Algoritmos y Estructura de Datos I

Taller de listas

1 de julio de 2015

Menú del día

- Repaso de memoria dinámica

Menú del día

- Repaso de memoria dinámica
- Repaso de listas

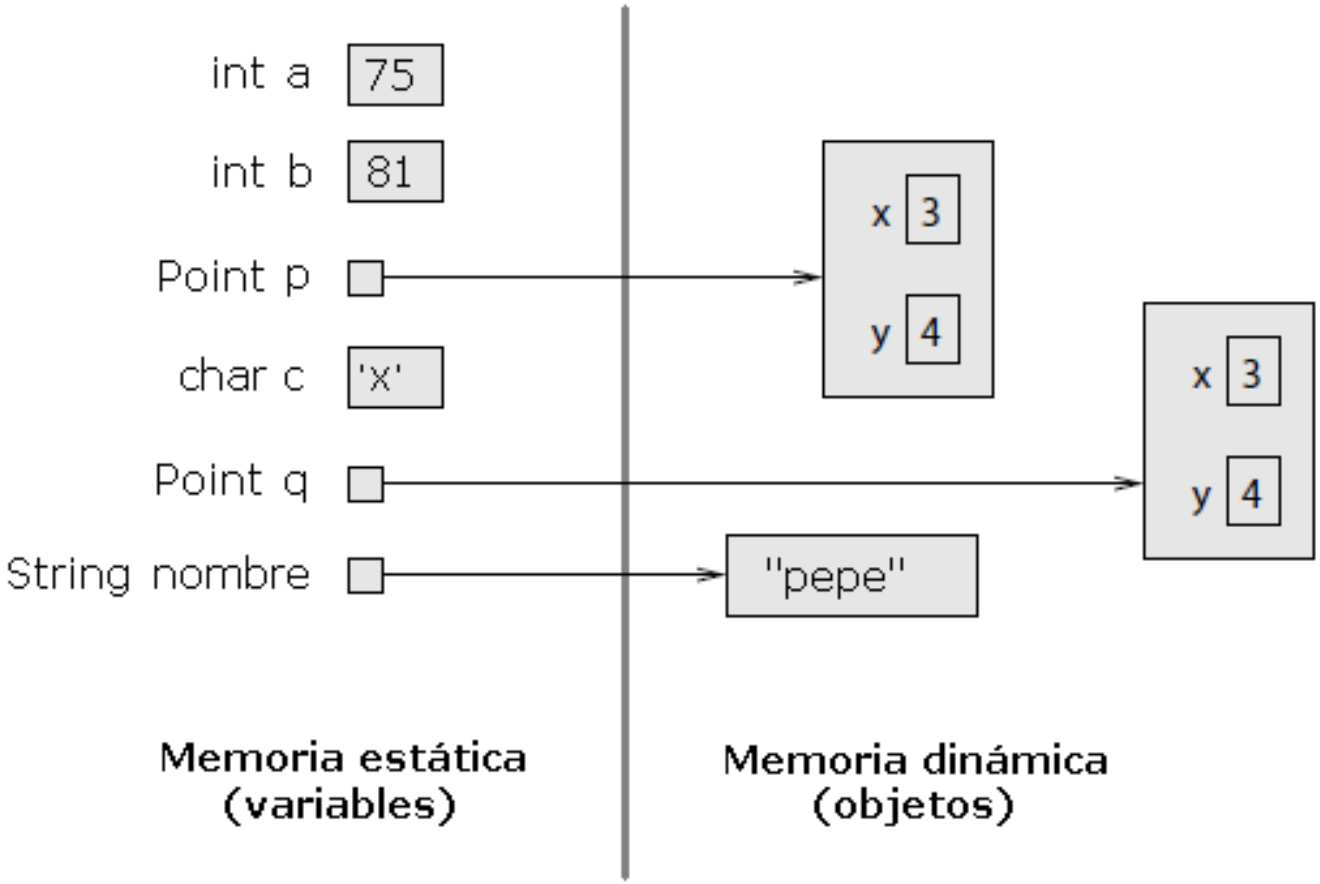
Menú del día

- Repaso de memoria dinámica
- Repaso de listas

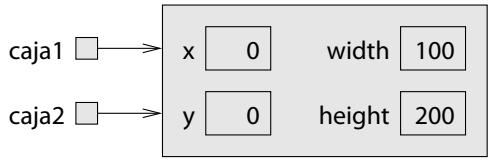
Menú del día

- Repaso de memoria dinámica
- Repaso de listas
- A programar

Repaso de memoria dinámica

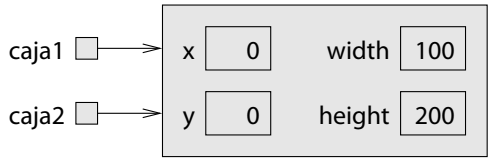


Aliasing

- ```
  Rectangle* caja1 = new Rectangle(0, 0, 100, 200);
  Rectangle* caja2 = caja1;
  ```
  ```

Aliasing

- ```
  Rectangle* caja1 = new Rectangle(0, 0, 100, 200);
  Rectangle* caja2 = caja1;
  ```


  ```

Aliasing

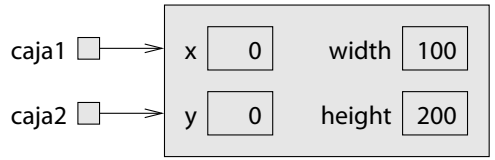
- ```
  Rectangle* caja1 = new Rectangle(0, 0, 100, 200);
  Rectangle* caja2 = caja1;
  ```



