UNIVERSIDAD NEBRIJA

Escuela Politécnica Superior

IIN151 – Estructuras de datos y algoritmos

Caso práctico

Nombre: _____

Una asociación de muchos a muchos permite relacionar dos conjuntos de datos entre sí. Por ejemplo, la relación entre los alumnos y los cursos que ofrece una universidad es una relación de muchos a muchos. Un alumno puede estar matriculado en uno o más cursos y cada curso puede tener uno o más alumnos matriculados.

La estructura de datos más simple para representar una relación de muchos a muchos es una tabla de matrículas, donde las filas pueden representar a los alumnos y las columnas a los cursos. En la siguiente tabla se puede observar que el alumno A002 está matriculado en los cursos C001 y C102 y que el curso C103 tiene matriculados a los alumnos A001, A003, A005 y A006.

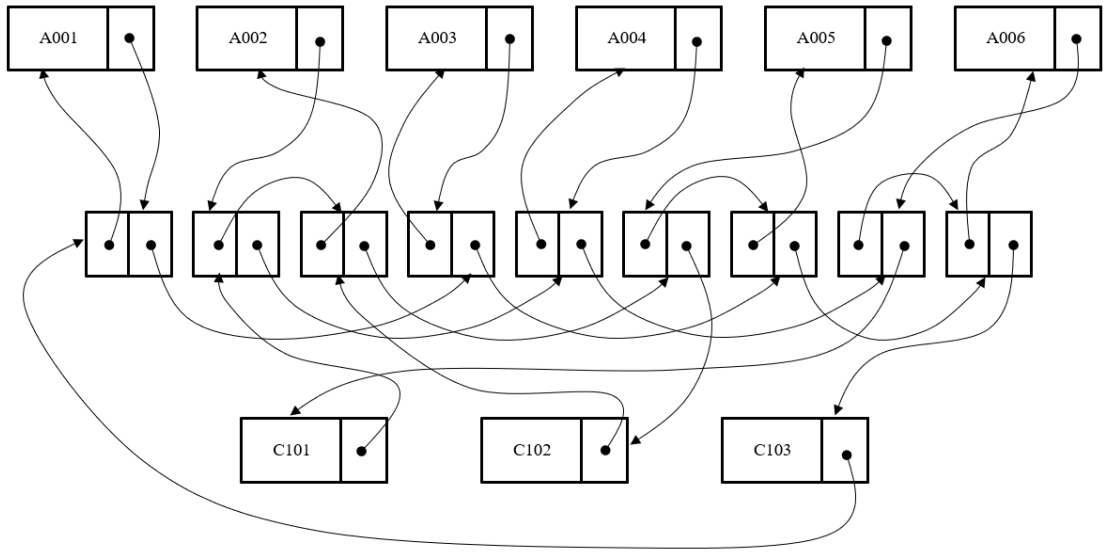
	C101	C102	C103
A001			✓
A002	✓	✓	
A003			✓
A004	✓		
A005		✓	✓
A006	✓		✓

Una estructura de tipo tabla permite establecer fácilmente la relación entre un alumno y un curso. Si el alumno registrado en la fila `i` está matriculado en el curso correspondiente a la columna `j`, basta con hacer `Matricula[i][j]` igual a verdadero. Esta estructura es simple, pero ocupa mucho espacio y la mayoría de sus registros no tienen valor.

Para una universidad con 10.000 alumnos que ofrece 750 cursos, la tabla de matrículas tendría 7.500.000 registros. Si cada alumno se matriculara de media en 8 cursos por año, se registrarían aproximadamente 80.000 matrículas, lo que equivaldría a un 1,1% de la capacidad total de la tabla.

Para evitar utilizar una tabla como estructura de almacenamiento, se propone una estructura de listas múltiples para registrar las matrículas de los alumnos. Esto significa gestionar las relaciones del conjunto `A` de alumnos con el conjunto `C` de cursos mediante un conjunto `M` de matrículas. Esta estructura permite saber cuáles son los cursos en los que está matriculado un alumno y los alumnos matriculados en cada curso. De la tabla de la página anterior, se ve que el alumno A002 está matriculado en los cursos C101 y C102. Esto se representa con el conjunto $A002_{Cursos}$ = {C101, C102}. De forma similar, el curso C102 tiene matriculados los alumnos A002 y A005. Esto se representa con el conjunto $C102_{Alumnos}$ = {A002, A005}.

