



Tecnológico de Monterrey

Documentación de código

Programming of data structures

Inés Alejandro Garcia Mosqueda A00834571
Jesus Fong Ruiz A01254062

Profesores: Luis Ricardo Peña Llamas
Jorge Gonzalez

23/11/2022

Para la entrega se pide implementar una estructura de datos Hash, la cual a partir de una llave que identifique cada elemento pueda ser almacenada en una lista/array, de esta manera conseguir una complejidad de inserción en el peor caso $O(n)$ y en el mejor caso $O(1)$, de igual manera la búsqueda de elementos

Para el almacenamiento de los datos, se creó una clase con los atributos del registro, además de dos listas ligadas que almacenaran los diferentes puertos de la ip que se registre, de igual manera sucede con las razones de negación de acceso

```
10  class Node{
11      public:
12          string month, day, hour, ip;
13          list<string> Date;
14          list<string> Access;
15          list<int> Port;
16
17      Node(){          //Constructor vacio
18          month = "";
19          day = "";
20          hour = "";
21          ip = "";
22      }
23      Node(string &month, string &day, string &hour, string &ip, string &reason){
24          this->month = month;
25          this->day = day;
26          this->hour = hour;
27          string ipNoPort = ip;
28          ipNoPort.erase(ipNoPort.length()-5);
29          this->ip = ipNoPort;
30          string date = month + " " + day + " " + hour;
31          addAttempt(reason, ipPort(ip), date);
32      }
33      void addAttempt(string access, int port, string date){
34          Access.push_back(access);
35          Port.push_back(port);
36          Date.push_back(date);
37      }
38      string getData(){          //Return de fecha en formato "mes día hora:minuto:segundo"
39          string ipNoPort = ip;
40          string res = "IP: " + ipNoPort + "\n";
41          return res;
42      }
43      void getAttempts(){
44          cout << "\tDate:\n";
45          for (auto z : Date)
46          {
47              cout << "\t\t" << z << "\n";
48          }
49          cout << "\tReason:\n";
50          for (auto x : Access)
```

La implementación de la tabla Hash se llevó a cabo con un manejo de colisiones quadratic probing, ya que consideramos en este caso sería de mayor ayuda, debido a que el uso de cadenas nos implicaría un mayor promedio de complejidad en la ejecución del programa, debido a que se tendría que recorrer una lista ligada

```

132 class HashTable_Quadratic{
133     int capacity;
134     Node** table;
135 private:
136     int getPrime(int); //Complejidad O(n)
137     bool checkPrime(int); //Complejidad O(sqrt(n))
138     unsigned long long hashFunction(unsigned long long); //Complejidad O(1)
139     unsigned long long quadratic(unsigned long long, unsigned long long);
140 public:
141     HashTable_Quadratic(){ //Complejidad O(1)
142         this->capacity = 0;
143         table = new Node*[capacity];
144     };
145     HashTable_Quadratic(int n){ //Complejidad O(sqrt(n))
146         int size = getPrime(n);
147         this->capacity = size;
148         table = new Node*[capacity];
149     };
150     ~HashTable_Quadratic()
151     {
152         delete[] table;
153         cout << "\nHashTable_Quadratic deleted. " << endl;
154     };
155     unsigned long long ipIntCode(string);
156     void insertItem(Node*); //O(n)
157     void deleteItem(string); //O(n)
158     void saveHash(); //O(n+m)
159     void fillHash(string); //O(n)
160     void getIpData(string ip);
161 };
162

```

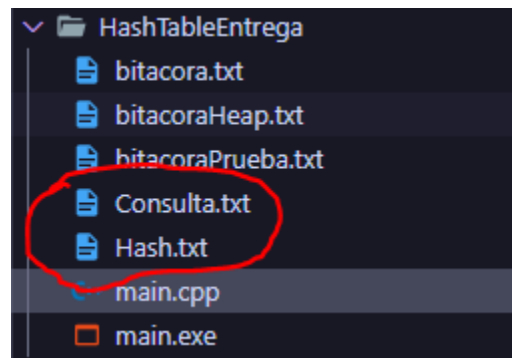
Para los casos de prueba se realizó un menú que ejecuta la carga de datos de una bitácora en la estructura de datos

```

318 int main()
319 {
320     HashTable_Quadratic hash_quadratic;
321     hash_quadratic.fillHash("bitacora.txt");
322     hash_quadratic.saveHash();
323     cout<<"\nSe ha generado archivo Hash.txt\n";
324     bool programCounter = true;
325     string ip, op;
326     while (programCounter)
327     {
328         cout<<"Ingresa la Ip por consultar\nEjemplo:\t423.2.230.77\n---> ";
329         cin>>ip;
330         cout<<endl;
331         cout<<".....\n";
332         hash_quadratic.getIpData(ip);
333         cout<<".....\n";
334         cout<<"\nArchivo Consulta.txt generado con la consulta\n";
335         cout<<"Continuar? [y/n] ";
336         cin>>op;
337         if(op != "y"){
338             programCounter = false;
339         }
340     }
341     return 0;
342 }

