```cpp
/*
 * File:   main.cpp
 * Author: Silvia Vargas
 *
 * Created on 25 de junio de 2025, 22:19
 */

#include <iostream>
#include <iomanip>
#include <fstream>

using namespace std;

#include "Curso.h"
#include "Escala.h"
#include "Alumno.h"
#include "funciones.h"

#define MAX_CURSOS 50
#define MAX_ESCALAS 50
#define MAX_ALUMNOS 100

int main(int argc, char** argv) {
    struct Curso *arrCursos;
    struct Escala *arrEscalas;
    struct Alumno *arrAlumnos;
    int cantCursos,cantEscalas,cantAlumnos,anho=0,ciclo=0;
    char nombreReporte[50];
    arrCursos=new struct Curso[MAX_CURSOS];
    arrEscalas=new struct Escala[MAX_ESCALAS];
    arrAlumnos=new struct Alumno[MAX_ALUMNOS]{};
    cargarCursos("cursos.csv",arrCursos,cantCursos);
    mostrarCursos(arrCursos,cantCursos,"reporteInicialCursos.txt");
    cargarEscalas("escalas.csv",arrEscalas,cantEscalas);
    mostrarEscalas(arrEscalas,cantEscalas,"reporteInicialEscalas.txt");
    cargarAlumnos("alumnos.csv",arrAlumnos,cantAlumnos);
    mostrarReporteAlumnos(arrAlumnos,cantAlumnos,anho,ciclo,"reporteInicialAlumnos.txt");
    //leerAnhoCiclo(anho,ciclo);
    anho=2023;
    ciclo=2;

    procesarMatricula("matricula.txt",anho,ciclo,arrAlumnos,cantAlumnos,arrEscalas,cantEscalas,
            arrCursos,cantCursos);
    ordenarAlumnos(arrAlumnos,cantAlumnos);
    formarNombreArchivo(anho,ciclo,nombreReporte);
    mostrarReporteAlumnos(arrAlumnos,cantAlumnos,anho,ciclo,nombreReporte);
    return 0;
}

/*
 * File:   funciones.cpp
 * Author: Silvia Vargas
 *
 * Created on 25 de junio de 2025, 22:24
 */

#include <iostream>
#include <iomanip>
#include <fstream>

using namespace std;

#include <cstring>
#include <cmath>

#include "Curso.h"
#include "Escala.h"
#include "Alumno.h"
```

```
#include "funciones.h"

#define MAX_CURSOS 8
#define NO_ENCONTRADO -1

void cargarCursos(const char *nombArch,struct Curso *arrCursos,int &cantCursos){
    ifstream archCursos(nombArch,ios::in);
    if (not archCursos.is_open()){
        cout<<"El archivo "<<nombArch<<" no se pudo abrir"<<endl;
        exit(1);
    }
    cantCursos=0;
    while(true){
        archCursos>>arrCursos[cantCursos].codCurso;
        if (archCursos.eof()) break;
        archCursos.get();
        arrCursos[cantCursos].nombre=leeCadenaExactaDelim(archCursos,',');
        archCursos>>arrCursos[cantCursos].creditos;
        cantCursos++;
    }
}

char *leeCadenaExactaDelim(ifstream &arch,char delim){
    char cadena[100],*ptr;
    arch.getline(cadena,100,delim);
    if (arch.eof()) return nullptr;
    ptr=new char[strlen(cadena)+1];
    strcpy(ptr,cadena);
    return ptr;
}

void mostrarCursos(struct Curso *arrCursos,int cantCursos,const char *nombArch){
    ofstream archReporte(nombArch,ios::out);
    if (not archReporte.is_open()){
        cout<<"El archivo "<<nombArch<<" no se pudo abrir"<<endl;
        exit(1);
    }
    archReporte<<"LISTA DE CURSOS"<<endl;
    archReporte<<"CODIGO"<<setw(30)<<"NOMBRE"<<setw(42)<<"CREDITOS"<<endl;
    archReporte<<fixed<<setprecision(2);
    for (int i=0;i<cantCursos;i++)
        archReporte<<setw(8)<<arrCursos[i].codCurso<<setw(3)<<' '<<left
                <<setw(55)<<arrCursos[i].nombre<<right<<setw(10)<<arrCursos[i].creditos<<
                endl;
}