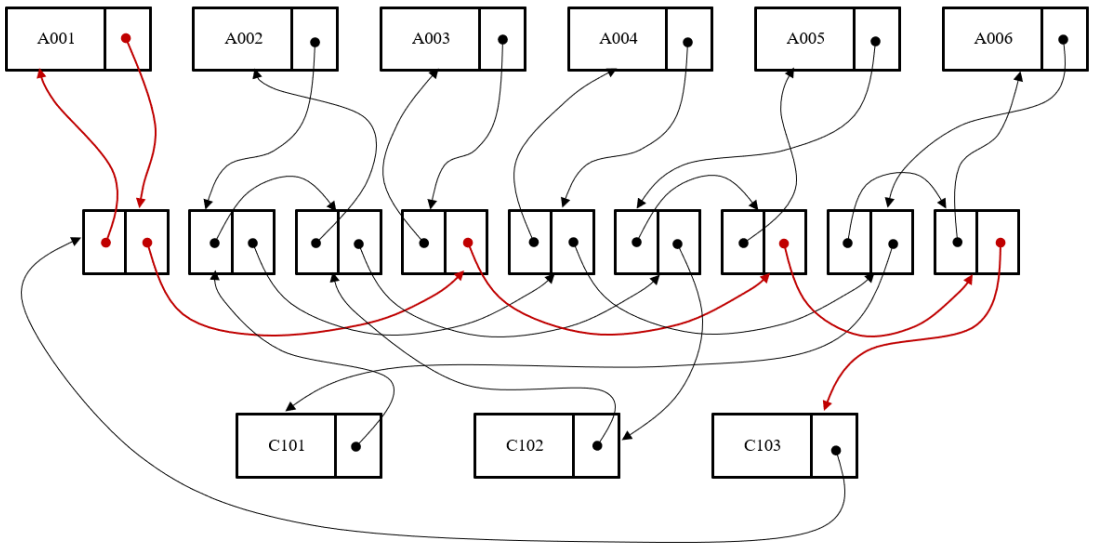
Una estructura de listas múltiples es una colección de nodos que pueden pertenecer a una o más listas a la vez.



Para gestionar las matrículas de una universidad se utilizan tres tipos de nodos: alumno, curso y matrícula. Los nodos de tipo alumno almacenan los datos del alumno y un puntero a la lista de cursos en los que está matriculado. Los nodos de tipo curso almacenan los datos del curso y un puntero a la lista de alumnos matriculados. Por último,

los nodos de tipo matrícula relacionan un alumno y un curso. Estos nodos matrícula utilizan dos punteros, uno para la lista de alumnos y otro para la lista de cursos.

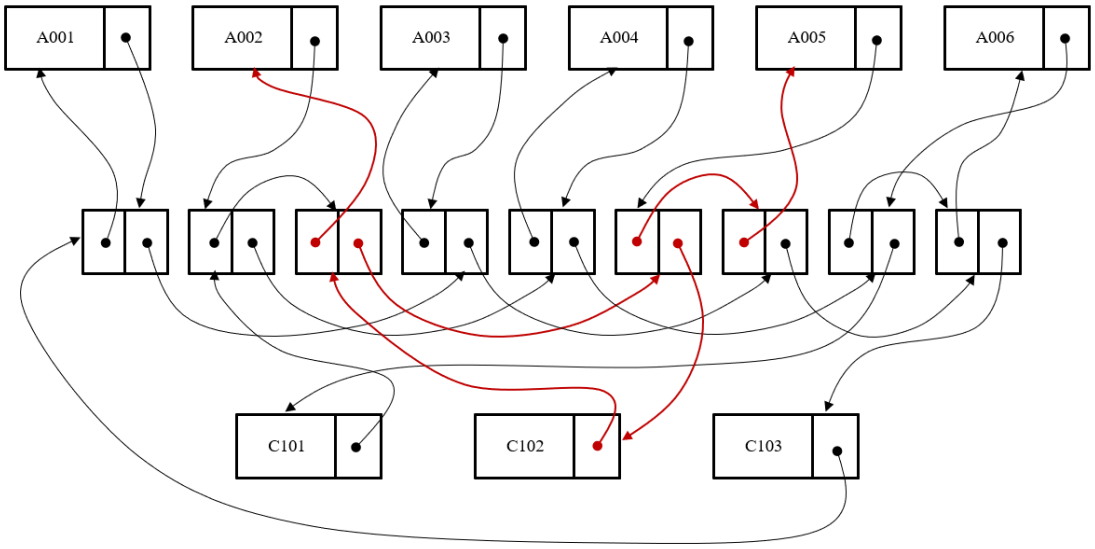
La siguiente figura muestra de color rojo los punteros que definen la lista de cursos en los que está matriculado el alumno A001.