```

Como resultado de esto se generan dos archivos txt (Hash.txt, Consulta.txt), Hash.txt contiene el registro de la tabla completa, con la información de la ip, sus accesos y puertos, Consulta.txt Contiene la consulta hecha en el menú



Resultado en Hash.txt

```
bitacora.txt x main.cpp Hash.txt x
Data-structure-and-algorithms-project > HashTableEntrega > Hash.txt
1 table[0] --> IP: 156.14.741.21
2 Date:
3 Aug 18 00:08:26
4 Reason:
5 Illegal user
6 Port:
7 6721
8
9
10 table[1] --> IP: 407.48.975.15
11 Date:
12 Oct 4 13:12:11
13 Reason:
14 Failed password for root
15 Port:
16 5209
17
18
19 table[2] --> IP: 368.92.321.55
20 Date:
21 Oct 22 19:19:12
22 Reason:
23 Failed password for illegal user guest
24 Port:
25 5465
26
27
28 table[3] --> IP: 487.9.420.83
29 Date:
30 Jul 15 00:33:38
31 Reason:
32 Illegal user
33 Port:
34 4403
35
36
37 table[4] --> IP: 833.95.38.3
38 Date:
39 Aug 23 04:12:46
40 Reason:
41 Failed password for illegal user test
42 Port:
43 5325
44
45
46 table[5] --> IP: 64.72.177.72
47 Date:
48 Jul 2 23:54:53
49 Reason:
```

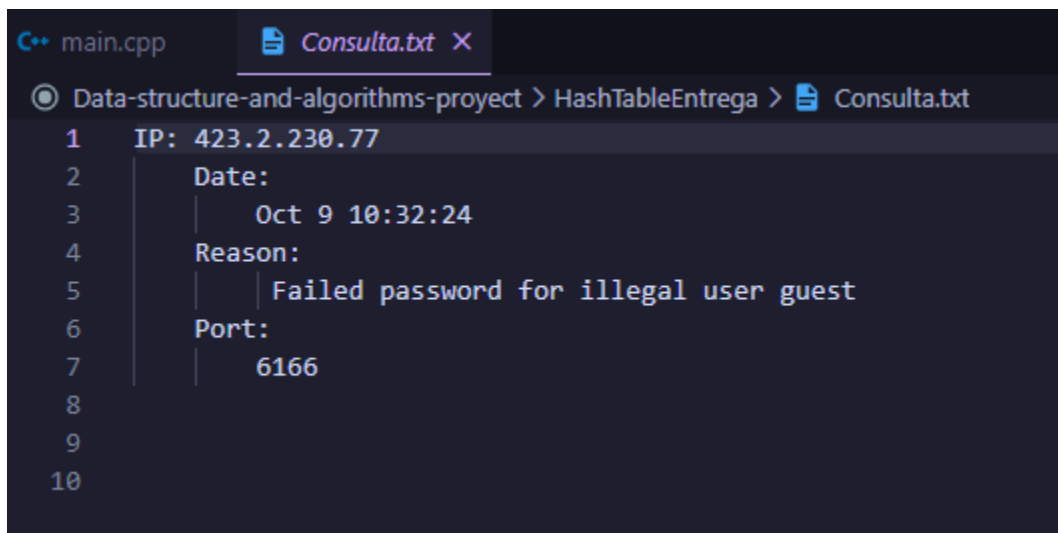
Resultado de la consulta en consola y Consulta.txt

```
Se ha generado archivo Hash.txt
Ingresa la Ip por consultar
Ejemplo:      423.2.230.77
---> 423.2.230.77

.....
IP: 423.2.230.77
  Date:
      Oct 9 10:32:24
  Reason:
      Failed password for illegal user guest
  Port:
      6166

.....

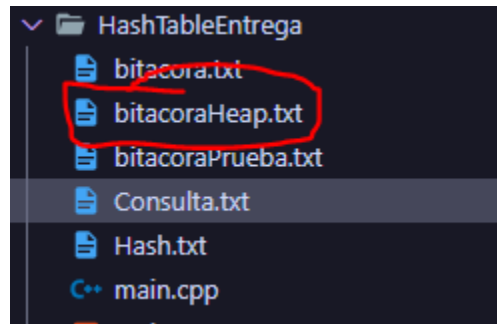
Archivo Consulta.txt generado con la consulta
Continuar? [y/n]
```



