Multilistas

Estructuras de Datos

Andrea Rueda

Pontificia Universidad Javeriana
Departamento de Ingeniería de Sistemas

Quiz: STL

Quiz: STL

Standard Template Library (STL)

`std::vector`

- Arreglo dinámico, crece en memoria según necesidad.

- Acceso aleatorio con operador [] (sobrecargado).

- Inserción y eliminación directa sólo en el final (cola) del arreglo.

www.cplusplus.com/reference/vector/vector

Quiz: STL

```
#include <vector>
#include <iostream>

int main() {
   std::vector<int> arr;
   std::cout << std::endl;
}
```

1. Insertar algunos datos...

2. Imprimir los datos insertados en pantalla...

Quiz: STL - vector

```
#include <vector>
#include <iostream>

int main() {
   std::vector<int> arr;
   for (int i=0; i<6; i++)
     arr.push_back(10-i);
   for (int ind=0; ind<arr.size(); ind++)
     std::cout << arr[ind] << " ";
   std::cout << std::endl;
}
```

Quiz: STL

Standard Template Library (STL)

`std::deque`

- Cola de dos cabezas, arreglo dinámico que crece por ambos extremos.

- Acceso aleatorio con operador [] (sobrecargado).

- Inserción y eliminación directa tanto en la cabeza como en la cola.

www.cplusplus.com/reference/deque/deque

Quiz: STL

```
#include <iostream>
#include <deque>

int main() {
   std::deque<int> dcola;
   std::cout << std::endl;
}
```

3. Insertar algunos datos en ambos extremos...

4. Imprimir los datos insertados en pantalla...

Quiz: STL - deque

```
#include <iostream>
#include <deque>

int main() {
  std::deque<int> dcola;

  for (int i=0; i<3; i++) {
    dcola.push_back(i);
    dcola.push_front(10-i);
  }

  for (int ind=0; ind<dcola.size(); ind++)

    std::cout << dcola[ind] << " ";

  std::cout << std::endl;
}
```

Quiz: STL

Standard Template Library (STL)

`std::list`

- Lista doblemente encadenada, conexión al siguiente y al anterior.

- Acceso directo a cabeza y cola, otros elementos a través de iteradores.

- Inserción y eliminación directa tanto en la cabeza como en la cola.

www.cplusplus.com/reference/list/list

Quiz: STL

```
#include <iostream>
#include <list>

int main() {
  std::list<int> lista;

  std::cout << std::endl;
}
```

5. Insertar algunos datos en ambos extremos...

6. Imprimir los datos insertados en pantalla...

Quiz: STL - list

```
#include <iostream>
#include <list>

int main() {
  std::list<int> lista;

  for (int i=0; i<3; i++) {
    lista.push_back(i);
    lista.push_front(10-i);
  }

  std::list<int>::iterator it;

  for (it=lista.begin(); it!=lista.end(); it++)

    std::cout << *it << " ";

  std::cout << std::endl;
}
```

Secuencias en STL

`vector`

`deque`

`list`

Impresión en pantalla - `vector`

```
vector<int> seq;
vector<int>::iterator it;


for (it=seq.begin(); it!=seq.end(); it++)
{
  std::cout << *it << std::endl;
}
```

Impresión en pantalla - deque

```
deque<int> seq;
deque<int>::iterator it;


for (it=seq.begin(); it!=seq.end(); it++)
{
  std::cout << *it << std::endl;
}
```

Impresión en pantalla - `list`

```
list<int> seq;
list<int>::iterator it;


for (it=seq.begin(); it!=seq.end(); it++)
{
  std::cout << *it << std::endl;
}
```

Impresión en reversa - `vector`

```
vector<int> seq;
vector<int>::reverse_iterator it;


for (it=seq.rbegin();it!=seq.rend(); it++)
{
  std::cout << *it << std::endl;
}
```

Impresión en reversa - deque

```
deque<int> seq;
deque<int>::reverse_iterator it;


for (it=seq.rbegin();it!=seq.rend(); it++)
{
  std::cout << *it << std::endl;
}
```

Impresión en reversa - `list`

```
list<int> seq;
list<int>::reverse_iterator it;


for (it=seq.rbegin();it!=seq.rend(); it++)
{
  std::cout << *it << std::endl;
}
```

Multilistas

Multilistas

- ¿Representación de una matriz utilizando memoria dinámica?