El alumno A001 está matriculado en el curso C103. El puntero del nodo A001 lleva al primer nodo matrícula. Este nodo es el primero de una lista circular que, eventualmente, lleva de nuevo al alumno A001. Dado que el alumno A001 solo está matriculado en el curso C103, el puntero de la izquierda del nodo matricula lleva nuevamente al nodo alumno, cerrando la lista circular.

Para saber cuál es el curso en el que está matriculado el alumno A001, basta con seguir el puntero de la derecha del nodo matrícula. Este puntero lleva a los nodos matrícula de las posiciones cuarta, séptima y novena. El puntero de la derecha de este último nodo matrícula lleva al nodo del curso C103. De este modo, el conjunto de cursos en los que está matriculado el alumno es $A001_{Cursos} = \{C103\}$.

La siguiente figura muestra de color rojo los punteros que definen la lista de alumnos matriculados en el curso C102.



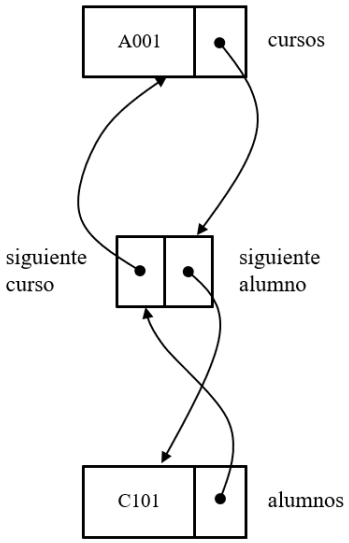
El curso C102 tiene matriculados a los alumnos A002 y A005. El puntero del nodo C102 lleva al tercer nodo matrícula. Este nodo es el primero de una lista circular a la que también pertenece el sexto nodo matrícula. Estos dos nodos definen la lista de alumnos matriculados en el curso C102. El puntero derecho del sexto nodo matrícula lleva nuevamente al nodo curso, cerrando la lista circular.

Para saber quiénes son los alumnos matriculados en el curso C102, basta con seguir el puntero de la izquierda de los nodos matrícula de la lista circular de C102 hasta llegar a un nodo alumno. El puntero izquierdo del tercer nodo lleva directamente al alumno A002 y el puntero izquierdo del sexto nodo lleva hasta el alumno A005 pasando por otro nodo matrícula. De este modo, el conjunto de alumnos matriculados en el curso es $C102_{Alumnos}$ = {A002, A005}.

Los nodos alumno, además de los datos del alumno, tienen un puntero `cursos` que apunta al nodo matrícula del primer curso en el que está matriculado el alumno. Los nodos curso, además de los datos del curso, tienen un puntero `alumnos` que apunta al nodo matrícula del primer alumno matriculado en el curso. Por último, los nodos matrícula tienen dos punteros, el puntero `siguienteAlumno` define la lista circular de los alumnos matriculados en un curso, donde el último nodo de esta lista lleva nuevamente al nodo curso. De manera análoga, el puntero `siguienteCurso` define la lista circular de los cursos en los que está matriculado un alumno. En este caso, el último nodo matrícula de esta lista lleva nuevamente al nodo alumno.

