

Capítulo V Multilistas.

1. Introducción.
2. Multilistas.
3. Representación Gráfica de una Multilista.
4. Estructura de Datos de una Multilista.
5. Operaciones básicas de una Multilista.
6. Implementación de la Multilista de asignaturas y estudiantes.
7. Otras Multilistas
8. Orientación para el laboratorio.

Introducción.

En este capítulo usted aprenderá a trabajar con diferentes Multilistas y aplicará todo lo estudiado en el capítulo anterior.

Multilistas

Las multilistas, como su nombre indica, son múltiples listas. Esas listas pueden ser de diferentes tipos ya estudiadas; listas simplemente enlazadas, doblemente enlazadas ó circulares.

Representación Gráfica de una Multilista.

Como se ha estudiado en el capítulo anterior, casi siempre se debe trabajar con conjuntos de elementos y no solo con uno, pero muchas veces estos elementos están divididos también por categorías; en ese caso ya una lista no es lo más factible a utilizar; sino que se debe utilizar otra estructura, como por ejemplo las multilistas.

Veamos los siguientes ejemplos:

- 1.- Suponga que es necesario en un colegio, llevar el control por curso o asignatura, de los estudiantes de ese curso o asignatura, esta situación se puede representar gráficamente como la figura 1.

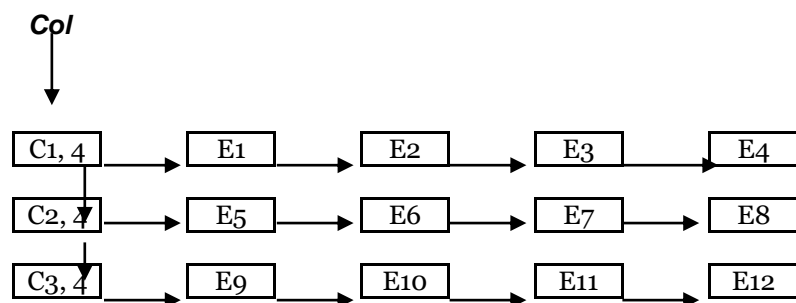


Fig. 1

En este ejemplo se utilizaron listas simplemente enlazadas, tanto para enlazar los cursos o asignaturas, como para enlazar a los estudiantes.

2.- Suponga que se quiere almacenar la información de varias brigadas de la construcción y de cada uno de ellos tiene asociado además su lista de sus trabajadores.

Esta situación se puede representar gráficamente de la siguiente forma:

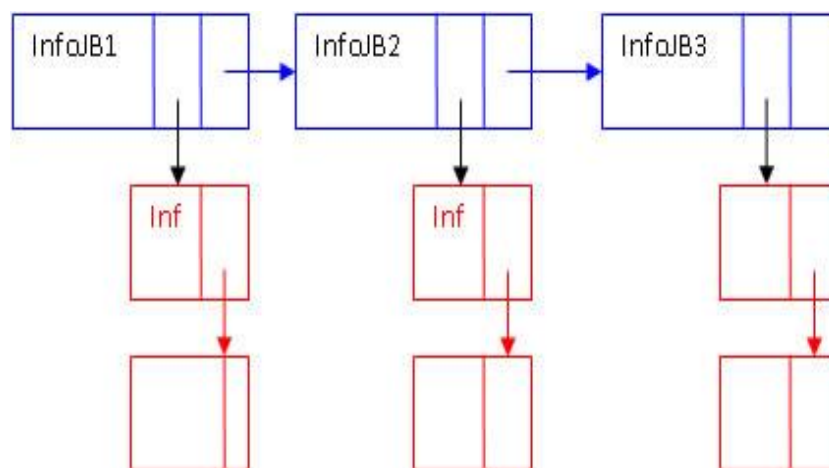


Fig.2

En la misma se han utilizado Listas Simplemente Enlazadas para conectar los nodos, tanto de las Brigadas como de los Trabajadores.

Estructura de Datos de la Multilista.

