

Documentación "Lista Ligada"

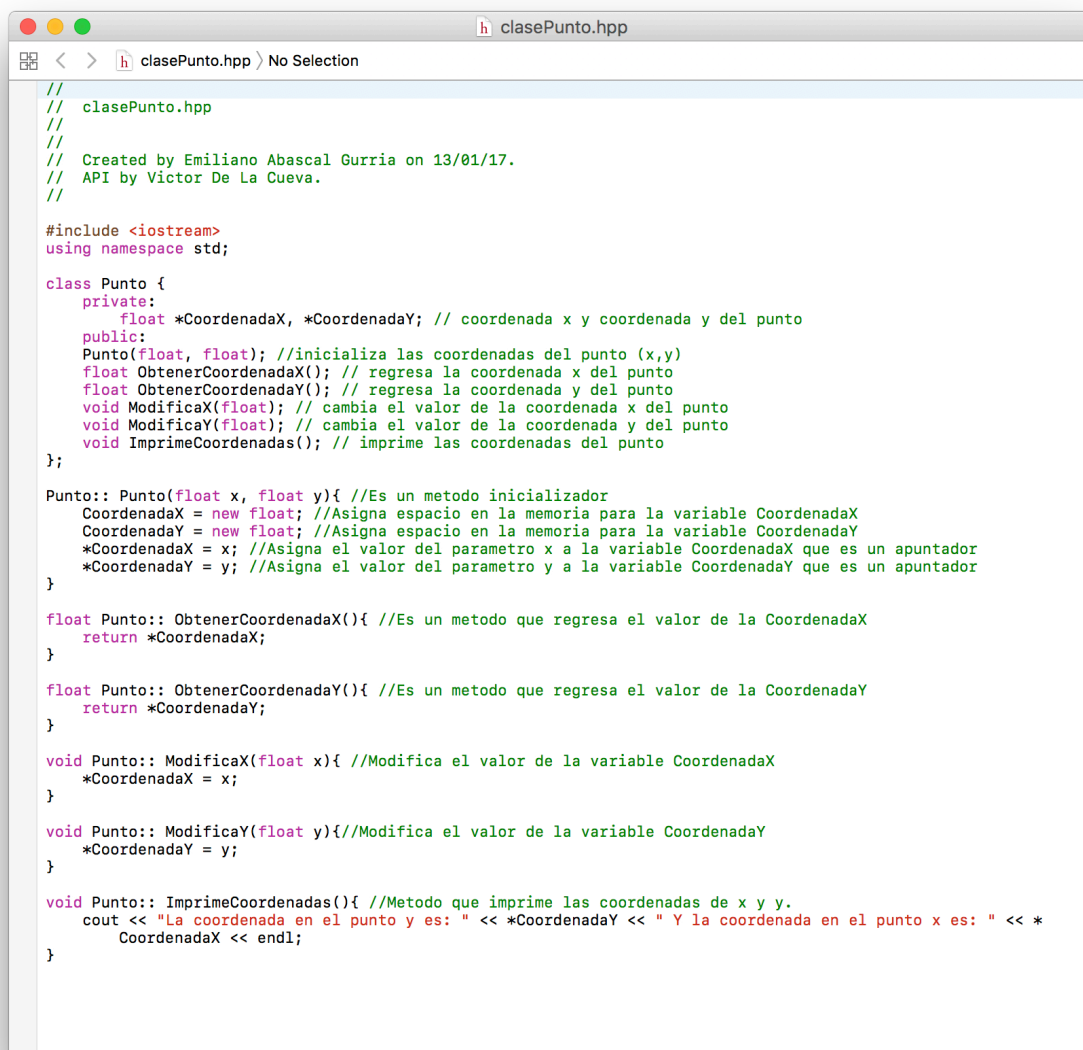
Emiliano Abascal Gurría

A01023234

Profesor: Victor Manuel De La Cueva

Instituto Tecnológico de Estudios Superiores de Monterrey

Campus Santa Fe



```
//
// clasePunto.hpp
//
// Created by Emiliano Abascal Gurría on 13/01/17.
// API by Victor De La Cueva.
//

#include <iostream>
using namespace std;

class Punto {
private:
    float *CoordenadaX, *CoordenadaY; // coordenada x y coordenada y del punto
public:
    Punto(float, float); //inicializa las coordenadas del punto (x,y)
    float ObtenerCoordenadaX(); // regresa la coordenada x del punto
    float ObtenerCoordenadaY(); // regresa la coordenada y del punto
    void ModificaX(float); // cambia el valor de la coordenada x del punto
    void ModificaY(float); // cambia el valor de la coordenada y del punto
    void ImprimeCoordenadas(); // imprime las coordenadas del punto
};

Punto::Punto(float x, float y){ //Es un metodo inicializador
    CoordenadaX = new float; //Asigna espacio en la memoria para la variable CoordenadaX