La siguiente figura representa los cursos en los que está matriculado el alumno A001.

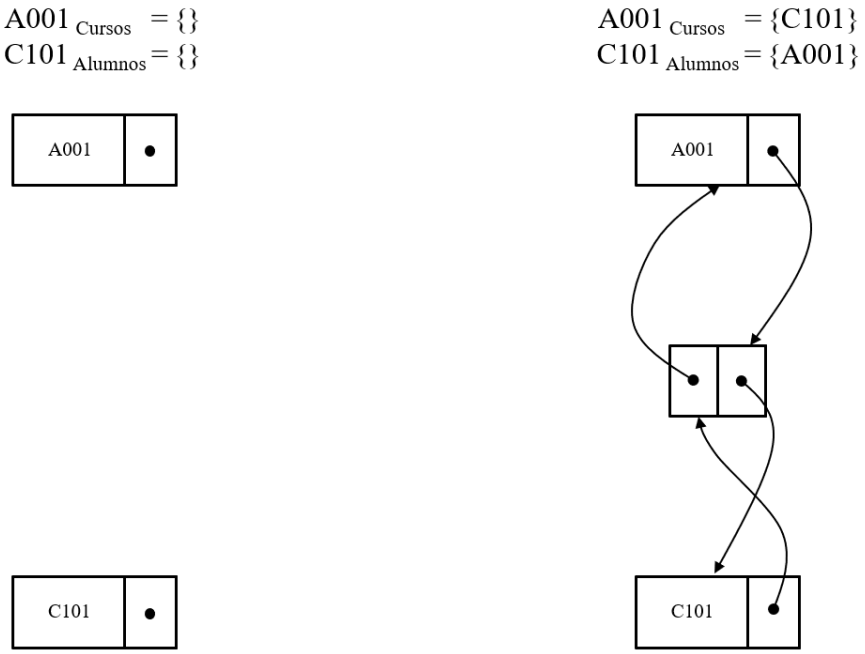
$$A001_{\text{Cursos}} = \{C101\}$$
$$C101_{\text{Alumnos}} = \{A001\}$$



¿Cómo se realiza una matrícula en la estructura de listas múltiples? A continuación se describen los casos de alta de matrícula.

- El nodo alumno no tiene cursos matriculados y el nodo curso no tiene alumnos. Por ejemplo, el alumno A001 se matricula en C101.

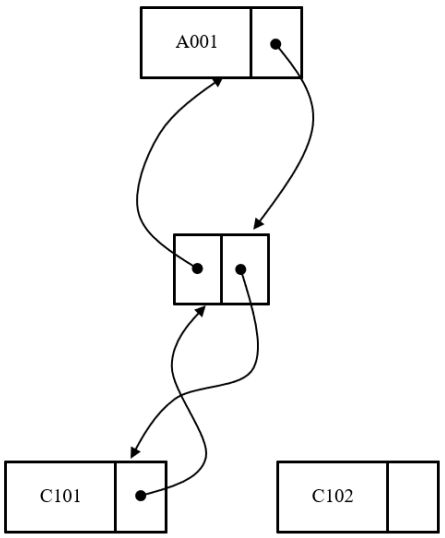
  El puntero `cursos` del nodo A001 y el puntero `alumnos` del nodo C101 deben apuntar al nuevo nodo matrícula. El puntero `siguienteCurso` del nuevo nodo matrícula debe apuntar al nodo A001 y el puntero `siguienteAlumno` al nodo C101.



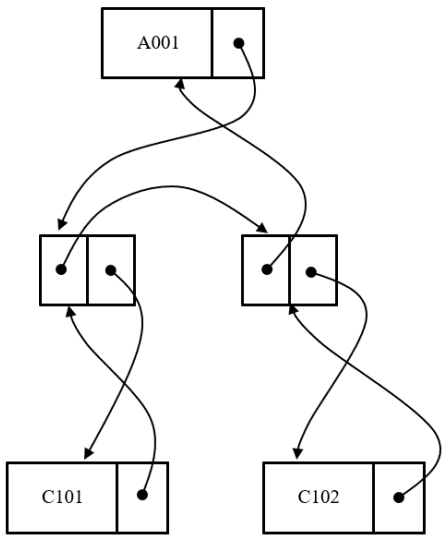
- El nodo alumno tiene al menos un curso matriculado y el nodo curso no tiene alumnos. Por ejemplo, el alumno A001 se matricula en el curso C102.

