Actividad Integral de Conceptos Básicos

Diego Alberto Cisneros Fajardo

Grupo 605

Programación de estructura de datos y algoritmos fundamentales

Tecnológico de Monterrey Campus Guadalajara

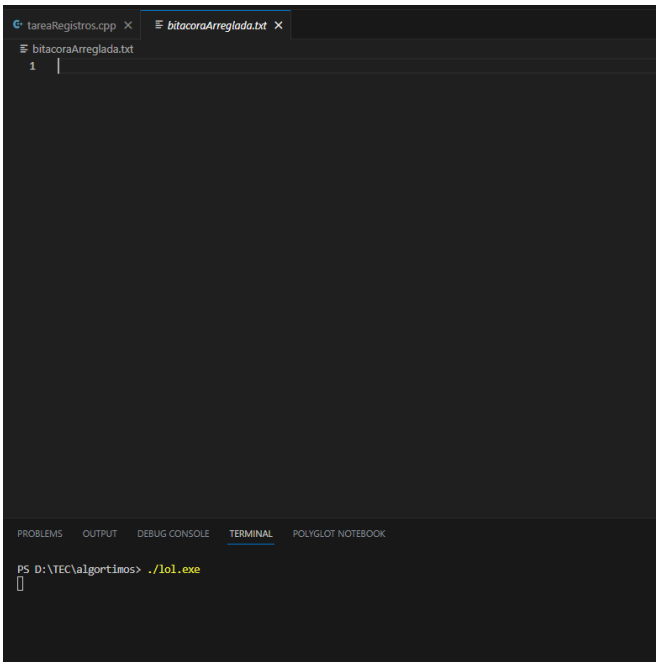
24 de agosto del 2023

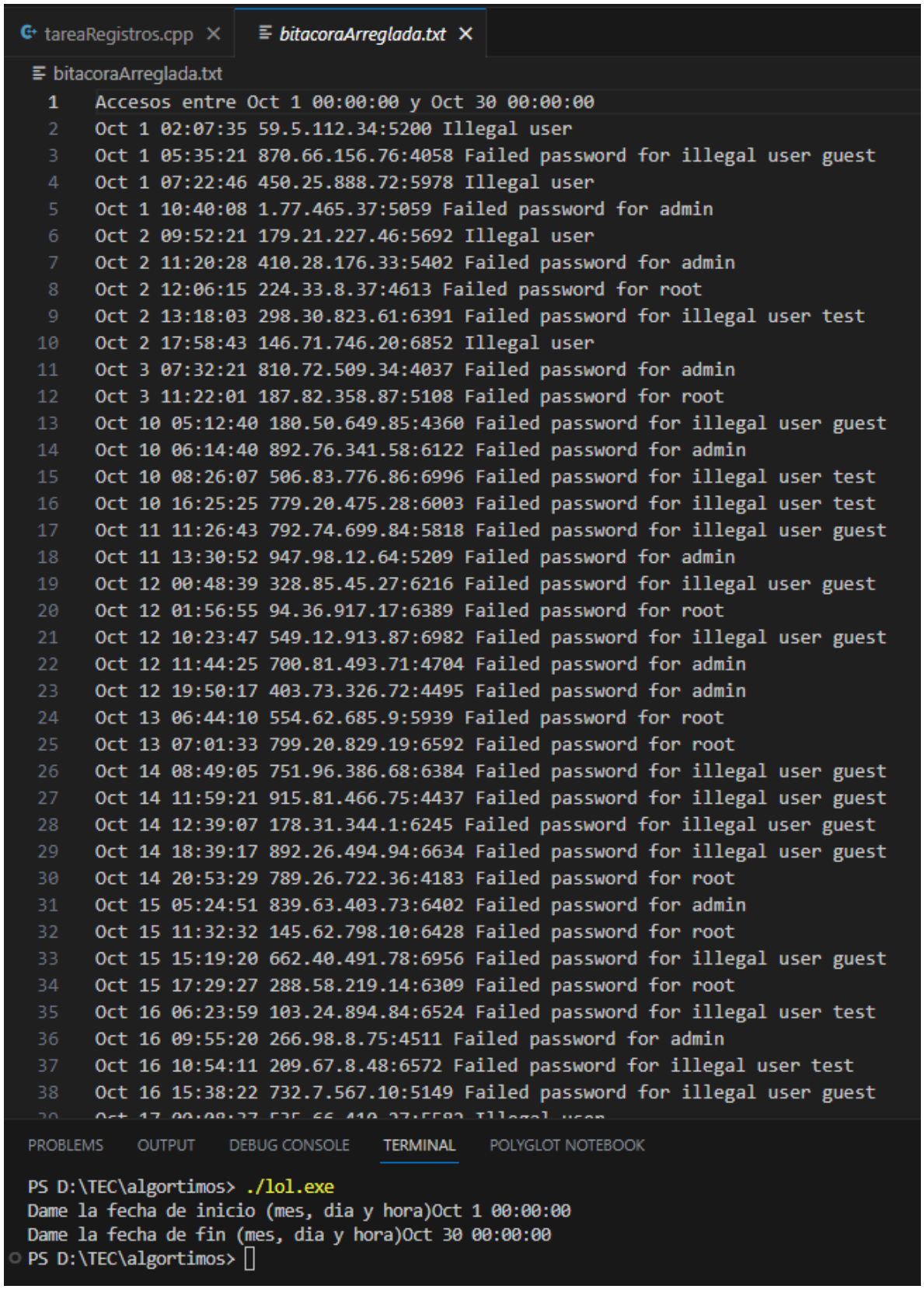
tareaRegistros.cpp × bitacoraArreglada.txt