    CoordenadaY = new float; //Asigna espacio en la memoria para la variable CoordenadaY
    *CoordenadaX = x; //Asigna el valor del parametro x a la variable CoordenadaX que es un apuntador
    *CoordenadaY = y; //Asigna el valor del parametro y a la variable CoordenadaY que es un apuntador
}

float Punto::ObtenerCoordenadaX(){ //Es un metodo que regresa el valor de la CoordenadaX
    return *CoordenadaX;
}

float Punto::ObtenerCoordenadaY(){ //Es un metodo que regresa el valor de la CoordenadaY
    return *CoordenadaY;
}

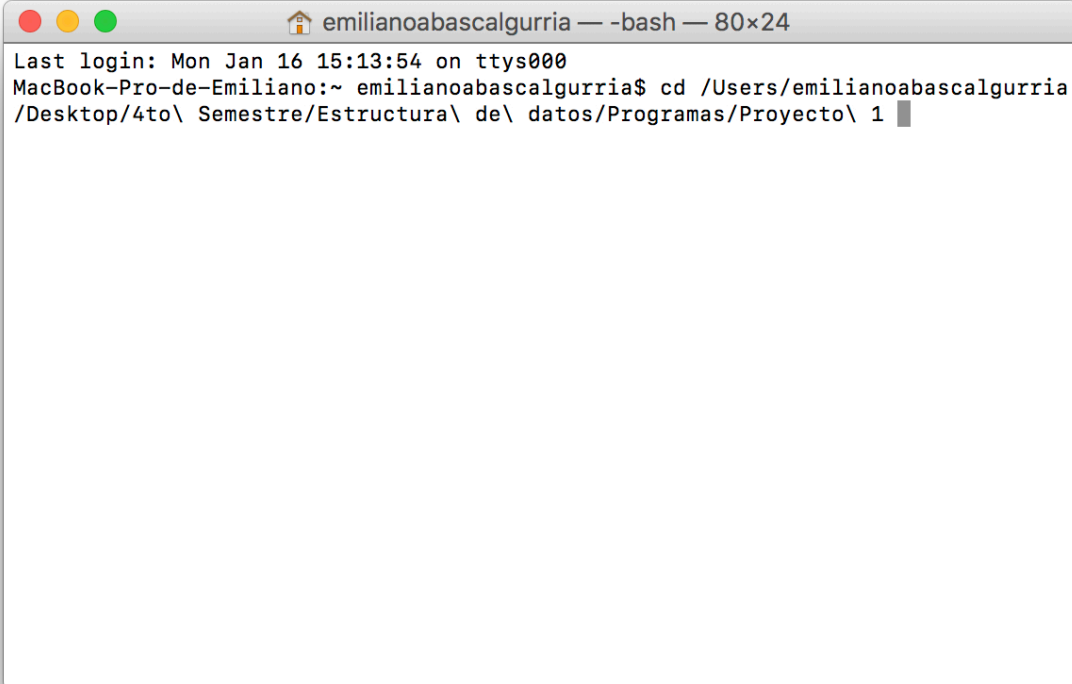
void Punto::ModificaX(float x){ //Modifica el valor de la variable CoordenadaX
    *CoordenadaX = x;
}

void Punto::ModificaY(float y){ //Modifica el valor de la variable CoordenadaY
    *CoordenadaY = y;
}

void Punto::ImprimeCoordenadas(){ //Metodo que imprime las coordenadas de x y y.
    cout << "La coordenada en el punto y es: " << *CoordenadaY << " Y la coordenada en el punto x es: " << *
        CoordenadaX << endl;
}
```

Manual de usuario:

- Una vez obtenido el software ubicarlo en un directorio de preferencia, por ejemplo en el escritorio.
- Abrir la terminal o un compilador para correr programas de C++.
- Escribir el siguiente comando: `cd "El directorio donde se encuentra el`



```
emilianoabascalgurria — -bash — 80x24
Last login: Mon Jan 16 15:13:54 on ttys000
MacBook-Pro-de-Emiliano:~ emilianoabascalgurria$ cd /Users/emilianoabascalgurria
/Desktop/4to\ Semestre/Estructura\ de\ datos/Programas/Proyecto\ 1
```

programa". como se encuentra en el siguiente ejemplo:

- Ya que se está en el directorio se requiere hacer un archivo `main.cpp` en el cual se importara el archivo de la siguiente manera: `#include "Lista.h"` y se llamarán a las funciones del programa.
- Al tener un archivo `main.cpp` regresa a la terminal y escribe el siguiente comando `g++ main.cpp` y da enter.
- Si no hay ningún problema, se creara un archivo `".out"` en el directorio donde se encuentran los archivos `"main.cpp"` y `Lista.h`.
- Escribe en la terminal `./a.out` y el programa se correrá, como se demuestra en al ejemplo a continuación.

06/04/2017

Proyecto2 — pi@raspberrypi: ~/Desktop — -bash — 80x24

```
iMacdeEmiliano2:Proyecto2 emilianoabascalgurria$ g++ main.cpp
```

Proyecto2 — pi@raspberrypi: ~/Desktop — -bash — 80x24

```
iMacdeEmiliano2:Proyecto2 emilianoabascalgurria$ ./a.out
```

Proyecto2 — pi@raspberrypi: ~/Desktop — -bash — 95x23

```
iMacdeEmiliano2:Proyecto2 emilianoabascalgurria$ ./a.out
Anidacion incorrecta para: '(()))'.
limpiaString para: 'aaaaabbbbccccdddeeeeggggggffffffiiiiiii' es: 'abcdegfi'.
La suma de digitos del numero: 5 es: 15.
El substring 'hola' esta 2 veces en el string 'hola Alejandro, hola Emiliano, soy Victor.'.
iMacdeEmiliano2:Proyecto2 emilianoabascalgurria$
```

Estructura General

El programa contiene métodos que modelan una lista ligada, tales como: `insertaInicio`, `insertaFinal`, `insertaDespues`, `eliminaPrimero`, `eliminaUltimo`, `eliminaNodo` e `imprimeLista`.

Algoritmos

- Función void `insertaInicio(int dato);`

```
Nodo x = new nodo
x -> cache = dato
x -> next = head
head = x
```

- Función void `insertaFinal(int dato);`

```
Nodo first = new nodo
Nodo last = new nodo

first->cache = dato
first->next = NULL
Si (head != NULL)
    last = head
    while (last -> next)
        last = last -> next

    last -> next = first
Sino
    head = first
```

- Función bool `insertaDespues(int dato, int ref);`

```
Nodo x = new nodo
Nodo y = new nodo
Si (head != NULL)
    y = head
    Mientras ((y != NULL )&& (y->cache != ref))
        y = y->next
    if (y == NULL)
        return false
    Sino entonces si (y != NULL)
        x -> cache = dato
        x -> next = y -> next
        y -> next = x
        return true

Sino
    return false
```

- Función bool eliminaPrimero(int &dato);

```

Nodo x = NULL
Si (head == NULL)
    return false
Sino entonces si(head != NULL)
    x = head
    head = x -> next
    dato = x -> cache
    delete x
    return true

```

- Función bool eliminaUltimo(int &dato);

```

Nodo x = new nodo
Nodo y = new nodo
Si (head != NULL)
    Si(!head -> next)
        delete head
        head = NULL
    Sino
        x = head
        Mientras(x -> next)
            y = x
            x = x -> next
        y -> next = NULL
        dato = x -> cache
        delete x
        return true

Sino
    return false;

```

- Función bool eliminaNodo(int ref, int &dato);

```
Nodo x = new nodo
```

```
Nodo y = new nodo
```

```
Si (head != NULL)
```

```
    x = head
```

```
    Mientras ((x->next) && (x->cache != ref))
```

```
        y = x
```

```
        x = x->next
```

```
    Si (x->cache != ref)
```

```
        return false
```

```
    Sino
```

```
    Si (head == x)
```

```
        head = head->next
```

```
    Sino
```

```
        y->next = x->next
```

```
    dato = x->cache
```

```
    delete x
```

```
    return true
```

```
else
```

```
    return false;
```

- Función void imprimeLista();

```
Nodo x = new nodo
```

```
x = head
```

```
if (x == NULL)
```

```
    cout << "No hay elementos en la lista."
```

```
else
```

```
Print "Lista":
```

```
    Mientras(x != NULL){
```

```
        Print x -> cache
```

```
        x = x->next
```

Descripción Técnica

- Variables de la clase:

- ```
typedef struct nodo{
 int cache;
 nodo *next;
};
```

 Que es una estructura que contiene como atributos, cache de tipo entero y \*next que es un apuntador de tipo nodo.