  El puntero `alumnos` del nodo C102 debe apuntar al nuevo nodo matrícula. El puntero `siguienteCurso` del nuevo nodo matrícula debe apuntar al nodo A001 y el puntero `siguienteAlumno` al nodo C102. El puntero `siguienteCurso` del nodo matrícula que apunta a A001 deba apuntar al nuevo nodo matrícula.

$A001_{Cursos} = \{C101\}$
$C101_{Alumnos} = \{A001\}$
$C102_{Alumnos} = \{\}$



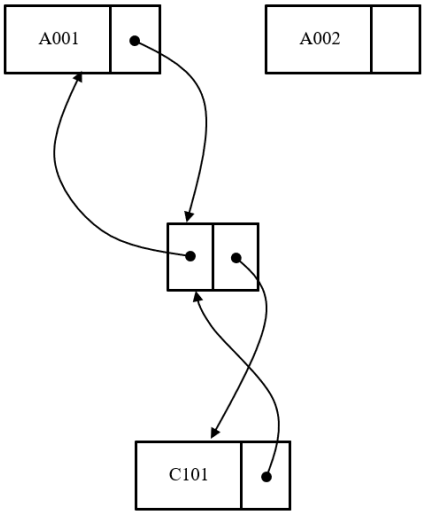
$A001_{Cursos} = \{C101, C102\}$
$C101_{Alumnos} = \{A001\}$
$C102_{Alumnos} = \{A001\}$



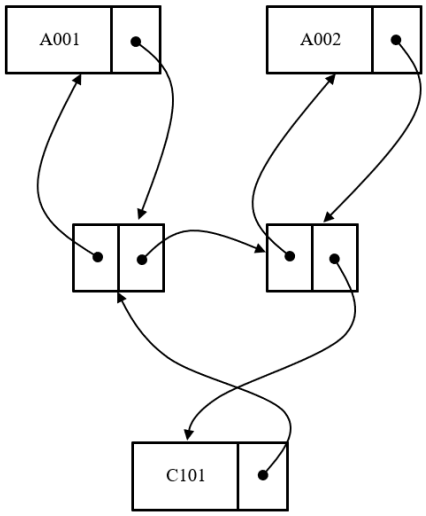
- El nodo alumno no tiene cursos matriculados y el nodo curso tiene al menos un alumno matriculado. Por ejemplo, el alumno A002 se matricula en el curso C101.

  El puntero `cursos` del nodo A002 debe apuntar al nuevo nodo matrícula. El puntero `siguienteCurso` del nuevo nodo matrícula debe apuntar al nodo A002 y el puntero `siguienteAlumno` al nodo C101. El puntero `siguienteAlumno` del nodo matrícula que apunta a C101 debe apuntar al nuevo nodo matrícula.

$A001_{Cursos} = \{C101\}$
$A002_{Cursos} = \{\}$
$C101_{Alumnos} = \{A001\}$



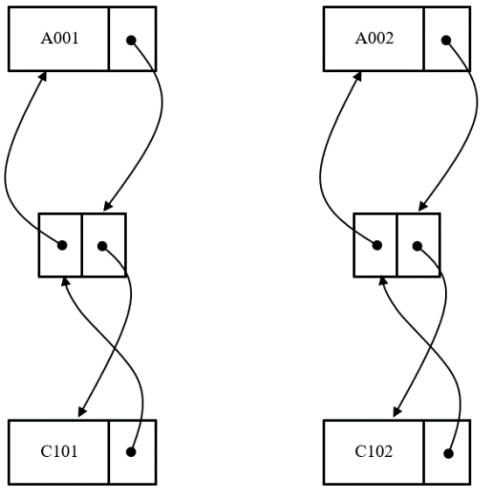
$A001_{Cursos} = \{C101\}$
$A002_{Cursos} = \{C101\}$
$C101_{Alumnos} = \{A001, A002\}$



