PUNTEROS A FUNCIONES
IMPLEMENTACIÓN DE UN
QSORT GENÉRICO

```
/*
 * Proyecto: ImplementacionDeUnQsortGenerico
 * Archivo:  main.cpp
 * Autor:    J. Miguel Guanira Erazo (Juan Miguel)
 *
 * Creado el 6 de junio de 2020, 12:21 PM
 */

#include <iostream>
#include <iomanip>
using namespace std;
#include "FuncionesDeOrdenaGenerico.h"
#include "FuncionesDeComparacion.h"
#include "FuncionesDeCadenas.h"
#include "FuncionesDePunterosVoid.h"

int main(int argc, char** argv) {
//    int a[50] = {10,25,7,15,8,33,45,1,19,10,6,16,41}, n= 13;
//    ordenarG(a, 0, n-1,miIntCmp);
//    for(int i=0; i<n; i++)
//        cout<<setw(4)<<a[i];
//    cout<<endl;

//*************************************
//  No se puede trabaja con un double*
//*************************************

//   char **nombres;
//   int numDat;
//
//   leerNombres(nombres,numDat);
//   ordenarG(nombres, 0, numDat-1,miStrCmp);
//   imprimirNombres(nombres,numDat);

    void *personal;
    int np;

    leerDatos(personal,np);
//    ordenarG(personal, 0, np-1,miRegVoidCodigoCmp);
//    ordenarG(personal, 0, np-1,miRegVoidNombreCmp);
    ordenarG(personal, 0, np-1,miRegVoidSueldoCmp);
    imprimirDatos(personal,np);

    return 0;
}

/*
 * Proyecto:  ImplementacionDeUnQsortGenerico
 * Archivo: FuncionesDeOrdenaGenerico.h
 * Autor:   J. Miguel Guanira Erazo (Juan Miguel)
 *
 * Creado el 6 de junio de 2020, 12:25 PM
 */
```

```
54  #ifndef FUNCIONESDEORDENAGENERICO_H
55  #define FUNCIONESDEORDENAGENERICO_H
56
57  void ordenarG(void*, int, int, int(*)(const void*, const void*));
58  void cambiarG(void**, int ,int );
59
60  #endif /* FUNCIONESDEORDENAGENERICO_H */
61
62  /*
63   * Proyecto:  ImplementacionDeUnQsortGenerico
64   * Archivo: FuncionesDeOrdenaGenerico.cpp
65   * Autor:   J. Miguel Guanira Erazo (Juan Miguel)
66   *
67   * Creado el 6 de junio de 2020, 12:26 PM
68   */
69  #include <iostream>
70  #include <iomanip>
71  using namespace std;
72  #include "FuncionesDeOrdenaGenerico.h"
73
74  void ordenarG(void*arr, int izq, int der,
75                int(*cmp)(const void*, const void*)){
76      void **arreglo = (void**)arr;
77      int limite;
78      if(izq >=der) return;
79      cambiarG(arreglo,izq,(izq+der)/2);
80      limite = izq;
81      for(int i=izq+1; i<=der; i++)
82         if(cmp(arreglo[i],arreglo[izq])<0)
83             cambiarG(arreglo,++limite,i);
84      cambiarG(arreglo, izq,limite);
85      ordenarG(arreglo,izq,limite-1,cmp);
86      ordenarG(arreglo,limite+1,der,cmp);
87  }
88
89  void cambiarG(void**arreglo, int i,int k){
90      void *aux;
91      aux = arreglo[i];
92      arreglo[i] = arreglo[k];
93      arreglo[k] = aux;
94  }
95
96  /*
97   * Proyecto:  ImplementacionDeUnQsortGenerico
98   * Archivo: FuncionesDeComparacion.h
99   * Autor:   J. Miguel Guanira Erazo (Juan Miguel)
100  *
101  * Creado el 6 de junio de 2020, 01:06 PM
102  */
103
104
105
106
```

```
107  #ifndef FUNCIONESDECOMPARACION_H
108  #define FUNCIONESDECOMPARACION_H
109  int miIntCmp(const void*, const void*);
110  int miStrCmp(const void*, const void*);
111  int miRegVoidCodigoCmp(const void*, const void*);