Para el caso de la Multilista de asignaturas y estudiantes, dado que se tienen dos nodos diferentes se puede definir la siguiente Estructura de Datos, tomando en cuenta que estos elementos tienen asociada la información que a continuación se expone:

De cada asignatura se conoce el código, nombre y la cantidad de horas, y de cada estudiante se conoce un identificador, su nombre y dirección. Quedando la estructura de datos de la siguiente forma:

Estructura de Datos utilizando C de asignaturas y estudiantes.

```
struct nodoestudiante{  
    string codigo, nombre, direccion;  
    struct nodoestudiante *siguiente;  
};  
  
struct nodoasig {  
    string codigo, descripcion;  
    int cantidad;  
    struct nodoasig *siguiente;  
    struct nodoestudiante *pestudiante;  
};  
  
nodoasig *primero,*ultimo;
```

Para el caso de la Multilista de Brigadas y trabajadores, dado que se tienen dos nodos diferentes se puede definir la siguiente estructura de datos, tomando en cuenta que estos elementos tienen asociada la información que a continuación se expone:

De la brigada se conoce el número, y la cantidad de estudiantes y de cada estudiante se conoce un identificador, su nombre y dirección. Quedando la estructura de datos de la siguiente forma:

Estructura de Datos en C de la Multilista de Brigadas y trabajadores.

```
struct nodotrabajador{  
    string identificador, nombre, oficio;  
    struct nodotrabajador *siguiente;  
};
```

```
struct nodobrigada {  
    string NumBrig, IdJB, NombreJB;  
    struct nodobrigada *siguiente;  
    struct nodotrabajador *ptrabajador;  
};
```

```
nodobrigada *primero,*ultimo;
```

Operaciones Básicas de una Multilista.

Para la multilista de asignaturas y estudiantes es necesario al menos las siguientes operaciones:

1. Insertar una asignatura.
2. Buscar una asignatura.
3. Insertar un estudiante en una asignatura.
4. Eliminar un estudiante.
5. Eliminar una asignatura.
6. Mostrar los estudiantes de una asignatura determinada.
7. Mostrar toda la multilista.

A continuación se implementan algunas de ellas, ya que algunas de ellas ya han sido estudiadas en el tema anterior:

Buscar una asignatura: A esta función se le pasa como parámetro la asignatura que se quiere buscar y devuelve un puntero a la asignatura, si la encontró. En caso de que devuelva NULL, significará que no existe esa asignatura.

```
nodoasig *Buscar_Asignatura(string valor)
{
    nodoasig *p,*q;
    q=primero;
    p=primero;
    while ((p!= NULL) && (p->codigo!=valor))
    {
        q=p;
        p = p->siguiente;
    }
    return p;
}
```

Insertar un estudiante en una asignatura: A esta operación se pasa como parámetro la asignatura, y los datos del estudiante a insertar. SE busca la asignatura, si existe, se adiciona el estudiante al inicio de la lista de estudiantes de la asignatura.

```
void insertar_estudiante ( string codasig, string codest, string nomb, string dir)
{
    nodoestudiante *nodo;
    nodoasig *p;
    nodo= new nodoestudiante;
    nodo->siguiente=NULL;
    nodo->codigo=codest;
    nodo->nombre=nomb;
    nodo->direccion=dir;
    p=Buscar_Asignatura(codasig);
    if (p==NULL)
```

```
        cout<<"No existe esa asignatura";

    else

        {

            nodo->siguiente=p->pestudiante;

            p->pestudiante=nodo;

        }

    return;

}
```