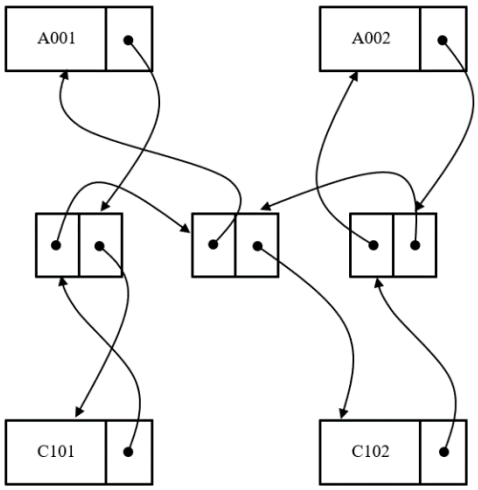
- El nodo alumno tiene al menos un curso matriculado y el nodo curso también tiene alumnos matriculado. Por ejemplo, el alumno A001 se matricula en el curso C102.

  El puntero `siguienteCurso` del nuevo nodo matrícula debe apuntar al nodo A001 y el puntero `siguienteAlumno` al nodo C102. El puntero `siguienteCurso` del nodo matrícula que apunta a A001 debe apuntar al nuevo nodo matrícula. El puntero `siguienteAlumno` del nodo matrícula que apunta a C102 debe apuntar al nuevo nodo matrícula.

$A001_{Cursos} = \{C101\}$
$A002_{Cursos} = \{C102\}$
$C101_{Alumnos} = \{A001\}$
$C102_{Alumnos} = \{A002\}$



$A001_{Cursos} = \{C101, C102\}$
$A002_{Cursos} = \{C102\}$
$C101_{Alumnos} = \{A001\}$
$C102_{Alumnos} = \{A002, A001\}$



La interfaz del TAD Listas Múltiples.

```
altaMatricula(alumno, curso) Matricula de un alumno en un curso

  Precondición:  Los punteros alumno y curso no deben ser nulos
  Postcondición: Se crea un nodo de tipo matricula que relaciona
                 los nodos alumno y curso

bajaMatricula(alumno, curso) Elimina la matrícula del alumno

  Precondición:  Los punteros alumno y curso no deben ser nulos
  Postcondición: Se elimina el nodo de tipo matricula que relaciona
                 los nodos alumno y curso

modificaNota(alumno, curso, nota) Modifica la nota del alumno en un
curso

  Precondición:  El alumno debe estar matriculado en el curso
  Postcondición: Se modifica la nota del alumno en el curso

promedioNotasAlumno(alumno) Calcula el promedio de las notas del
alumno

  Precondición:  Ninguna
  Postcondición: Ninguna

promedioNotasCurso(curso) Calcula el promedio de las notas del curso

  Precondición:  Ninguna
  Postcondición: Ninguna

imprimeAlumnos(curso) Muestra los cursos en los que está matriculado
el alumno

  Precondición:  Ninguna
  Postcondición: Ninguna

imprimeCursos(alumno) Muestra los alumnos matriculados en el curso

  Precondición:  Ninguna
  Postcondición: Ninguna
```