void cargarEscalas(const char *nombArch,struct Escala *arrEscalas,int &cantEscalas){
    ifstream archEscalas(nombArch,ios::in);
    if (not archEscalas.is_open()){
        cout<<"El archivo "<<nombArch<<" no se pudo abrir"<<endl;
        exit(1);
    }
    char car;
    cantEscalas=0;
    while(true){
        archEscalas>>arrEscalas[cantEscalas].anho;
        if (archEscalas.eof()) break;

        archEscalas>>car>>arrEscalas[cantEscalas].ciclo>>car>>arrEscalas[cantEscalas].esc
        ala
                >>car>>arrEscalas[cantEscalas].valorCred;
        cantEscalas++;
    }
}

void mostrarEscalas(struct Escala *arrEscalas,int cantEscalas,const char *nombArch){
    ofstream archReporte(nombArch,ios::out);
    if (not archReporte.is_open()){
```

```
133         cout<<"El archivo "<<nombArch<<" no se pudo abrir"<<endl;
134         exit(1);
135     }
136     archReporte<<"LISTA DE ESCALAS"<<endl;
137 
        archReporte<<"ANHO"<<setw(6)<<"CICLO"<<setw(8)<<"ESCALA"<<setw(12)<<"VALOR.CRED"<<end
        l;
138     archReporte<<fixed<<setprecision(2);
139     for (int i=0;i<cantEscalas;i++)
140         archReporte<<arrEscalas[i].anho<<setw(4)<<arrEscalas[i].ciclo<<setw(7)
141                 <<arrEscalas[i].escala<<setw(12)<<arrEscalas[i].valorCred<<endl;
142 }
143 
144 void cargarAlumnos(const char *nombArch,struct Alumno *arrAlumnos,int &cantAlumnos){
145     ifstream archAlumnos(nombArch,ios::in);
146     if (not archAlumnos.is_open()){
147         cout<<"El archivo "<<nombArch<<" no se pudo abrir"<<endl;
148         exit(1);
149     }
150     char car;
151     cantAlumnos=0;
152     while(true){
153         archAlumnos>>arrAlumnos[cantAlumnos].codAlumno;
154         if (archAlumnos.eof()) break;
155         archAlumnos.get();
156         arrAlumnos[cantAlumnos].nombre=leeCadenaExactaDelim(archAlumnos,',');
157         archAlumnos>>arrAlumnos[cantAlumnos].escAlumno.escala;
158         arrAlumnos[cantAlumnos].cursos=new struct Curso[MAX_CURSOS];
159         cantAlumnos++;
160     }
161 }
162 
163 void mostrarReporteAlumnos(struct Alumno *arrAlumnos,int cantAlumnos,int anho,
164         int ciclo, const char *nombArch){
165     ofstream archReporte(nombArch,ios::out);
166     if (not archReporte.is_open()){
167         cout<<"El archivo "<<nombArch<<" no se pudo abrir"<<endl;
168         exit(1);
169     }
170     double totPago=0,cont=0;
171     archReporte<<fixed<<setprecision(2);
172     archReporte<<setw(60)<<"INSTITUCION EDUCATIVA TP"<<endl;
173     if (anho==0 and ciclo==0)
174         archReporte<<setw(70)<<"DETALLE PRELIMINAR DE ALUMNOS"<<endl;
175     else{
176         archReporte<<setw(70)<<"DETALLE DE PAGOS REALIZADO POR CICLO"<<endl;
177         archReporte<<setw(60)<<"CICLO:"<<setw(5)<<anho<<"-"<<ciclo<<endl;
178     }
179     for (int i=0;i<cantAlumnos;i++){
180         if (ciclo==0 or arrAlumnos[i].cantidadCursos>0){
181             archReporte<<"ALUMNO: "<<setw(10)<<arrAlumnos[i].codAlumno<<" - "
182                 <<left<<setw(40)<<arrAlumnos[i].nombre<<"ESCALA: "
183                 <<arrAlumnos[i].escAlumno.escala<<right<<setw(20)<<"Valor Crédito:"
184                 <<setw(8)<<arrAlumnos[i].escAlumno.valorCred<<endl;
185             archReporte<<setw(20)<<"CURSO MATRICULADO"<<setw(57)<<"CREDITOS"
