cuál fue el valor previo. Se crearon templates para regresar un tipo de dato dónde el enfoque fue crear funciones CRUD (create, read, update,delete/destroy) las cuales fueron analizadas bajo su complejidad. Dado que se trabajó con una estructura simple, en el peor de los casos si para encontrar un valor era

requerido recorrer toda la cadena de nodos; el peor caso sería O(n). Qué es lo que se pudo observar en todas las funciones. En esta actividad se pudo trabajar y reforzar cómo utilizar está estructura de datos y cómo se conectan los nodos y celem para desarrollar memoria dinámica y poder trabajar con archivos al mismo tiempo.