

Anexo 10. DATOS

Este programa se encarga de hacer la conversión de los archivos .txt con los casos de prueba a archivos .dat. En este anexo se muestran las líneas de código que ayudan a lograr este objetivo de conversión.

```
1  using namespace std;
2
3  #include <algorithm>
4  #include <iostream>
5  #include <fstream>
6  #include <string>
7  #include <vector>
8  #include <locale>
9  #include <sstream>
10 #include <random>
11 #include <numeric>
12 #include <filesystem>
13
14 //namespace fs = filesystem;
15
16 vector<int> quitarChar(string s) // Recibe un string con valores y espacios.
    Devuelve un vector unicamente con los valores.
17 {
18     locale loc;
19     for (int i = 0; i < s.size(); i++) {
20         if (isdigit(s[i], loc) == false) { // Quita los espacios.
21             s[i] = ' ';
22         }
23     }
24     vector<int> v;
25     int aux;
26     stringstream daniel(s);
27     while (daniel >> aux) {
28         v.push_back(aux); // Los valores que estaban en el string van siendo
            puestos en el vector.
29     }
30     return v; // Retorna el vector.
31 }
32 void mostrar(const vector <int>& v, ofstream& k) {
33
34     if (!v.empty()) {
35         k << v.front();
36         for (int i = 1; i < v.size(); i++) { // Desde el inicio hasta el final
            del vector de entrada, se escriben los valores en el nuevo .txt.
37             k << " " << v[i];
38         }
39     }
40 }
41 void dat(string nombre) {
42     int N, M, merca;
43     vector < vector<int> > mCluster;
44     vector < vector<int> > mDistance;
45     vector <int> vvulne;
46     vector <int> vdem;
47     vector<int> v;
48     vector<int> v1;
49     vector<int> v2;
```

```
50     string name = nombre;
51     string aux, aux1;
52     fstream f;
53     int A;
54
55     string dire = ("C:\\Users\\User\\Escritorio\\PD C++\\Instancias .txt\\" +
56         name + ".txt"); // Obtiene los nombres de cada instancia que estan en
57         un .txt.
58     f.open(dire, fstream::in); // Abre el .txt con el nombre de la instancia.
59     if (f.is_open() == false) {
60         cout << "No se puede abrir el archivo de lectura.\n";
61         cout << dire << endl;
62     }
63
64     f >> N;
65     f >> M;
66
67     for (int i = 0; i < M + 1; i++) {
68         getline(f, aux);
69         v = quitarChar(aux); // Llama la función "quitarChar" con el string
70         obtenido y el vector retornado queda guardado en v.
71         if (aux != "") mCluster.push_back(v);
72     }
73
74     for (int i = 0; i < M; i++) {
75         for (int j = 0; j < mCluster[i].size(); j++) {
76             if ((mCluster[i][j]) == 1) {
77                 A = i + 1;
78                 break;
79             }
80         }
81     }
82
83     for (int i = 0; i < N; i++) {
84         getline(f, aux); // Obtiene como string las líneas que representan
85         las distancias entre nodos.
86         v1 = quitarChar(aux); // Llama la función "quitarChar" con el string
87         obtenido y el vector retornado queda guardado en v1.
88         mDistance.push_back(v1); // Los vectores obtenidos, son puestos en el
89         vector de vectores (mCDistance).
90     }
91
92     getline(f, aux);
93     vdem = quitarChar(aux);
94     getline(f, aux);
95     vvulne = quitarChar(aux);
96     f >> merca;
97
98     f.close();
99
100     ofstream k("C:\\Users\\User\\Escritorio\\PD C++\\Instancias .dat\\" + nombre
101         + ".dat"); // Se crea el nuevo archivo y se imprimen en el los valores
```

```
del archivo anterior

95
96     k << "param N:= " << N << ";" << endl;
97     k << endl;
98
99     k << "param A:= " << A << ";" << endl;
100    k << endl;
101
102    k << "set Cluster:= ";
103    for (int q = 1; q < M + 1; q++) {
104        k << q;
105        if (q != M) k << " ";
106    }
107    k << ";" << endl;
108    k << endl;
109
110    for (int i = 0; i < M; i++) {
111        k << "set Nodo_Cluster[" << i + 1 << "]:= ";
112        mostrar(mCluster[i], k);
113        k << ";" << endl;
114    }
115
116    k << endl;
117    k << "param Merca:= " << merca << ";" << endl;
118    k << endl;
119
120    k << "param C: " << endl;
121
122    for (int a = 1; a < N + 1; a++) {
123        v2.push_back(a);
124    }
125    k << " ";
126    mostrar(v2, k);
127    k << ":= " << endl;
128
129    for (int i = M; i < M + N; i++) {
130        k << i - (M - 1) << " ";
131        mostrar(mDistance[i - M], k);
132        k << endl;
133    }
134
135    k << ";" << endl;
136    k << endl;
137
138    k << "param D:= " << endl;
139    for (int b = 1; b < N + 1; b++) {
140        k << b << " " << vdem[b - 1] << endl;
141    }
142    k << ";" << endl;
143    k << endl;
144
145    k << "param V:= " << endl;
```

```
146     for (int c = 1; c < N + 1; c++) {
147         k << c << " " << vvulne[c - 1] << endl;
148     }
149     k << ";" << endl;
150 }
151
152 #include "Header.h"
153
154 int main() {
155     vector<string> vName;    // Vector de strings que será lleno con los nombres de las instancias
156     string aux;
157     fstream f("C:\\Users\\User\\Escritorio\\PD C++\\nombres2.txt", fstream::in);
158     // Se lee el .txt con los nombres de cada una de las instancias
159     while (f >> aux) {
160         vName.push_back(aux);    // Lee todos los nombres de las instancias y se van poniendo en el vector de strings vName.
161     }
162     for (int i = 0; i < vName.size(); i++) {
163         dat(vName[i]);    // Llama la acción "crearDatos" para cada uno de los nombres presentes en el vector de strings
164     }
165
166     cout << "fin." << endl;    // Muestra "fin." en pantalla cuando haya terminado de ejecutarse el programa.
167     cin.get();
168     return 0;
169 }
170
171
```