Multilistas

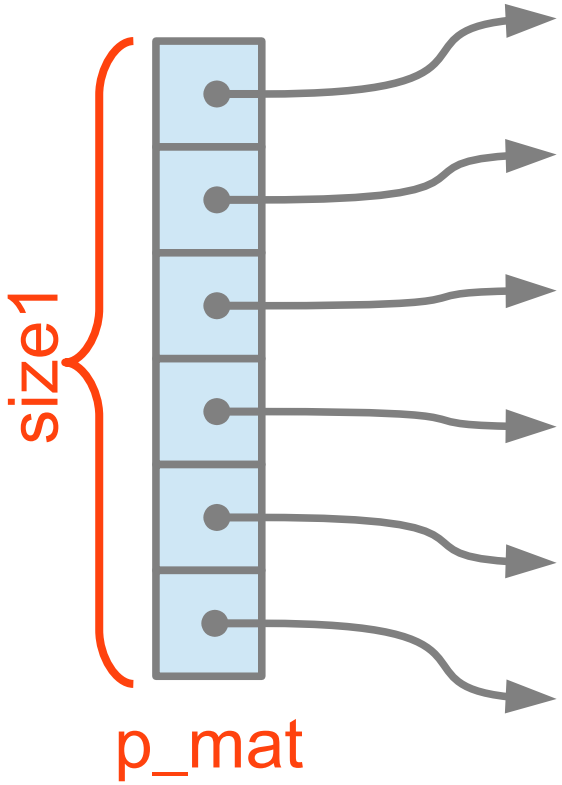
- ¿Representación de una matriz utilizando memoria dinámica?

```
int **p_mat;
p_mat = new int* [size1];

for (int i=0; i<size1; i++)
  *(p_mat+i) = new int [size2];
```