The screenshot shows a code editor with two tabs: 'main.cpp' and 'Consulta.txt'. The 'Consulta.txt' tab is active, displaying the output of a program. The text is as follows:

```
1 IP: 423.2.230.77
2   Date:
3     Oct 9 10:32:24
4   Reason:
5     Failed password for illegal user guest
6   Port:
7     6166
8
9
10
```

Caso de prueba con bitacoraHeap.txt



Hash.txt

```
structure-and-algorithms-proyect > HashTableEntrega > Hash.txt
table[159] --> IP: 10.15.170.220
Date:
Oct 30 4:21:12
Jul 13 19:32:41
Jun 15 22:31:08
Jul 6 0:45:05
Aug 27 16:12:53
Jul 26 2:45:58
Sep 30 16:16:39
Jun 29 1:40:56
Oct 11 0:47:03
Oct 7 12:58:57
Sep 15 14:53:00
Jun 11 4:22:49
Jun 20 1:29:55
Aug 5 11:54:01
Jun 1 10:07:04
Oct 8 3:00:57
Oct 19 4:50:14
Oct 6 14:52:36
Oct 28 13:59:18
Sep 9 8:01:59
Reason:
Illegal user
Failed password for admin
Failed password for illegal user test
Failed password for illegal user test
Failed password for root
Failed password for root
Illegal user
Illegal user
Failed password for admin
Failed password for illegal user test
Illegal user
Failed password for admin
Failed password for illegal user guest
Failed password for illegal user guest
Failed password for root
Illegal user
Failed password for illegal user test
Failed password for illegal user test
Failed password for illegal user test
Failed password for illegal user guest
Port:
6918
5055
5942
4311
5920
```

Consulta en Consola y Consulta .txt

```
Se ha generado archivo Hash.txt
Ingresa la Ip por consultar
Ejemplo:      423.2.230.77
---> 10.15.170.221

.....
IP: 10.15.170.221
    Date:
        Aug 17 6:36:05
        Jun 16 20:08:42
        Jul 12 8:24:04
        Jun 29 14:59:18
        Jul 10 12:19:19
        Sep 11 22:58:41
        Sep 16 22:49:34
        Oct 22 9:32:55
        Jul 11 15:23:57
        Jun 5 1:47:26
        Jun 30 8:36:11
        Oct 20 7:04:31
        Jul 14 9:42:28
        Oct 30 15:31:12
        Jul 1 0:56:01
        Jul 20 18:30:18
        Oct 6 12:31:31
        Oct 1 5:49:10
        Jun 4 6:27:23
        Jun 26 8:53:38
        Jul 29 3:02:09
        Jul 19 11:36:41
        Oct 4 1:39:57
        Sep 30 15:13:34
    Reason:
        Failed password for admin
        Failed password for illegal user guest
        Failed password for illegal user test
        Failed password for admin
        Failed password for root
        Failed password for root
        Failed password for illegal user test
        Failed password for illegal user guest
        Failed password for admin
        Failed password for admin
        Illegal user
        Failed password for root
        Failed password for illegal user test
        Failed password for illegal user guest
        Failed password for root
        Illegal user
```

```
main.cpp Consultat.txt X
Data-structure-and-algorithms-project > HashTableEntrega > Consultat.txt
27 //
28 Failed password for admin
29 Failed password for illegal user guest
30 Failed password for illegal user test
31 Failed password for admin
32 Failed password for root
33 Failed password for root
34 Failed password for illegal user test
35 Failed password for illegal user guest
36 Failed password for admin
37 Failed password for admin
38 Illegal user
39 Failed password for root
40 Failed password for illegal user test
41 Failed password for illegal user guest
42 Failed password for root
43 Illegal user
44 Failed password for admin
45 Failed password for admin
46 Failed password for admin
47 Illegal user
48 Failed password for illegal user test
49 Failed password for root
50 Failed password for root
51 Failed password for illegal user guest
52 Port:
53 6673
54 4770
55 6454
56 4129
57 4969
58 6972
59 5145
60 5061
61 5162
62 5077
63 4048
64 6139
65 4227
66 5956
67 4659
68 6225
69 4620
70 6020
71 6005
72 5259
73 5543
74 5943
75 6947
76 6448
```

Inés Alejandro Garcia Mosqueda A00834571
Jesus Fong Ruiz A01254062