tareaRegistros.cpp > main()

```cpp
#include <iostream>
#include <fstream>
#include "string"
#include <vector>

using namespace std;


struct linea{
    string mes;
    int dia;
    string hora;
    string ip;
    string razon;
};

bool mayor(linea a1, linea a2){
    if (a1.dia == a2.dia) {
        return a1.hora > a2.hora;
    }
    return a1.dia > a2.dia;
}

void bubbleSort(vector<linea>&lineas){
    int n=lineas.size();

    for (int i = 0; i < n - 1; i++){
        for(int j=0; j<n-i-1;j++){

            if (mayor(lineas[j], lineas[j+1])){
                swap(lineas[j], lineas[j + 1]);
            }
        }
    }
}
//OpO
int main(){
    vector<linea> lineas;
    ifstream text("bitacora.txt");
    ofstream textOut("bitacoraArreglada.txt");
    linea agg;

    while(text>> agg.mes>> agg.dia >> agg.hora >>agg.ip){
    getline(text, agg.razon);
    lineas.push_back(agg);
    bubbleSort(lineas);
    }

```

```
47      }
48
49
50      string inicio, final;
51      cout<<"Dame la fecha de inicio (mes, dia y hora)";
52      getline(cin, inicio);
53
54      cout<<"Dame la fecha de fin (mes, dia y hora)";
55      getline(cin, final);
56
57      textOut<< "Accesos entre "<< inicio<<" y "<<final<<endl;
58
59      for (linea agg : lineas) {
60          string fechaagg = agg.mes + " " + to_string(agg.dia) + " " + agg.hora;
61          if (fechaagg >= inicio && fechaagg <= final) {
62              textOut << agg.mes << " " << agg.dia << " " << agg.hora << " " << agg.ip << agg.razon << "\n";
63          }
64      }
65      text.close();
66      textOut.close();
67
68  }
```