186                 <<setw(14)<<"MONTO PAGADO"<<endl;
187             imprimirCursos(arrAlumnos[i].cursos,arrAlumnos[i].cantidadCursos,
188                     arrAlumnos[i].escAlumno.valorCred,archReporte);
189             archReporte<<"TOTAL CREDITOS:"<<setw(8)<<arrAlumnos[i].totalCreditos
190                 <<setw(25)<<"TOTAL MONTO PAGADO:
                    "<<setw(8)<<arrAlumnos[i].totalPago<<endl;
191             totPago+=arrAlumnos[i].totalPago;
192             cont++;
193         }
194     }
195     archReporte<<"RESUMEN:"<<endl;
196     archReporte<<"CANTIDAD TOTAL DE ALUMNOS: "<<setw(3)<<cont<<setw(25)
197             <<"MONTO TOTAL PAGADO: "<<setw(12)<<totPago<<endl;
198 }
```

```
199
200 void imprimirCursos(struct Curso *cursos,int cant,double valorCred,ofstream
    &archReporte){
201     for (int i=0;i<cant;i++)
202         archReporte<<setfill('0')<<setw(2)<<i+1<<')'<<setfill('
            ')<<setw(6)<<cursos[i].codCurso
203                 <<" - "<<left<<setw(55)<<cursos[i].nombre<<right<<setw(8)
204                 <<cursos[i].creditos<<setw(12)<<valorCred*cursos[i].creditos<<endl;
205 }
206
207 void leerAnhoCiclo(int &anho,int &ciclo){
208     cout<<"Datos para mostrar el reporte"<<endl;
209     cout<<"Ingrese el anho: ";
210     cin>>anho;
211     cout<<"Ingrese el ciclo: ";
212     cin>>ciclo;
213 }
214
215 void procesarMatricula(const char *nombArch,int anho,int ciclo,struct Alumno *arrAlumnos,
216         int cantAlumnos,
217         struct Escala *arrEscalas,int cantEscalas,struct Curso *arrCursos,int
            cantCursos){
218     ifstream archMatricula(nombArch,ios::in);
219     if (not archMatricula.is_open()){
220         cout<<"El archivo "<<nombArch<<" no se pudo abrir"<<endl;
221         exit(1);
222     }
223     int anioMat,cicloMat,alumno,curso,idAlumno,idEscala,idCurso,cantCur;
224     char c,escAlumno;
225     double valorCred;
226     while(true){
227         archMatricula>>anioMat;
228         if (archMatricula.eof()) break;
229         archMatricula>>c>>cicloMat>>alumno;
230         if (anioMat==anho and ciclo==cicloMat){
231             //buscar la escala del alumno
232             idAlumno=buscarAlumno(arrAlumnos,cantAlumnos,alumno);
233             if (idAlumno!=NO_ENCONTRADO){
234                 escAlumno=arrAlumnos[idAlumno].escAlumno.escala;
235                 idEscala=buscarEscala(arrEscalas,cantEscalas,escAlumno,anho,ciclo);
236                 arrAlumnos[idAlumno].escAlumno.anho=anho;
237                 arrAlumnos[idAlumno].escAlumno.ciclo=ciclo;
238                 if (idEscala==NO_ENCONTRADO)
239                     valorCred=0;
240                 else
241                     valorCred=arrEscalas[idEscala].valorCred;
242                 arrAlumnos[idAlumno].escAlumno.valorCred=valorCred;
243                 while(true){
244                     archMatricula>>curso;
245                     idCurso=buscarCurso(arrCursos,cantCursos,curso);
246                     if (idCurso!=NO_ENCONTRADO){
247                         cantCur=arrAlumnos[idAlumno].cantidadCursos;
248                         arrAlumnos[idAlumno].cursos[cantCur].codCurso=curso;
249
                            arrAlumnos[idAlumno].cursos[cantCur].creditos=arrCursos[idCurso].