Eliminar un estudiante: En esta operación, se pasará como parámetro, la asignatura y el identificador del estudiante, se busca la asignatura, una vez que se ha encontrado, se busca al estudiante y se elimina si existe.

```
void eliminar_estudiante(string valor1, string valor2)

{

    nodoasig *p;

    nodoestudiante *q,*r;

    p=Buscar_Asignatura(valor1);

    if (p==NULL)

        cout<<"No existe esa asignatura";

    else

        {

            q=p->pestudiante;

            r=q;

            while ((q!= NULL) && (q->codigo!=valor2))

                {

                    r=q;

                    q = q->siguiente;
```

```
}  
if (q==NULL)  
    cout<<"Ese estudiante no cursa esa asignatura";  
else  
{  
    r->siguiente=q->siguiente;  
    if (q==p->pestudiante)  
        p->pestudiante=q->siguiente;  
    }  
}  
return;  
}
```

Mostrar los estudiantes de una asignatura determinada: En esta operación se pasa como parámetro la asignatura, la cual se busca, si existe, se recorre su lista de estudiantes y se van mostrando.

```
void mostrar_estudiantes (string valor)  
{  
    nodoasig *p;  
    nodoestudiante *q;  
    p=Buscar_Asignatura(valor);  
    if (p==NULL)  
        cout<<"No existe esa asignatura";  
    else  
    {  
        q= p->pestudiante    ;
```

```
        while (q != NULL){  
            cout<< q->codigo<<endl;  
            cout<< q->nombre<<endl;  
            cout<< q->direccion<<endl;  
            q = q->siguiente;  
            getch();  
        }  
    }  
    return;  
}
```

Mostrar toda la multilista: Esta operación se ocupa de mostrar todas las asignaturas y para cada una de ellas, todos sus estudiantes.

```
void mostrar_todo ()  
{  
    nodoasig *p;  
    nodoestudiante *q;  
    if (vacio())  
        cout<<"Lista vacia";  
    else  
    {  
        p = primero    ;  
        while (p != NULL){  
            cout<< p->codigo<<endl;  
            cout<< p->descripcion<<endl;  
            cout<< p->cantidad<<endl;  
            q=p->pestudiante;
```



```
        while (q != NULL){  
            cout<< q->codigo<<endl;  
            cout<< q->nombre<<endl;  
            cout<< q->direccion<<endl;  
            q=q->siguiente;  
            getchar();  
        }  
        p = p->siguiente;  
        getchar();  
    }  
}  
return;  
}
```

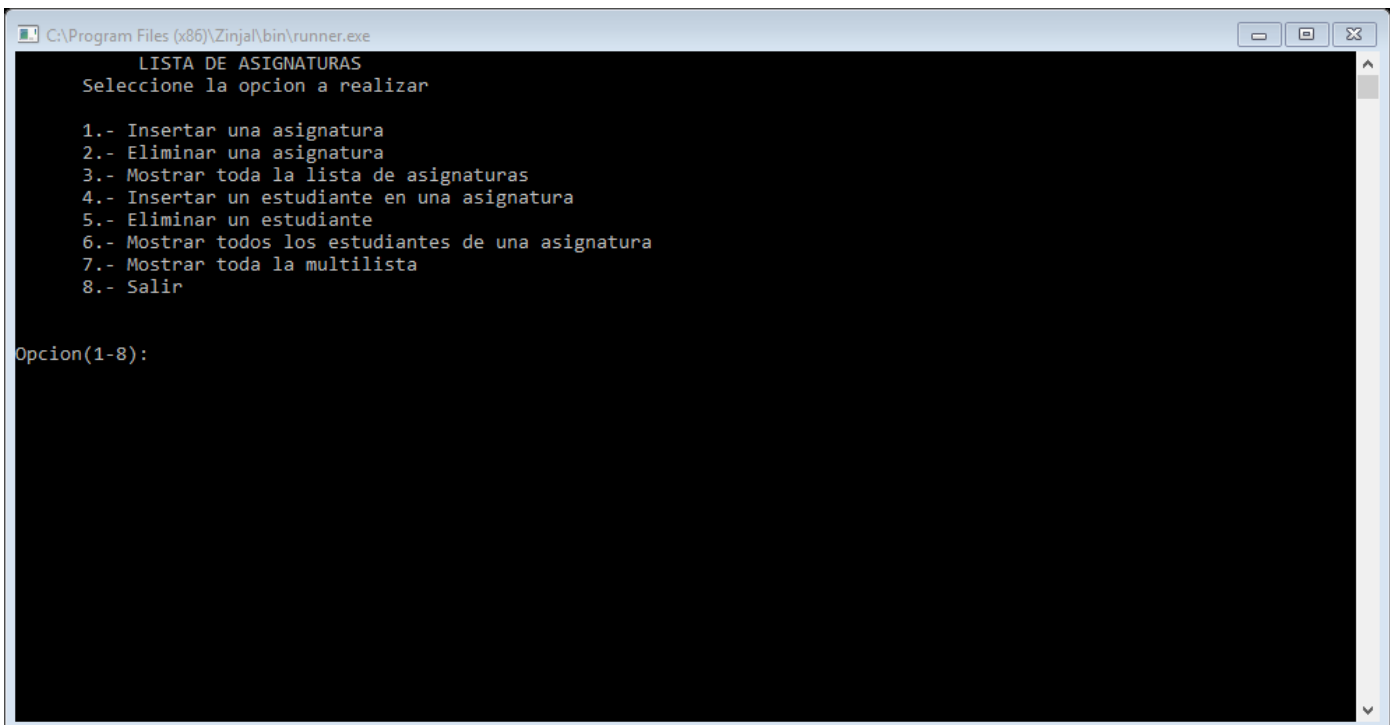
Para la Multilista de Brigadas y Trabajadores es necesario al menos las siguientes operaciones:

1. Insertar una brigada.
2. Insertar un trabajador en una brigada.
3. Eliminar una brigada.
4. Eliminar un trabajador.
5. Mostrar todos los trabajadores de una brigada

Implementación de la multilista de asignaturas y estudiantes

A continuación se muestra como quedaría el programa una vez implementado en consola, y la interfaz.

INTERFAZ:



The screenshot shows a Windows console window titled "C:\Program Files (x86)\Zinjal\bin\runner.exe". The console displays the following text:

```
LISTA DE ASIGNATURAS
Seleccione la opcion a realizar

1.- Insertar una asignatura
2.- Eliminar una asignatura
3.- Mostrar toda la lista de asignaturas
4.- Insertar un estudiante en una asignatura
5.- Eliminar un estudiante
6.- Mostrar todos los estudiantes de una asignatura
7.- Mostrar toda la multilista
8.- Salir

Opcion(1-8):
```

PROGRAMA:

```
#include <iostream>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <conio.h>

using namespace std;

struct nodoestudiante {
    string codigo, nombre, direccion;
    struct nodoestudiante *siguiente;
};
```

```
struct nodoasig {
    string codigo, descripcion;
    int cantidad;
    struct nodoasig *siguiente;
    struct nodoestudiante *pestudiante;
};

nodoasig *primero,*ultimo;

void iniciar()
{
    primero=NULL;
    ultimo=NULL;
}

int vacia()
{
    if (primero==NULL)
        return 1;
    else
        return 0;
}

void insertar_delante ( string cod, string nomb, int cant)
{
    nodoasig *nodo;

    nodo= new nodoasig;
    nodo->siguiente=NULL;
    nodo->pestudiante=NULL;
    nodo->codigo=cod;
    nodo->descripcion=nomb;
    nodo->cantidad=cant;

    if (vacía())
    {
        primero=nodo;
        ultimo=nodo;
    }
    else
    {
        nodo->siguiente=primero;
        primero=nodo;
    }
    return;
}
```

```
void mostrar ()
{
    nodoasig *p;
    if (vacía())
        cout<<"Lista vacía";
    else
    {
        p = primero ;
        while (p != NULL){
            cout<< p->codigo<<endl;
            cout<< p->descripcion<<endl;
            cout<< p->cantidad<<endl;
            p = p->siguiente;
            getch();
        }
    }
    return;
}
```

```
void eliminar(string valor)
{
    nodoasig *p,*q;

    q=primero;
    p=primero;
    while ((p!= NULL) && (p->codigo!=valor))
    {
        q=p;
        p = p->siguiente;
    }
    if (p==NULL)
        cout<<"No existe ese elemento en la lista";
    else
    {
        if (primero==ultimo)
        {
            primero=NULL;
            ultimo=NULL;
        }
        else
        {
            q->siguiente=p->siguiente;
            if (p==primero)
                primero=p->siguiente;
            else
                if (p==ultimo)
                    ultimo=q;
        }
    }
}
```

```
        }
    }

    return;
}

nodoasig *Buscar_Asignatura(string valor)
{
    nodoasig *p,*q;

    q=primero;
    p=primero;
    while ((p!= NULL) && (p->codigo!=valor))
    {
        q=p;
        p = p->siguiente;
    }
    return p;
}

void insertar_estudiante ( string codasig, string codest, string nomb, string dir)
{
    nodoestudiante *nodo;
    nodoasig *p;

    nodo= new nodoestudiante;
    nodo->siguiente=NULL;
    nodo->codigo=codest;
    nodo->nombre=nomb;
    nodo->direccion=dir;
    p=Buscar_Asignatura(codasig);
    if (p==NULL)
        cout<<"No existe esa asignatura";
    else
    {
        nodo->siguiente=p->pestudiante;
        p->pestudiante=nodo;
    }

    return;
}
```

```
void mostrar_estudiantes (string valor)
{
    nodoasig *p;
    nodoestudiante *q;

    p=Buscar_Asignatura(valor);

    if (p==NULL)
        cout<<"No existe esa asignatura";

        else
        {
            q= p->pestudiante      ;
            while (q != NULL){
                cout<< q->codigo<<endl;
                cout<< q->nombre<<endl;
                cout<< q->direccion<<endl;