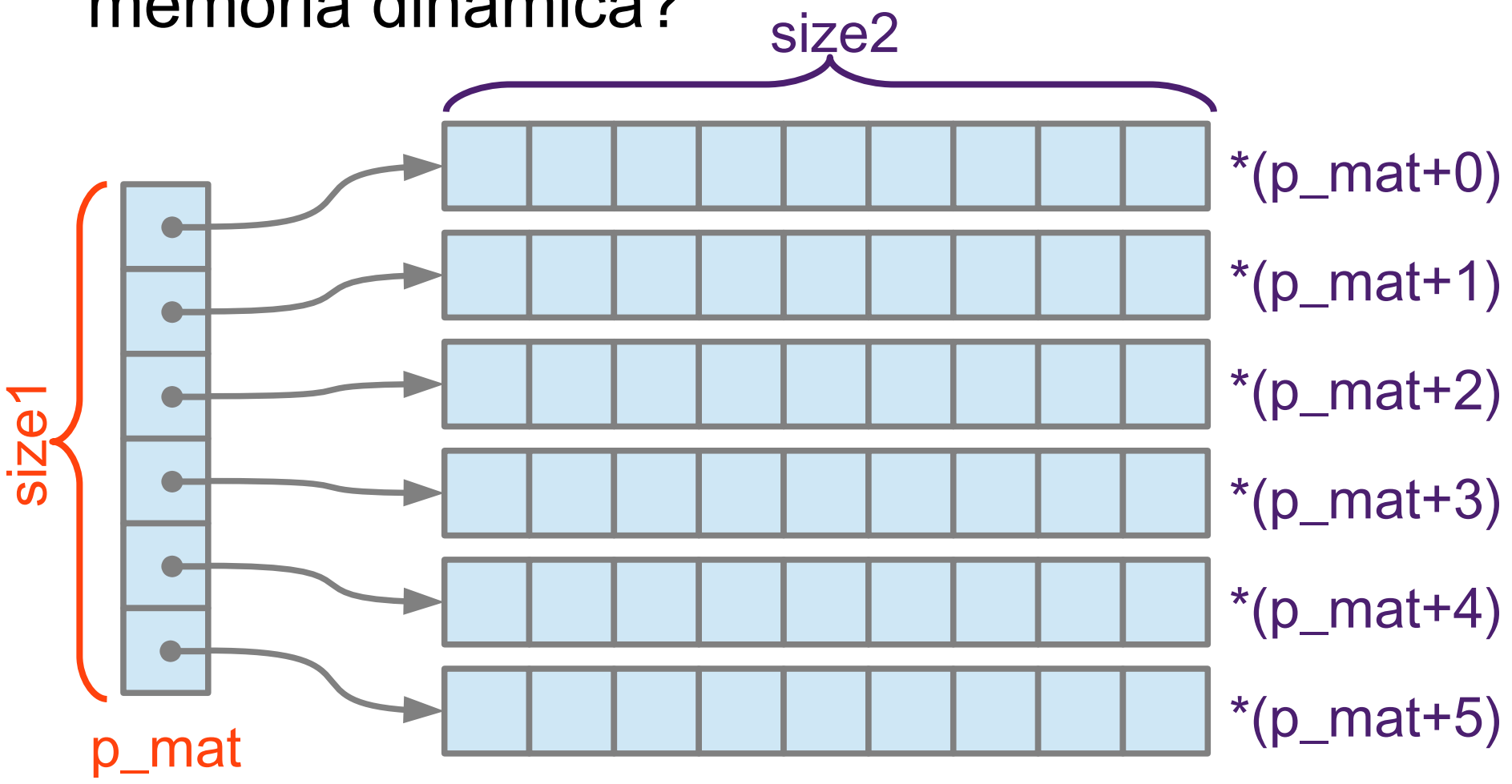
Multilistas

- ¿Representación de una matriz utilizando memoria dinámica?



Multilistas

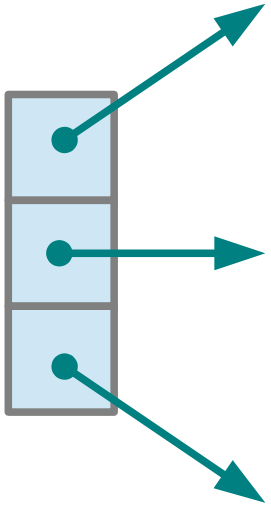
- ¿Representación de una matriz utilizando memoria dinámica?



Multilistas

- Si una lista (secuencia) es análoga a un arreglo...