                            creditos;
250                         arrAlumnos[idAlumno].cursos[cantCur].nombre=new
                            char[strlen(arrCursos[idCurso].nombre)+1];
251
                            strcpy(arrAlumnos[idAlumno].cursos[cantCur].nombre,arrCursos[idCu
                            rso].nombre);
252                         arrAlumnos[idAlumno].cantidadCursos++;
253                         arrAlumnos[idAlumno].totalCreditos+=arrCursos[idCurso].creditos;
254
                            arrAlumnos[idAlumno].totalPago+=(arrCursos[idCurso].creditos*valo
                            rCred);
255                     }
256                     if (archMatricula.get()=='\n') break;
257                 }
```

```
258             }
259             else
260                 while(archMatricula.get()!='\n');
261         }
262         else
263             while(archMatricula.get()!='\n');
264     }
265 }
266
267 int buscarAlumno(struct Alumno *arrAlumnos,int cantAlumnos,int alumno){
268     for (int i=0;i<cantAlumnos;i++)
269         if (arrAlumnos[i].codAlumno==alumno) return i;
270     return NO_ENCONTRADO;
271 }
272
273 int buscarEscala(struct Escala *arrEscalas,int cantEscalas,char escAlumno,int anho,int
    ciclo){
274     for (int i=0;i<cantEscalas;i++)
275         if (arrEscalas[i].anho==anho and arrEscalas[i].ciclo==ciclo and
          arrEscalas[i].escala==escAlumno) return i;
276     return NO_ENCONTRADO;
277 }
278
279 int buscarCurso(struct Curso *arrCursos,int cantCursos,int curso){
280     for (int i=0;i<cantCursos;i++)
281         if (arrCursos[i].codCurso==curso) return i;
282     return NO_ENCONTRADO;
283 }
284
285 void ordenarAlumnos(struct Alumno *arrAlumnos,int cantAlumnos){
286     for (int i=0;i<cantAlumnos-1;i++)
287         for (int j=i+1;j<cantAlumnos;j++)
288             if (arrAlumnos[i].escAlumno.escala>arrAlumnos[j].escAlumno.escala or
289                 (arrAlumnos[i].escAlumno.escala==arrAlumnos[j].escAlumno.escala
290                 and arrAlumnos[i].totalCreditos<arrAlumnos[j].totalCreditos))
291                 cambiarAlumno(arrAlumnos[i],arrAlumnos[j]);
292     for (int i=0;i<cantAlumnos;i++)
293         ordenarCurso(arrAlumnos[i].cursos,arrAlumnos[i].cantidadCursos);
294 }
295
296 void cambiarAlumno(struct Alumno &alumnoI,struct Alumno &alumnoJ){
297     struct Alumno aux;
298     aux=alumnoI;
299     alumnoI=alumnoJ;
300     alumnoJ=aux;
301 }
302
303 void ordenarCurso(struct Curso *arrCursos,int cant){
304     for (int i=0;i<cant-1;i++)
305         for (int j=i+1;j<cant;j++)
306             if (strcmp(arrCursos[i].nombre,arrCursos[j].nombre)>0)
307                 cambiarCurso(arrCursos[i],arrCursos[j]);
308 }
309
310 void cambiarCurso(struct Curso &cursoI,struct Curso &cursoJ){
311     struct Curso aux;
312     aux=cursoI;
313     cursoI=cursoJ;
314     cursoJ=aux;
315 }
316
317 void formarNombreArchivo(int anho,int ciclo,char *nombreReporte){
318     int longNomb;
319     strcpy(nombreReporte,"ReporteAlumnos_");
320     anadirNumero(anho,nombreReporte);
321     longNomb=strlen(nombreReporte);