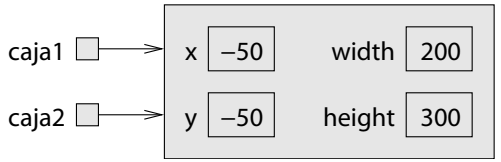

- ```
  cout << caja2->width << endl;
  caja1->grow(100,100);
  cout << *caja2 << endl;
  ```

Aliasing

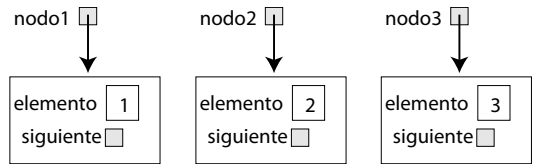
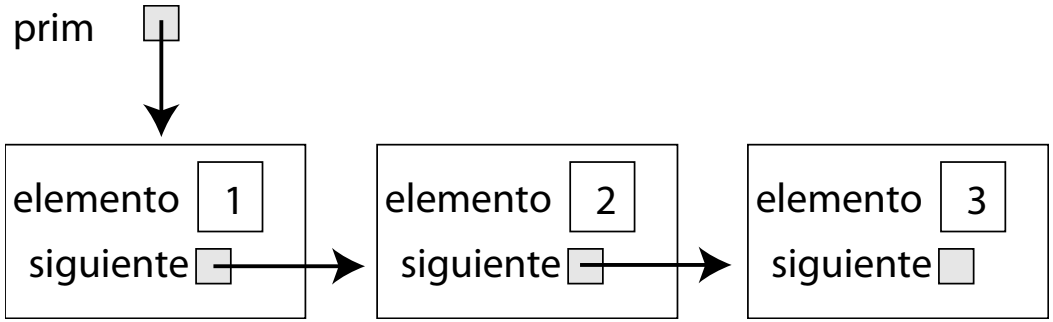
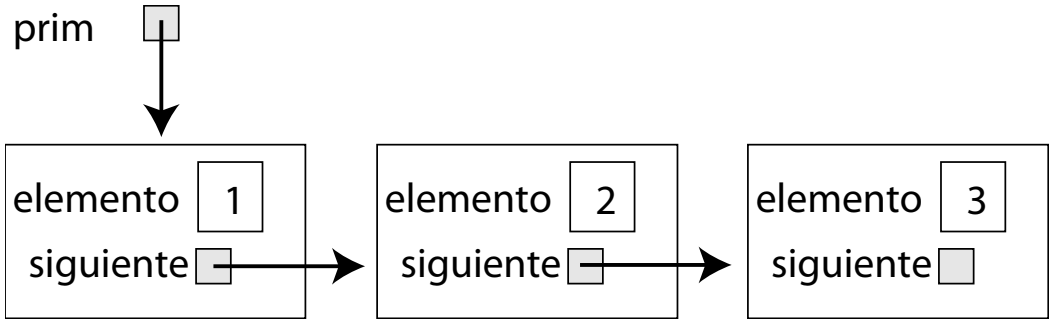
- ```
  Rectangle* caja1 = new Rectangle(0, 0, 100, 200);
  Rectangle* caja2 = caja1;
  ```




- ```
  cout << caja2->width << endl;
  caja1->grow(100,100);
  cout << *caja2 << endl;
  ```



Repaso de listas

- ```
  class Nodo
  {
    Nodo* siguiente;
    int elemento;
  }
  ```
  ```

Repaso de listas

- ```
  class Nodo
  {
    Nodo* siguiente;
    int elemento;
  }
  ```
- ```
  Nodo* nodo1 = new Nodo();
  nodo1->elemento=1;
  Nodo* nodo2 = new Nodo();
  nodo2->elemento=2;
  Nodo* nodo3 = new Nodo();
  nodo3->elemento=3;
  ```

Repaso de listas

- ```
  class Nodo
  {
    Nodo* siguiente;
    int elemento;
  }
  ```
- ```
  Nodo* nodo1 = new Nodo();
  nodo1->elemento=1;
  Nodo* nodo2 = new Nodo();
  nodo2->elemento=2;
  Nodo* nodo3 = new Nodo();
  nodo3->elemento=3;
  ```



Repaso de listas

- Notemos entonces que con sólo mantener una referencia al primer nodo de la lista, podemos acceder a todos los elementos:



Repaso de listas

- Notemos entonces que con sólo mantener una referencia al primer nodo de la lista, podemos acceder a todos los elementos:




- ```
  class Lista
  {
    Nodo* cabeza;
  }
  ```
  ```

A programar !!

- Programar la lista, agregar adelante un elemento y una función que las imprima

A programar !!

- Programar la lista, agregar adelante un elemento y una función que las imprima
- Programar agregar atras de la lista;

A programar !!

- Programar la lista, agregar adelante un elemento y una función que las imprima
- Programar agregar atras de la lista;
- Hacer un método copiar, que copie la lista, pero que al modificar el elemento en una de ellas no lo modifique en la otra.

A programar !!

- Programar la lista, agregar adelante un elemento y una función que las imprima
- Programar agregar atras de la lista;
- Hacer un método copiar, que copie la lista, pero que al modificar el elemento en una de ellas no lo modifique en la otra.
- Ordenar una lista