112  int miRegVoidNombreCmp(const void*, const void*);
113  int miRegVoidSueldoCmp(const void*, const void*);
114  #endif /* FUNCIONESDECOMPARACION_H */
115
116  /*
117   * Proyecto:  ImplementacionDeUnQsortGenerico
118   * Archivo: FuncionesDeComparacion.cpp
119   * Autor:   J. Miguel Guanira Erazo (Juan Miguel)
120   *
121   * Creado el 6 de junio de 2020, 01:07 PM
122   */
123
124  #include <iostream>
125  #include <iomanip>
126  using namespace std;
127  #include <cstring>
128  enum Registro {CODIGO,NOMBRE,SUELDO};
129
130  int miIntCmp(const void*dI, const void*dK){
131      int datoI = (int)dI, datoK = (int)dK;
132      return datoI - datoK;
133  }
134
135  int miStrCmp(const void*dI, const void*dK){
136      char *datoI = (char*)dI, *datoK = (char*)dK;
137      return strcmp(datoI,datoK);
138  }
139
140  int miRegVoidCodigoCmp(const void*dI, const void*dK){
141      void**regI = (void**)dI,**regK = (void**)dK;
142      int *codigoI = (int*)(regI[CODIGO]),
143          *codigoK = (int*)(regK[CODIGO]);
144      return *codigoI - *codigoK;
145  }
146
147  int miRegVoidNombreCmp(const void*dI, const void*dK){
148      void**regI = (void**)dI,**regK = (void**)dK;
149      char *nombreI = (char*)(regI[NOMBRE]),
150           *nombreK = (char*)(regK[NOMBRE]);
151      return strcmp(nombreI,nombreK);
152  }
153
154  int miRegVoidSueldoCmp(const void*dI, const void*dK){
155      void**regI = (void**)dI,**regK = (void**)dK;
156      double *sueldoI = (double*)(regI[SUELDO]),
157             *sueldoK = (double*)(regK[SUELDO]);
158      return *sueldoI - *sueldoK;
159  }
```

```
160  /*
161   * Proyecto:  UsoDeQsortDeStdlib
162   * Archivo: FuncionesDeCadenas.h
163   * Autor:   J. Miguel Guanira Erazo (Juan Miguel)
164   *
165   * Creado el 4 de junio de 2020, 12:32 PM
166   */
167
168  #ifndef FUNCIONESDECADENAS_H
169  #define FUNCIONESDECADENAS_H
170  #include <fstream>
171  using namespace std;
172
173  void leerNombres(char**&,int &);
174  void imprimirNombres(char **,int );
175  char *leeCad(ifstream &);
176
177  #endif /* FUNCIONESDECADENAS_H */
178
179  /*
180   * Proyecto:  UsoDeQsortDeStdlib
181   * Archivo: FuncionesDeCadenas.cpp
182   * Autor:   J. Miguel Guanira Erazo (Juan Miguel)
183   *
184   * Creado el 4 de junio de 2020, 12:32 PM
185   */
186  #include <iostream>
187  #include <fstream>
188  #include <iomanip>
189  using namespace std;
190  #include "FuncionesDeCadenas.h"
191  #include <cstring>
192
193  void leerNombres(char**&persona,int &np){
194      ifstream arch("personas.txt", ios::in);
195      if(!arch){
196          cout<<"ERROR: no se pudo abrir el archivo personas.txt"<<endl;
197          exit(1);
198      }
199      char *buff[500], *cadena;
200      np=0;
201      while(1){
202          cadena = leeCad(arch);
203          if (cadena == nullptr) break;
204          buff[np]=cadena;
205          np++;
206      }
207      persona = new char*[np];
208      for(int i=0; i<np; i++)
209          persona[i] = buff[i];
210  }
211
212
```

```
213  char *leeCad(ifstream &arch){
214      char cad[100], *cadena;