322     nombreReporte[longNomb]='_';
323     nombreReporte[longNomb+1]=0;
324     anadirNumero(ciclo,nombreReporte);
```

```
325         strcat(nombreReporte,".txt");
326     }
327 
328     void anadirNumero(int num,char *nombreReporte){
329         int cantDig=0,numCopia=num,dig,longNomb;
330         char digCar;
331         while(numCopia>0){
332             numCopia=numCopia/10;
333             cantDig++;
334         }
335         while(num>0){
336             dig=num/(int)pow(10,cantDig-1);
337             num=num%(int)pow(10,cantDig-1);
338             cantDig--;
339             longNomb=strlen(nombreReporte);
340             digCar='0'+dig;
341             nombreReporte[longNomb]=digCar;
342             nombreReporte[longNomb+1]=0;
343         }
344     }
345 
346     /*
347      * File:   Curso.h
348      * Author: Silvia Vargas
349      *
350      * Created on 25 de junio de 2025, 22:20
351      */
352 
353     #ifndef CURSO_H
354     #define CURSO_H
355 
356     struct Curso{
357         int codCurso;
358         char *nombre;
359         double creditos;
360     };
361 
362     #endif /* CURSO_H */
363 
364     /*
365      * File:   Escala.h
366      * Author: Silvia Vargas
367      *
368      * Created on 25 de junio de 2025, 22:21
369      */
370 
371     #ifndef ESCALA_H
372     #define ESCALA_H
373 
374     struct Escala{
375         int anho;
376         int ciclo;
377         char escala;
378         double valorCred;
379     };
380 
381     #endif /* ESCALA_H */
382 
383     /*
384      * File:   Alumno.h
385      * Author: Silvia Vargas
386      *
387      * Created on 25 de junio de 2025, 22:22
388      */
389 
390     #ifndef ALUMNO_H
391     #define ALUMNO_H
392 
393     struct Alumno{
```

```
    int codAlumno;
    char *nombre;
    struct Escala escAlumno;
    struct Curso *cursos;
    int cantidadCursos;
    double totalCreditos;
    double totalPago;
};

#endif /* ALUMNO_H */

/*
 * File:   funciones.h
 * Author: Silvia Vargas
 *
 * Created on 25 de junio de 2025, 22:24
 */

#ifndef FUNCIONES_H
#define FUNCIONES_H

void cargarCursos(const char *,struct Curso *,int &);
char *leeCadenaExactaDelim(ifstream &,char );
void mostrarCursos(struct Curso *,int ,const char *);
void cargarEscalas(const char *,struct Escala *,int &);
void mostrarEscalas(struct Escala *,int ,const char *);
void cargarAlumnos(const char *,struct Alumno *,int &);
void mostrarReporteAlumnos(struct Alumno *,int ,int ,int , const char *);
void imprimirCursos(struct Curso *,int ,double ,ofstream &);
void leerAnhoCiclo(int &,int &);
void procesarMatricula(const char *,int ,int ,struct Alumno *,int ,
        struct Escala *,int ,struct Curso *,int );
int buscarAlumno(struct Alumno *,int ,int );
int buscarEscala(struct Escala *,int ,char ,int ,int );
int buscarCurso(struct Curso *,int ,int );
void ordenarAlumnos(struct Alumno *,int );
void cambiarAlumno(struct Alumno &,struct Alumno &);
void ordenarCurso(struct Curso *,int );
void cambiarCurso(struct Curso &,struct Curso &);
void anadirNumero(int ,char *);
void formarNombreArchivo(int ,int ,char *);
#endif /* FUNCIONES_H */
```