                q = q->siguiente;
                getchar();
            }
        }

    return;
}

void mostrar_todo ()
{
    nodoasig *p;
    nodoestudiante *q;

    if (vacía())
        cout<<"Lista vacía";
    else
    {
        p = primero      ;
        while (p != NULL){
            cout<< p->codigo<<endl;
            cout<< p->descripcion<<endl;
            cout<< p->cantidad<<endl;
            q=p->pestudiante;
            while (q != NULL){
                cout<< q->codigo<<endl;
                cout<< q->nombre<<endl;
                cout<< q->direccion<<endl;
                q=q->siguiente;
                getchar();
            }
            p = p->siguiente;
            getchar();
        }
    }
}
```

```
        return;
    }

void eliminar_estudiante(string valor1, string valor2)
{
    nodoasig *p;
    nodoestudiante *q,*r;
    p=Buscar_Asignatura(valor1);
    if (p==NULL)
        cout<<"No existe esa asignatura";
    else
    {
        q=p->pestudiante;
        r=q;
        while ((q!= NULL) && (q->codigo!=valor2))
        {
            r=q;
            q = q->siguiente;
        }
        if (q==NULL)
            cout<<"Ese estudiante no cursa esa asignatura";
        else
        {
            r->siguiente=q->siguiente;
            if (q==p->pestudiante)
                p->pestudiante=q->siguiente;
        }
    }

    return;
}
```

```
int main(int argc, char *argv[])
{
    int cant;

    string cod, nombre, c,c1;
    string codest,nombreest,direst;
    int opc;

    iniciar();

    do
    {
        system("cls");
```

```
cout<<"    LISTA DE ASIGNATURAS"<<endl;
cout<<"    Seleccione la opcion a realizar\n\n";
cout<<"    1.- Insertar una asignatura\n";
cout<<"    2.- Eliminar una asignatura\n";
cout<<"    3.- Mostrar toda la lista de asignaturas\n";
cout<<"    4.- Insertar un estudiante en una asignatura\n";
cout<<"    5.- Eliminar un estudiante\n";
cout<<"    6.- Mostrar todos los estudiantes de una asignatura\n";
cout<<"    7.- Mostrar toda la multilista\n";
cout<<"    8.- Salir";
cout<<"    \n\n\nOpcion(1-8): ";

cin>>opc;

switch(opc)
{
case 1:
    cod="";
    nombre="";
    cant=0;
    cout<<"Entre los Datos de la Asignatura: \n ";
    cout<<"Codigo o siglas de la Asignatura: \n";
    cin>>cod;
    cout<<"Nombre de la Asignatura: \n";
    cin>>nombre;
    cout<<"Cantidad de Horas:\n ";
    cin>>cant;
    insertar_delante(cod,nombre,cant);
    break;
case 2:
    cout<<"codigo de la Asignatura a eliminar: \n";
    cin>>c;
    eliminar(c);
    getchar();
    break;
case 3:
    mostrar();
    cout<<"Oprima una tecla para salir";
    getchar();
    break;
case 4:
    cout<<"Entre los Datos de la Asignatura y del Estudiante:\n ";
    cout<<"Codigo o siglas de la Asignatura: \n";
    cin>>cod;
    cout<<"Identificador del estudiante: \n";
    cin>>codest;
    cout<<"Nombre del estudiante: \n";
    cin>>nombreest;
    cout<<"Direccion del Estudiante:\n ";
    cin>>direst;
    insertar_estudiante(cod, codest,nombreest,direst);
    cout<<"Oprima una tecla para salir";
    getchar();
    break;
```



```

        case 5:
            cout<<"codigo de la Asignatura que cursa el estudiante: \n";
            cin>>c;
            cout<<"codigo del estudiante: \n";
            cin>>c1;
            eliminar_estudiante(c, c1);
            getchar();
            break;
        case 6:
            cout<<"codigo de la Asignatura a Mostrar sus estudiantes: \n";
            cin>>c;
            mostrar_estudiantes(c);
            cout<<"Oprima una tecla para salir";
            getchar();
            break;
        case 7:
            mostrar_todo();
            cout<<"Oprima una tecla para salir";
            getchar();
            break;
        case 8:
            exit(0);
            break;
    }
}
while ((opc!=8));

return 0;
}