215      arch.getline(cad,100);
216      if(arch.eof())return nullptr;
217      cadena = new char[strlen(cad)+1];
218      strcpy(cadena,cad);
219      return cadena;
220  }
221
222  void imprimirNombres(char **persona,int np){
223      for(int i=0; i<np; i++)
224          cout<<persona[i]<<endl;
225  }
226
227  /*
228   * Proyecto:  UsoDeQsortDeStdlib
229   * Archivo: FuncionesDePunterosVoid.h
230   * Autor:   J. Miguel Guanira Erazo (Juan Miguel)
231   *
232   * Creado el 4 de junio de 2020, 05:50 PM
233   */
234
235  #ifndef FUNCIONESDEPUNTEROSVOID_H
236  #define FUNCIONESDEPUNTEROSVOID_H
237  #include <fstream>
238  using namespace std;
239  void leerDatos(void *&,int &);
240  void imprimirDatos(void *,int);
241  void *leerReg(ifstream&);
242  void imprimeRegistro(void*);
243  #endif /* FUNCIONESDEPUNTEROSVOID_H */
244
245  /*
246   * Proyecto:  UsoDeQsortDeStdlib
247   * Archivo: FuncionesDePunterosVoid.cpp
248   * Autor:   J. Miguel Guanira Erazo (Juan Miguel)
249   *
250   * Creado el 4 de junio de 2020, 05:50 PM
251   */
252  #include <iostream>
253  #include <fstream>
254  #include <iomanip>
255  using namespace std;
256  #include <cstring>
257  #include "FuncionesDePunterosVoid.h"
258  enum Registro {CODIGO, NOMBRE, SUELDO};
259
260  void leerDatos(void *&persona,int &numDat){
261      ifstream arch("personal.csv");
262      if(!arch){
263          cout<<"ERROR: No se pudo abrir el archivo personal.csv"<<endl;
264          exit(1);
265      }
```

```
266         void *buff[500], **per, *p;
267         numDat = 0;
268         while(1){
269             p = leerReg(arch);
270             if(p == nullptr) break;
271             buff[numDat] = p;
272             numDat++;
273         }
274         per = new void*[numDat];
275         for(int i=0; i<numDat; i++) per[i] = buff[i];
276         persona = per;
277     }
278     void *leerReg(ifstream&arch){
279         void **r;
280         int *codigo, cod;
281         char*nombre, buff[100];
282         double*sueldo;
283         arch>>cod;
284         if(arch.eof()) return nullptr;
285         codigo = new int;
286         *codigo = cod;
287         arch.get(); // Sacamos la coma
288         arch.getline(buff,100,',');
289         nombre = new char[strlen(buff)+1];
290         strcpy(nombre,buff);
291         sueldo = new double;
292         arch>>*sueldo;
293 
294         r = new void*[3];
295         r[CODIGO] = codigo;
296         r[NOMBRE] = nombre;
297         r[SUELDO] = sueldo;
298         return r;
299     }
300 
301     void imprimirDatos(void *persona ,int numDat){
302         void **per = (void **)persona;
303 
304         for(int i=0; i<numDat; i++)
305             imprimeRegistro(per[i]);
306     }
307 
308     void imprimeRegistro(void*reg){
309         void **r = (void**)reg;
310         int *codigo = (int *)r[CODIGO];
311         char *nombre = (char*)r[NOMBRE];
312         double *sueldo = (double*)r[SUELDO];
313 
314         cout.precision(2);
315         cout<<fixed;
316         cout<<left<<setw(10)<<*codigo<<setw(45)<<nombre
317             <<right<<setw(10)<<*sueldo<<endl;
318     }
```