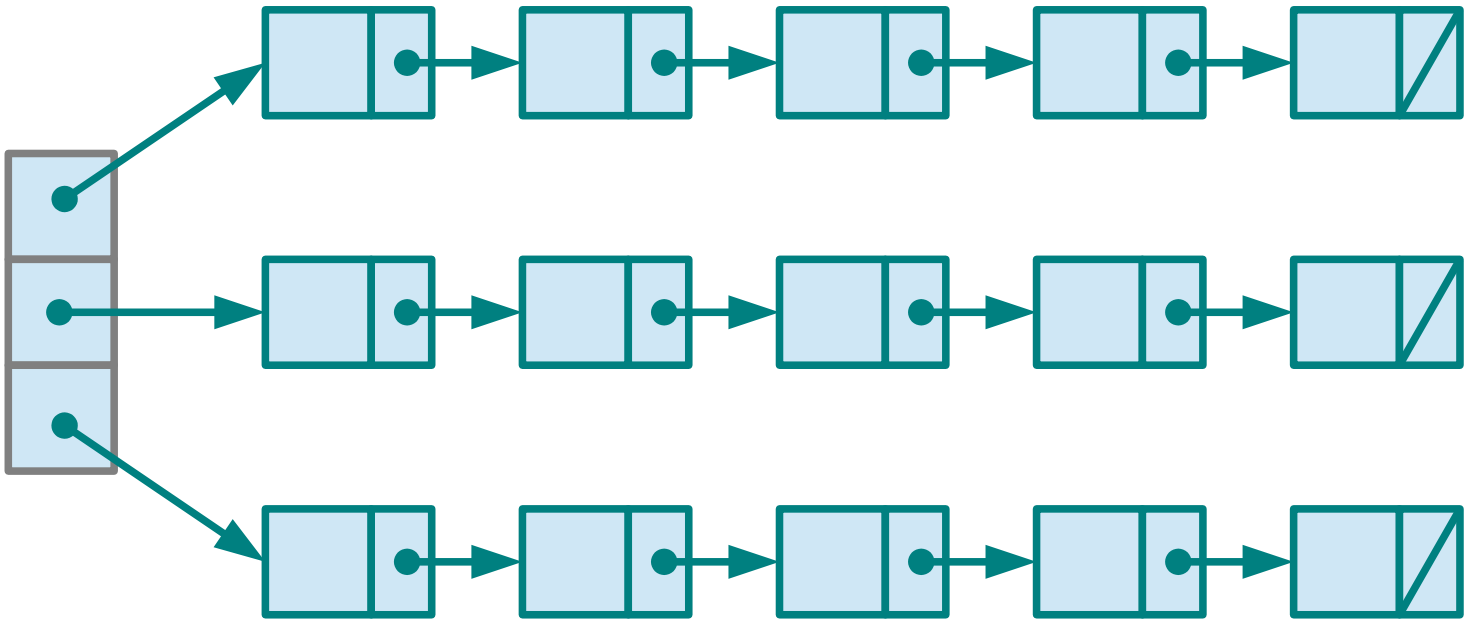
  ...¿podemos representar una matriz como un conjunto de listas (secuencias)?



Multilistas

- Si una lista (secuencia) es análoga a un arreglo...

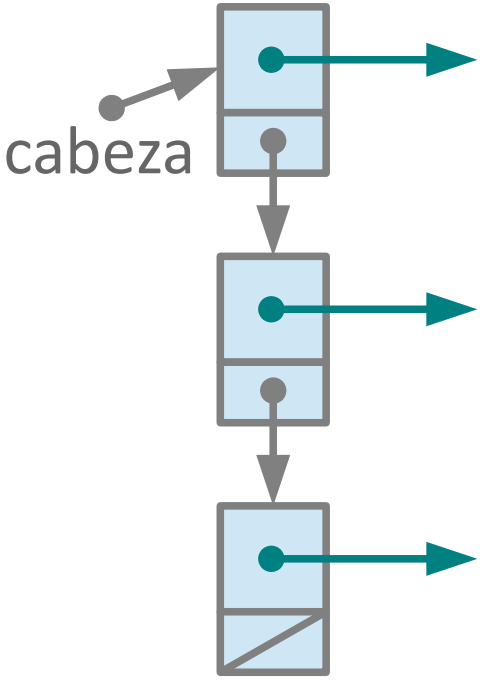
  ...¿podemos representar una matriz como un conjunto de listas (secuencias)?



Multilistas

- Si una lista (secuencia) es análoga a un arreglo...