Implemente el TAD Listas Múltiples utilizando el siguiente código.

```
#include "stdafx.h"
#include "iostream"

class Nodo {
   public:

      char tipo;
};

class Alumno : public Nodo {
   public:

      const char* matricula;
      const char* nombre;
      Nodo* cursos;

      Alumno(const char* matricula, const char* nombre,
             Nodo* cursos);
      void imprime();
};

Alumno::Alumno(const char* matricula, const char* nombre,
               Nodo* cursos) {
   this->tipo = 'A';
   this->matricula = matricula;
   this->nombre = nombre;
   this->cursos = cursos;
}

void Alumno::imprime() {
   std::cout << this->matricula << " " << this->nombre << "\n";
}

class Curso : public Nodo {
   public:

      const char* codigo;
      const char* nombre;
      Nodo* alumnos;

      Curso(const char* codigo, const char* nombre, Nodo* alumnos);
      void imprime();
};
```

```
Curso::Curso(const char* codigo, const char* nombre,
             Nodo* alumnos) {
   this->tipo = 'C';
   this->codigo = codigo;
   this->nombre = nombre;
   this->alumnos = alumnos;
}

void Curso::imprime() {
   std::cout << this->codigo << " " << this->nombre << "\n";
}

class Matricula : public Nodo {
   public:

      int nota;
      Nodo* siguienteCurso;
      Nodo* siguienteAlumno;

      Matricula();
};

Matricula::Matricula() {
   this->tipo = 'M';
   this->nota = 0;
   this->siguienteCurso = NULL;
   this->siguienteAlumno = NULL;
}
```

```
void altaMatricula(Alumno *&alumno, Curso *&curso) {

   // inserta un nodo de tipo Matricula para definir una relación
   // entre el alumno y el curso

}

void bajaMatricula(Alumno *&alumno, Curso *&curso) {

   // elimina el nodo de tipo matrícula que relaciona al alumno
   // con el curso

}

void modificaNota(Alumno *alumno, Curso *curso, int nota) {

   // modifica la nota del alumno en un curso

}

double promedioNotasAlumno(Alumno *alumno) {

   // calcula el promedio de las notas del alumno

}

double promedioNotasCurso(Curso *curso) {

   // calcula el promedio de las notas de los alumnos del curso

}
```

```
void imprimeAlumnos(Curso *curso) {

  // muestra la lista de alumnos matriculados en el curso

  // se define un puntero p de tipo Nodo que se inicializa con el
  // puntero curso->alumnos, el primer alumno matriculado en el
  // curso

  // el puntero p recorre la lista de punteros siguienteAlumno
  // de los nodos de tipo Matricula hasta que p apunta nuevamente
  // al nodo del curso, es decir, cuando se ha llegado al final
  // de la lista circular

  // para cada nodo Matricula apuntado por p, se define un puntero
  // q que recorre la lista de punteros siguienteCurso de los nodos
  // Matrícula hasta que se llega a un nodo de tipo Alumno y se
  // muestran los datos del nombre del alumno

  // cuando se llega a un un nodo Alumno, el puntero p avanza al
  // siguienteAlumno y, cuando p apunta nuevamente al nodo Curso, se
  // ha recorrido toda la lista circular de alumnos del curso

  // como las listas utilizan nodos de distinto tipo, es necesario
  // hacer casting para los punteros de tipo Nodo, por ejemplo si
  // p es de tipo puntero a Nodo y se desea usarlo como puntero a un
  // nodo de tipo Alumno, se puede hacer:

  //   static_cast<Alumno*>(p)->matricula = "A001";
  //   static_cast<Alumno*>(p)->nombre = "Juan";

  // si no se hace el casting, no se puede acceder a los datos
  // propios de la subclase Alumno

}

void imprimeCursos(Alumno *alumno) {

  // muestra la lista de cursos en los que está matriculado el alumno

}
```