tareaRegistros.cpp
bitacoraArreglada.txt
bitacoraArreglada.txt
Accesos entre Oct 1 00:00:00 y Oct 30 00:00:00
Oct 1 02:07:35 59.5.112.34:5200 Illegal user
Oct 1 05:35:21 870.66.156.76:4058 Failed password for illegal user guest
Oct 1 07:22:46 450.25.888.72:5978 Illegal user
Oct 1 10:40:08 1.77.465.37:5059 Failed password for admin
Oct 2 09:52:21 179.21.227.46:5692 Illegal user
Oct 2 11:20:28 410.28.176.33:5402 Failed password for admin
Oct 2 12:06:15 224.33.8.37:4613 Failed password for root
Oct 2 13:18:03 298.30.823.61:6391 Failed password for illegal user test
Oct 2 17:58:43 146.71.746.20:6852 Illegal user
Oct 3 07:32:21 810.72.509.34:4037 Failed password for admin
Oct 3 11:22:01 187.82.358.87:5108 Failed password for root
Oct 10 05:12:40 180.50.649.85:4360 Failed password for illegal user guest
Oct 10 06:14:40 892.76.341.58:6122 Failed password for admin
Oct 10 08:26:07 506.83.776.86:6996 Failed password for illegal user test
Oct 10 16:25:25 779.20.475.28:6003 Failed password for illegal user test
Oct 11 11:26:43 792.74.699.84:5818 Failed password for illegal user guest
Oct 11 13:30:52 947.98.12.64:5209 Failed password for admin
Oct 12 00:48:39 328.85.45.27:6216 Failed password for illegal user guest
Oct 12 01:56:55 94.36.917.17:6389 Failed password for root
Oct 12 10:23:47 549.12.913.87:6982 Failed password for illegal user guest
Oct 12 11:44:25 700.81.493.71:4704 Failed password for admin
Oct 12 19:50:17 403.73.326.72:4495 Failed password for admin
Oct 13 06:44:10 554.62.685.9:5939 Failed password for root
Oct 13 07:01:33 799.20.829.19:6592 Failed password for root
Oct 14 08:49:05 751.96.386.68:6384 Failed password for illegal user guest
Oct 14 11:59:21 915.81.466.75:4437 Failed password for illegal user guest
Oct 14 12:39:07 178.31.344.1:6245 Failed password for illegal user guest
Oct 14 18:39:17 892.26.494.94:6634 Failed password for illegal user guest
Oct 14 20:53:29 789.26.722.36:4183 Failed password for root
Oct 15 05:24:51 839.63.403.73:6402 Failed password for admin
Oct 15 11:32:32 145.62.798.10:6428 Failed password for root
Oct 15 15:19:20 662.40.491.78:6956 Failed password for illegal user guest
Oct 15 17:29:27 288.58.219.14:6309 Failed password for root
Oct 16 06:23:59 103.24.894.84:6524 Failed password for illegal user test
Oct 16 09:55:20 266.98.8.75:4511 Failed password for admin
Oct 16 10:54:11 209.67.8.48:6572 Failed password for illegal user test
Oct 16 15:38:22 732.7.567.10:5149 Failed password for illegal user guest
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
POLYGLOT NOTEBOOK
PS D:\TEC\algortimos> ./lol.exe
Dame la fecha de inicio (mes, dia y hora)Oct 1 00:00:00
Dame la fecha de fin (mes, dia y hora)Oct 30 00:00:00
PS D:\TEC\algortimos>

Usamos el algoritmo de ordenamiento (bubbleSort)

El Bubble Sort, debido a su complejidad cuadrática $O(n^2)$, se vuelve ineficiente en escenarios con grandes volúmenes de datos. Esto se debe a que realiza numerosas comparaciones y swaps innecesarios, lo que resulta en un alto costo computacional (tiempo excesivo).

En cambio, al optar por algoritmos de ordenamiento más eficientes como el Merge Sort con su complejidad O(n log n), estás aplicando una estrategia "divide & conquer" que se adapta mucho mejor a conjuntos de datos grandes. Divide y conquista implica dividir el problema en partes más pequeñas, resolverlas de manera independiente y luego combinar los resultados. Esto reduce significativamente la cantidad de operaciones necesarias, lo que hace que el Merge Sort sea una elección sólida para tareas de ordenamiento en conjuntos de datos extensos.

Por otro lado, el Insertion Sort, aunque simple y fácil de implementar, también sufre de una complejidad cuadrática. Es eficiente en listas pequeñas o casi ordenadas, pero se vuelve costoso en términos de tiempo a medida que crece la cantidad de datos.

Lo que nos produjo un sentimiento de equivocación al realizar Bubble sort.