```

Otras Multilistas.

Se pueden enlazar las Multilistas de diferentes formas, a continuación se muestra algunas de las formas que se pueden enlazar tomando en cuenta la Multilista de Asignaturas y Estudiantes.

Multilista Doblemente Enlazadas.

Supongamos la representación gráfica de la figura Nro3, en ella se muestra la misma multilista estudiada, es decir, se tiene un conjunto de asignaturas y para cada una sus estudiantes, con la misma información. Pero los nodos tienen dos enlaces. Estando en presencia en este caso de listas doblemente enlazadas.

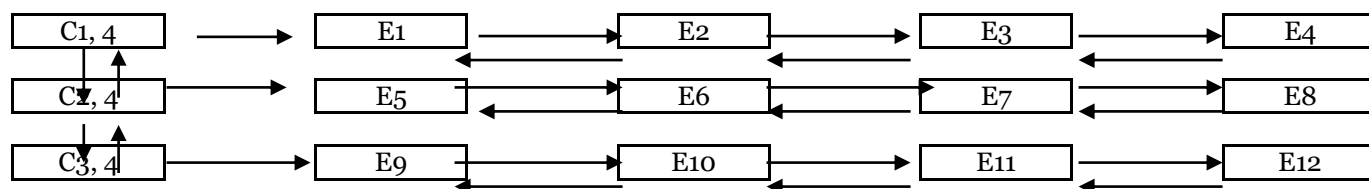


Fig. 3

Multilista Simplemente Enlazada Circular

Supongamos la representación gráfica de la figura Nro4, En ella se muestra la misma multilista estudiada, es decir, se tiene un conjunto de asignaturas y para cada una sus estudiantes, con la misma información. Pero los nodos tienen un enlace y son circulares. En este caso se está en presencia de listas simplemente enlazadas circulares