El programa de prueba del TAD Listas Múltiples.

```
int main() {

   Alumno *a1 = new Alumno("A001", "Juan", NULL);
   Alumno *a2 = new Alumno("A002", "Alberto", NULL);
   Alumno *a3 = new Alumno("A003", "Luis", NULL);
   Alumno *a4 = new Alumno("A004", "Maria", NULL);
   Alumno *a5 = new Alumno("A005", "Ana", NULL);
   Alumno *a6 = new Alumno("A006", "Marta", NULL);

   Curso *c1 = new Curso("I001", "Estructuras", NULL);
   Curso *c2 = new Curso("I002", "Matematicas", NULL);
   Curso *c3 = new Curso("I003", "Programacion", NULL);
   Curso *c4 = new Curso("I004", "Negocios web", NULL);

   altaMatricula(a1, c1);
   altaMatricula(a1, c2);
   altaMatricula(a2, c1);
   altaMatricula(a2, c3);
   altaMatricula(a1, c3);
   altaMatricula(a3, c2);
   altaMatricula(a1, c4);
   altaMatricula(a4, c2);
   altaMatricula(a5, c3);
   altaMatricula(a6, c3);
   altaMatricula(a4, c4);
   altaMatricula(a3, c4);

   imprimeAlumnos(c1);
   imprimeAlumnos(c2);
   imprimeAlumnos(c3);
   imprimeAlumnos(c4);

   imprimeCursos(a1);
   imprimeCursos(a2);
   imprimeCursos(a3);
   imprimeCursos(a4);
   imprimeCursos(a5);
   imprimeCursos(a6);

   modificaNota(a1, c1, 10);
   modificaNota(a1, c2, 8);
   modificaNota(a2, c1, 7);
   modificaNota(a2, c3, 6);
   modificaNota(a1, c3, 10);
   modificaNota(a3, c2, 8);
   modificaNota(a1, c4, 8);
   modificaNota(a4, c2, 10);
   modificaNota(a5, c3, 7);
```

```
    modificaNota(a6, c3, 9);
    modificaNota(a4, c4, 5);
    modificaNota(a3, c4, 6);

    imprimeAlumnos(c1);
    imprimeAlumnos(c2);
    imprimeAlumnos(c3);
    imprimeAlumnos(c4);

    imprimeCursos(a1);
    imprimeCursos(a2);
    imprimeCursos(a3);
    imprimeCursos(a4);
    imprimeCursos(a5);
    imprimeCursos(a6);

    return 0;

}
```

La salida por la consola:

```
Alumnos matriculados en I001 Estructuras

A001 Juan      0
A002 Alberto   0

Promedio: 0

Alumnos matriculados en I002 Matematicas

A001 Juan      0
A003 Luis      0
A004 Maria     0

Promedio: 0

Alumnos matriculados en I003 Programacion

A002 Alberto   0
A001 Juan      0
A005 Ana       0
A006 Marta     0

Promedio: 0
```

```
Alumnos matriculados en I004 Negocios web

A001 Juan      0
A004 Maria     0
A003 Luis      0

Promedio: 0

Cursos en los que esta matriculado el alumno A001 Juan

I001 Estructuras     0
I002 Matematicas     0
I003 Programacion    0
I004 Negocios web    0

Promedio: 0

Cursos en los que esta matriculado el alumno A002 Alberto

I001 Estructuras     0
I003 Programacion    0

Promedio: 0

Cursos en los que esta matriculado el alumno A003 Luis

I002 Matematicas     0
I004 Negocios web    0

Promedio: 0

Cursos en los que esta matriculado el alumno A004 Maria

I002 Matematicas     0
I004 Negocios web    0

Promedio: 0

Cursos en los que esta matriculado el alumno A005 Ana

I003 Programacion    0

Promedio: 0

Cursos en los que esta matriculado el alumno A006 Marta

I003 Programacion    0

Promedio: 0
```

```
Alumnos matriculados en I001 Estructuras

A001 Juan      10
A002 Alberto   7

Promedio: 8.5

Alumnos matriculados en I002 Matematicas

A001 Juan      8
A003 Luis      8
A004 Maria     10

Promedio: 8.66667

Alumnos matriculados en I003 Programacion

A002 Alberto   6
A001 Juan      10
A005 Ana       7
A006 Marta     9

Promedio: 8

Alumnos matriculados en I004 Negocios web

A001 Juan      8
A004 Maria     5
A003 Luis      6

Promedio: 6.33333

Cursos en los que esta matriculado el alumno A001 Juan

I001 Estructuras     10
I002 Matematicas     8
I003 Programacion    10
I004 Negocios web    8

Promedio: 9

Cursos en los que esta matriculado el alumno A002 Alberto

I001 Estructuras     7
I003 Programacion    6

Promedio: 6.5
```

```
Cursos en los que esta matriculado el alumno A003 Luis

I002 Matematicas     8
I004 Negocios web    6

Promedio: 7

Cursos en los que esta matriculado el alumno A004 Maria

I002 Matematicas     10
I004 Negocios web    5

Promedio: 7.5

Cursos en los que esta matriculado el alumno A005 Ana

I003 Programacion    7

Promedio: 7

Cursos en los que esta matriculado el alumno A006 Marta

I003 Programacion    9

Promedio: 9
```