```
1   /*
2    * Proyecto: ImplementacionDeUnaListaGenerica
3    * Archivo:  main.cpp
4    * Autor:    J. Miguel Guanira E.//miguel.guanira.
5    *
6    * Created on 23 de abril de 2024, 09:42 AM
7    */
8
9   #include <iostream>
10  #include <iomanip>
11  using namespace std;
12  #include "BibliotecaListaGenerica.h"
13  #include "ListaConEnteros.h"
14  #include "ListaConCadenas.h"
15  #include "ListaConRegistrosVoid.h"
16
17  int main(int argc, char** argv) {
18      void *lista;
19
20  //    crearLista("datos.txt",lista, leeEnteros,intcmp);
21  //    imprimirLista(lista,imprimeEntros);
22  //    eliminarLista(lista,eliminaEntero);
23
24  //    crearLista("personas.txt",lista, leeCad,cadcmp);
25  //    imprimirLista(lista,imprimeCad);
26  //    eliminarLista(lista,eliminaCad);
27  //
28  //    crearLista("personal.csv",lista, leeReg,regNombcmp);
29  //    crearLista("personal.csv",lista, leeReg,regCodcmp);
30      crearLista("personal.csv",lista, leeReg, regSuelcmp);
31      imprimirLista(lista,imprimeReg);
32      eliminarLista(lista,eliminaReg);
33
34      return 0;
35  }
36
37  /*
38   * Proyecto: ImplementacionDeUnaListaGenerica
39   * Archivo:  BibliotecaListaGenerica.h
40   * Autor:    J. Miguel Guanira E. //miguel.guanira.
41   *
42   * Created on 23 de abril de 2024, 09:48 AM
43   */
44
45  #ifndef BIBLIOTECALISTAGENERICA_H
46  #define BIBLIOTECALISTAGENERICA_H
47
48  void crearLista(const char *,void *&lista, void* (*lee)(ifstream&),
49                  int intcmp(const void*, const void*));
50  void imprimirLista(void *lista,void (*imprime)(void*));
51  void eliminarLista(void *lista,void (*elimina)(void*));
52  void insertar(void *dato, void*&lista,int(*cmp)(const void*, const void*));
53  #endif /* BIBLIOTECALISTAGENERICA_H */
54
55  /*
56   * Proyecto: ImplementacionDeUnaListaGenerica
57   * Archivo:  BibliotecaListaGenerica.cpp
58   * Autor:    J. Miguel Gunira E//miguel.guanira.
59   *
60   * Created on 23 de abril de 2024, 09:48 AM
61   */
62
63  #include <iostream>
64  #include <fstream>
65  #include <iomanip>
66  using namespace std;
```

```
67  #include "BibliotecaListaGenerica.h"
68  enum Nodo {DATO,SIG};
69
70  void crearLista(const char *nombArch,void *&lista, void* (*lee)(ifstream&),
71                  int (*cmp)(const void*, const void*)){
72      void *dato;
73      ifstream arch(nombArch,ios::in);
74      if(not arch.is_open()){
75          cout<<"ERROR: No se pudo abrir el archivo "<<nombArch<<endl;
76          exit(1);
77      }
78      lista = nullptr;
79      while(true){
80          dato = lee(arch);
81          if(arch.eof())break;
82          insertar(dato,lista,cmp);
83      }
84  }
85
86  void insertar(void *dato, void*&lista,int(*cmp)(const void*, const void*)){
87      void **p=(void**)lista, **ant=nullptr, **nuevo;
88
89      nuevo = new void*[2];
90      nuevo[DATO] = dato;
91      while(p){
92          if(cmp(p[DATO],dato)>0) break;
93          ant = p;
94          p = (void**)(p[SIG]);
95      }
96      nuevo[SIG] = p;
97      if(ant == nullptr)
98          lista = nuevo;
99      else
100         ant[SIG] = nuevo;
101
102 }
103
104 void imprimirLista(void *lst,void (*imprime)(void*) ){
105     void **lista = (void **)lst;
106     while(lista){
107         imprime(lista[DATO]);
108         lista = (void **)(lista[SIG]);
109     }
110     cout<<endl;
111 }
112
113 void eliminarLista(void *lst,void (*elimina)(void*)){
114     void **lista = (void **)lst, **sale;
115     while(lista){
116         sale = lista;
117         elimina(lista[DATO]);
118         lista = (void **)(lista[SIG]);
119         delete sale;
120     }
121 }
122
123 /*
124  * Proyecto: ImplementacionDeUnaListaGenerica
125  * Archivo:  ListaConEnteros.h
126  * Autor:    J. Miguel Guanira E. //miguel.guanira.
127  *
128  * Created on 24 de abril de 2024, 09:01 AM
129  */
130
131 #ifndef LISTACONENTEROS_H
132 #define LISTACONENTEROS_H
```

```
133
134 void *leeEnteros(ifstream &arch);
135 int intcmp(const void*, const void*);
136 void imprimeEntros(void*);
137 void eliminaEntero(void*);
138
139 #endif /* LISTACONENTEROS_H */
140
141 /*
142  * Proyecto: ImplementacionDeUnaListaGenerica
143  * Archivo:  ListaConEnteros.cpp
144  * Autor:    J. Miguel Gunira E//miguel.guanira.