- \*Nodo: que es un apuntador

- Nodo head: Que es una variable de tipo Nodo.

- Función void insertaInicio(int dato);

- Argumentos: dato que es un entero.

- Variables: x que es de tipo Nodo.

- Funcionalidad: Se declara el nodo x de tipo nodo, el dato se almacena en el nodo x, el siguiente elemento de la lista será el primer elemento. El primer elemento será x.

- Funcion void insertaFinal(int dato);

- Argumentos: dato que es un entero.

- Variables: first y last que son de tipo Nodo.

- Funcionalidad: El nodo first que apunta al cache es igual al dato ingresado, el nodo first que apunta a next, es nulo. Si la lista tiene uno o mas nodos entonces el nodo last va a ser igual a head, mientras last apunte a valor nulo (next) entonces last sera last apuntando al valor siguiente. El nodo last apuntando a next sera igual al valor del nodo first. Si no, entonces el primer elemento sera el nodo first.

- Funcion bool insertaDespues(int dato, int ref);

- Argumentos: dato y ref, que son enteros.

- Variables: "x" y "y" que son de tipo Nodo.

- Funcionalidad: Si la lista tiene uno o mas nodos entonces el nodo y va a ser igual a head, mientras el valor de la referencia no se encuentre, se recorrerá la lista. Si el nodo y es nulo entonces se regresa falso, sino,

entonces si el nodo no es nulo, x apuntando a cache será el dato, x apuntando a next será y apuntando a next, y apuntando a next será x, y se regresa true. Si lo anterior no se cumple entonces se regresa falso.

- Funcion bool eliminaPrimero(int &dato);

- Argumentos: &dato que es una referencia a la variable dato.
- Variables: "x" de tipo Nodo.
- Funcionalidad: Si la lista no tiene ningun nodo entonces se regresa falso. Si tiene un elemento o mas, entonces x sera el primer nodo, el primer elemento sera x apuntando a next, dato sera igual a x apuntando a cache almacenando lo que se eliminara. Se borra el espacio de x y se devuelve true.

- Funcion bool eliminaUltimo(int &dato);

- Argumentos: &dato que es una referencia.
- Variables: "x" y "y" que son de tipo Nodo.
- Funcionalidad: Si la lista tiene uno o mas nodos entonces, si en la lista solo hay un elemento, entonces se borra el espacio del primero y se marca como nulo y eliminamos el elemento. Sino, entonces el nodo x almacena a head. Mientras 'x' apunte a 'next' y almacenara la dirección de p y x almacenara el apuntador a next, el nodo y que apunta a next es nulo, el dato es igual a x apuntando a cache, almacenando su información, se elimina x, y se regresa true. Si la lista está vacía entonces se regresa falso.

- Función bool eliminaNodo(int ref, int &dato)

- Argumentos: &dato que es una referencia y ref que es un entero.
- Variables: "x" y "y" que son de tipo Nodo.
- Funcionalidad: Si la lista tiene uno o mas nodos entonces el nodo x será el valor del primer nodo. Mientras el valor de la referencia no se encuentre, se cambia el valor del nodo al siguiente, hasta que se encuentre. Si x apuntando a cache es diferente a ref entonces se regresa falso porque no se encontró el nodo. Si se encontró el nodo entonces, si el primero tiene la dirección del nodo x, entonces head, guarda el apuntador a next, sino, se anexa el valor del nodo. Se almacena la información de lo que se borrara, Se elimina el nodo x y se devuelve true. Si la lista no contiene ningún elemento, se regresa falso.

- Función void imprimeLista();



- Argumentos: N/D
- Variables: "x" de tipo Nodo.
- Funcionalidad: Si no hay nodos, entonces se da mensaje de error, si hay nodos, entonces se imprimen los valores de la lista mientras el valor del nodo x, sea diferente a NULL.