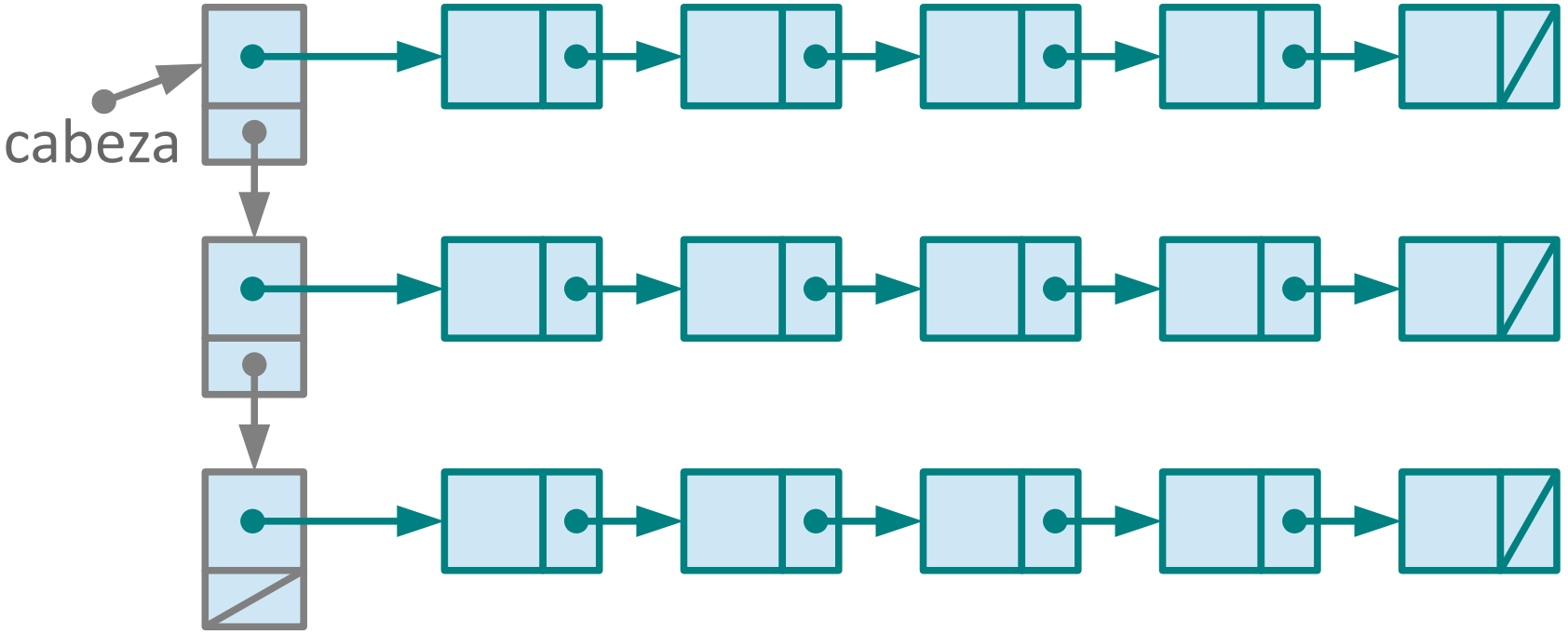
  ...¿podemos representar una matriz como un conjunto de listas (secuencias)?



Multilistas

- Si una lista (secuencia) es análoga a un arreglo...