145  *
146  * Created on 24 de abril de 2024, 09:01 AM
147  */
148
149 #include <iostream>
150 #include <fstream>
151 #include <iomanip>
152 using namespace std;
153
154 void *leeEnteros(ifstream &arch){
155     int d, *dato;
156     arch >> d;
157     if(arch.eof())return nullptr;
158     dato = new int;
159     *dato = d;
160     return dato;
161 }
162
163 int intcmp(const void* a, const void*b){
164     int*ai=(int *)a, *bi=(int*)b;
165     return *ai-*bi;
166 }
167 void imprimeEntros(void*a){
168     int *ai= (int*)a;
169     cout<<setw(6)<<*ai;
170 }
171
172 void eliminaEntero(void*a){
173     int *ai= (int*)a;
174     delete ai;
175 }
176
177 /*
178  * Proyecto: ImplementacionDeUnaListaGenerica
179  * Archivo:  ListaConCadenas.h
180  * Autor:    J. Miguel Guanira E. //miguel.guanira.
181  *
182  * Created on 24 de abril de 2024, 09:14 AM
183  */
184
185 #ifndef LISTACONCADENAS_H
186 #define LISTACONCADENAS_H
187
188 void *leeCad(ifstream &arch);
189 int cadcmp(const void*, const void*);
190 void imprimeCad(void*);
191 void eliminaCad(void*);
192
193 #endif /* LISTACONCADENAS_H */
194
195 /*
196  * Proyecto: ImplementacionDeUnaListaGenerica
197  * Archivo:  ListaConCadenas.cpp
198  * Autor:    J. Miguel Gunira E//miguel.guanira.
```

```
199     *
200     * Created on 24 de abril de 2024, 09:15 AM
201     */
202
203    #include <iostream>
204    #include <fstream>
205    #include <iomanip>
206    using namespace std;
207    #include <cstring>
208
209    void *leeCad(ifstream &arch){
210        char* cad, cadena[60];
211        arch.getline(cadena,60);
212        if(arch.eof())return nullptr;
213        cad = new char[strlen(cadena)+1];
214        strcpy(cad,cadena);
215        return cad;
216    }
217
218    int cadcmp(const void*cad1, const void*cad2){
219        char *cadI1 = (char*)cad1, *cadI2=(char*)cad2;
220        return strcmp(cadI1,cadI2);
221    }
222
223    void imprimeCad(void*cad){
224        char *cadi=(char*)cad;
225        cout<<cadi<<endl;
226    }
227
228    void eliminaCad(void*cad){
229        char *cadi=(char*)cad;
230        delete cadi;
231    }
232
233    /*
234     * Proyecto: ImplementacionDeUnaListaGenerica
235     * Archivo:  ListaConRegistrosVoid.h
236     * Autor:    J. Miguel Guanira E. //miguel.guanira.
237     *
238     * Created on 24 de abril de 2024, 09:27 AM
239     */
240
241    #ifndef LISTACONREGISTROSVOID_H
242    #define LISTACONREGISTROSVOID_H
243
244    void *leeReg(ifstream &arch);
245    int regNombcmp(const void*, const void*);
246    void imprimeReg(void*);
247    void eliminaReg(void*);
248    int regCodcmp(const void*, const void*);
249    int regSuelcmp(const void*, const void*);
250
251    #endif /* LISTACONREGISTROSVOID_H */
252
253    /*
254     * Proyecto: ImplementacionDeUnaListaGenerica
255     * Archivo:  ListaConRegistrosVoid.cpp
256     * Autor:    J. Miguel Gunira E//miguel.guanira.
257     *
258     * Created on 24 de abril de 2024, 09:27 AM
259     */
260
261    #include <iostream>
262    #include <fstream>
263    #include <iomanip>
264    using namespace std;
```

```
265  #include <cstring>
266  #include <cmath>
267  enum Reg {COD,NOM,SUE};
268
269  void *leeReg(ifstream &arch){
270      int *cod, codigo;
271      char *cad, cadena[60];
272      double *sueldo;
273      void**reg;
274      arch>>codigo;
275      if(arch.eof())return nullptr;
276      arch.get();
277      cod = new int;
278      *cod = codigo;
279      arch.getline(cadena,60,',');
280      cad = new char[strlen(cadena)+1];
281      strcpy(cad,cadena);
282      sueldo = new double;
283      arch>>*sueldo;
284      reg=new void*[3];
285      reg[COD]=cod;
286      reg[NOM]=cad;
287      reg[SUE]=sueldo;
288      return reg;
289  }
290
291  int regNombcmp(const void*a, const void*b){
292      void **regI = (void**)a, **regK=(void**)b; //apuntamos al registro
293      char*nombreI = (char*)regI[NOM], *nombreK=(char*)regK[NOM];
294      return strcmp(nombreI,nombreK);
295  }
296
297  int regCodcmp(const void*a, const void*b){
298      void **regI = (void**)a, **regK=(void**)b; //apuntamos al registro
299      int*codigoI = (int*)regI[COD], *codigoK=(int*)regK[COD];
300      return *codigoI-*codigoK;
301  }
302
303  int regSuelcmp(const void*a, const void*b){
304      void **regI = (void**)a, **regK=(void**)b; //apuntamos al registro
305      double*sueldoI = (double*)regI[SUE], *sueldoK=(double*)regK[SUE];
306      return *sueldoK-*sueldoI;
307  }
308
309  void imprimeReg(void*a){
310      void **reg = (void**)a; //apuntamos al registro
311      int* codigo = (int*)reg[COD];
312      char*nombre = (char*)reg[NOM];
313      double*sueldo = (double*)reg[SUE];
314      cout.precision(2);
315      cout<<fixed;
316      cout<<setw(10)<<*codigo<<"  "<<left<<setw(50)<<nombre<<right<<setw(10)
317              <<*sueldo<<endl;
318  }
319
320  void eliminaReg(void*a){
321      void **reg = (void**)a; //apuntamos al registro
322      int* codigo = (int*)reg[COD];
323      char*nombre = (char*)reg[NOM];
324      double*sueldo = (double*)reg[SUE];
325      delete codigo;
326      delete nombre;
327      delete sueldo;
328      delete reg;
329  }
330
```