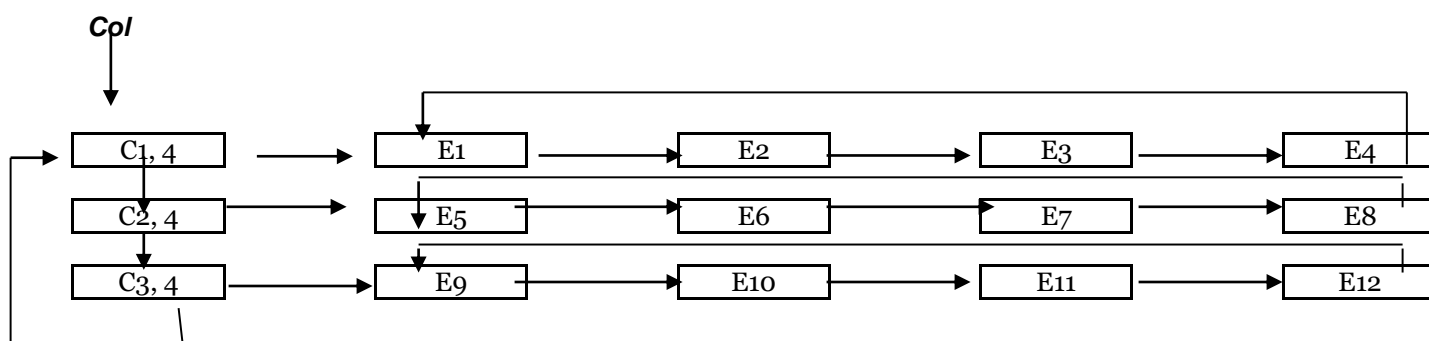


Fig. 4

Multilista simplemente Enlazadas con Nodo Cabeza.

Supongamos la representación gráfica de la figura Nro5, En ella se muestra la misma multilista estudiada, es decir, se tiene un conjunto de asignaturas y para cada una sus estudiantes, con la misma información. Pero tiene además un nodo cabeza. . Estando en presencia en este caso de listas simplemente enlazadas con nodo cabeza.

Cpl

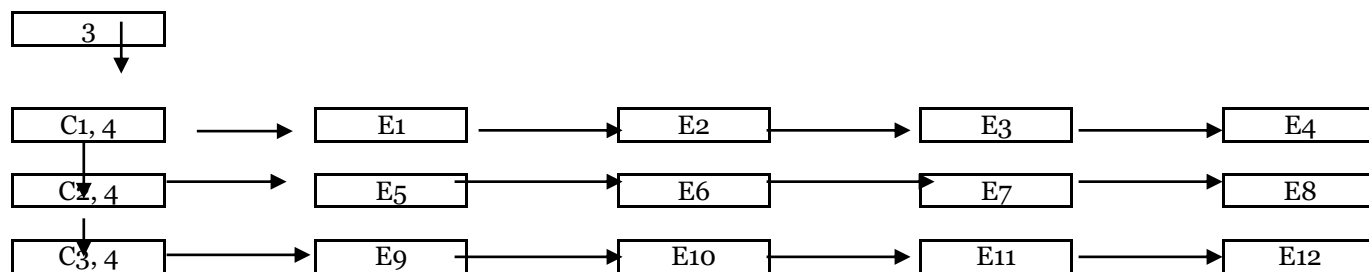


Fig. 5

Y de esta forma se pueden enumerar un sinnúmero de combinaciones de listas.

Laboratorio.

Implemente en el laboratorio la multilista de asignaturas y estudiantes, vista anteriormente, adicione las siguientes opciones:

- 1.- Cantidad de Estudiantes en una determinada asignatura.
- 2.- Adicione el campo o atributo sexo a los estudiantes.
- 3.- Muestre todos los estudiantes del sexo masculino.
- 4.- Muestre todos los estudiantes del sexo femenino.