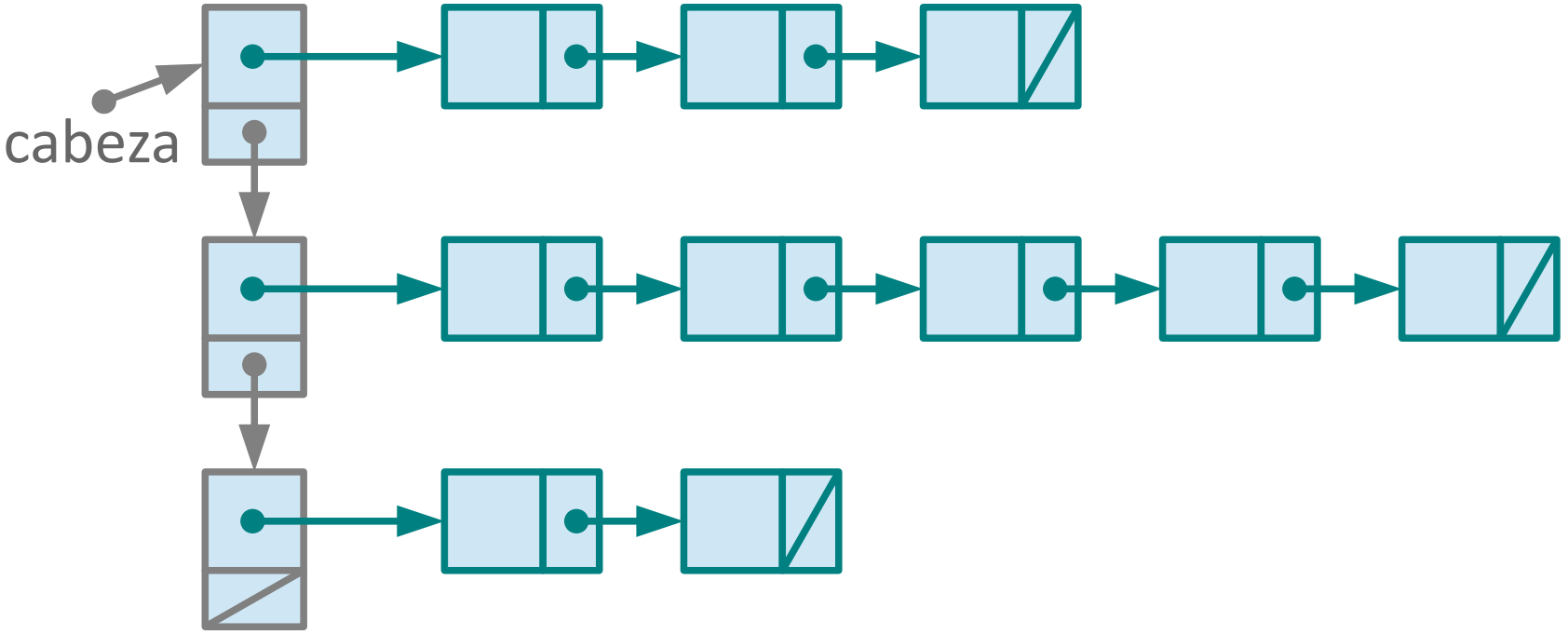
  ...¿podemos representar una matriz como un conjunto de listas (secuencias)?



Multilistas

- Esto es una multilista: una lista de listas (secuencia de secuencias).

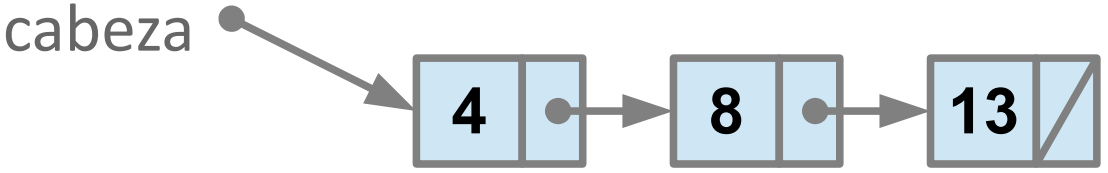
  Ventaja: al crearse nodo a nodo, no todas deben ser del mismo tamaño.



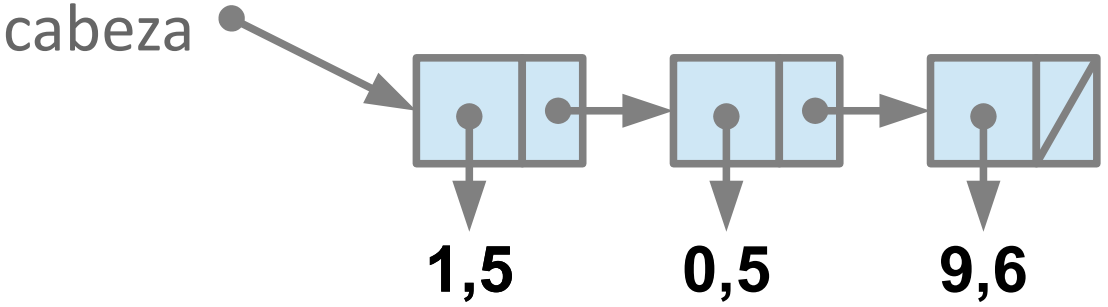
Multilistas

- Implementación:

  Lista de números enteros:
  `std::list<int> enteros;`



  Lista de apuntadores a números reales:
  `std::list<float*> reales;`



Multilistas

- Implementación:

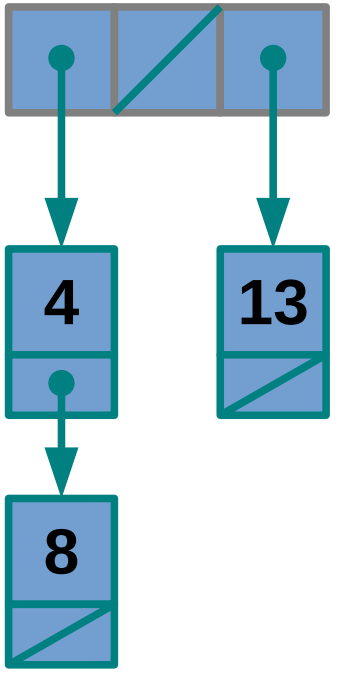
  ¿Arreglo de listas?

Multilistas

- Implementación:

  ¿Vector de listas?

```
std::vector< std::lista<int> > arr;
arr[0].push_back(4);
arr[2].push_back(13);
arr[0].push_back(8);
```



Multilistas

- Implementación:

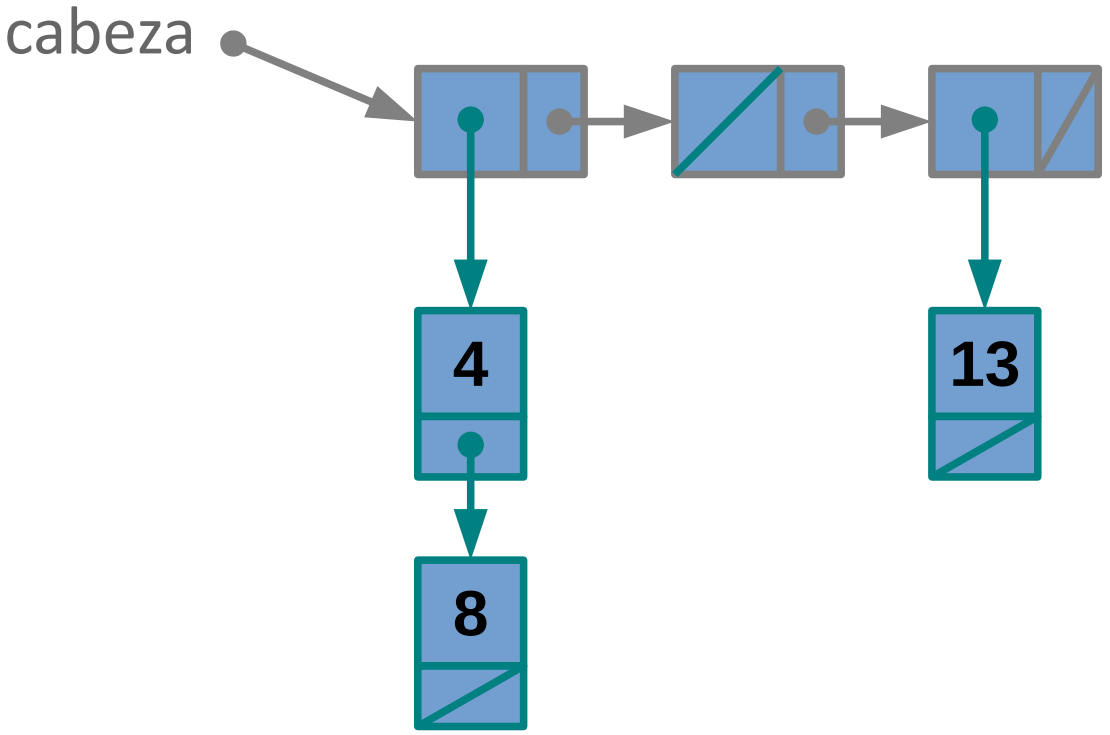
  ¿Lista de listas?

Multilistas

- Implementación:

  ¿Lista de listas?

```
std::list< std::list<int> > multilista;
```



Multilistas

- Implementación:

  ¿Lista de listas?

```
std::list< std::list<int> > multilista;
std::list< std::list<int> >::iterator it;
it = multilista.begin();
it->push_back(4);
it->push_back(8);
it++;
it++;
it->push_back(13);
```

Multilistas

- Ejercicio:

  Considere el TAD Vehiculo, implementado:

```
class Vehiculo {
  protected:
    unsigned short modelo;
    string placa;
    unsigned int tarifaDiaria;
  public:
    unsigned short obtenerModelo();
    string obtenerPlaca();
    unsigned int obtenerTarifa();
    ...
};
```

Multilistas

- Ejercicio:

  … y considere una entidad que administra diferentes parqueaderos, cada uno visto como una lista (secuencia) de vehículos.

Multilistas

- Ejercicio:

  1. ¿Cómo se representaría la entidad?

Multilistas

- Ejercicio:

  1. Representar la entidad como una lista de listas de vehículos.

```
std::list< std::list<Vehiculo> > entidad;
```

Multilistas

- Ejercicio:

  2. Asumiendo que los parqueaderos se identifican con un número entero consecutivo, ¿Cómo se insertaría un vehículo en un parqueadero dado?

Multilistas

- Ejercicio:

  2. Insertar un vehículo en un parqueadero dado.

```
void InsertarVeh( Vehiculo nVeh, int idParq )
{
  std::list< std::list<Vehiculo> >::iterator itP;
  itP = entidad.begin();
  for (int i = 1; i <= idParq; i++)
    itP++;
  itP->push_back(nVeh);
}
```

Multilistas

- Ejercicio:

  3. ¿Cómo se calcularía la cantidad total de vehículos en la entidad?

Multilistas

- Ejercicio:

  3. Cantidad total de vehículos en la entidad.

```
unsigned int totalVehs( )
{
  unsigned int total = 0;
  std::list< std::list<Vehiculo> >::iterator itP;
  itP = entidad.begin();
  for ( ; itP != entidad.end(); itP++)
    total += itP->size();
  return total;
}
```

Multilistas

- Ejercicio:

  4. ¿Cómo se calcularía el valor total diario que recibe la entidad por los vehículos parqueados?

Multilistas

- Ejercicio:

  4. Valor total diario de vehículos parqueados.

```
unsigned int totalDiario( )
{
  unsigned int total = 0;
  std::list< std::list<Vehiculo> >::iterator itP;
  std::list<Vehiculo>::iterator itV;
  itP = entidad.begin();
  for ( ; itP != entidad.end(); itP++) {
    itV = itP->begin();
    for ( ; itV != itP->end(); itV++)
      total += itV->obtenerTarifa();
  }
  return total;
}
```

Referencias

- T.H. Cormen, C.E. Leiserson, R.L. Rivest, C. Stein. Introduction to Algorithms, 3rd edition. MIT Press, 2009.
- L. Joyanes Aguilar, I. Zahonero. Algoritmos y estructuras de datos: una perspectiva en C. McGraw-Hill, 2004.