Cú pháp khai báo kiểu dữ liệu Node và liên quan:

typedef <ElementType> ElementType;

ElementType có thể là 1 kiểu cơ bản (int, char,..) hoặc 1 kiểu struct tự định nghĩa như SinhVien, GiaoVien,..

Ví dụ:

struct SinhVien{  
 char MSSV[10];  
 char HoTen[50];  
 float DLT, DiemTH1, DiemTH2;  
};

typedef int ElementType;

Ở đây chọn ElemenType là kiểu int

struct Node {

ElementType Element;

struct Node \*Next;

};

typedef struct Node \*Position;

Giống với: typedef struct Node \*Position;

typedef Position List;

Giống với: typedef struct Node \*List;

Hay: typedef struct \*Node List;

Như vậy: Position và List đều là kiểu con trỏ struct Node

Các ví dụ về khai báo biến:

1. List L: L là con trỏ cấp 1 thuộc kiểu struct Node
2. List \*pL: là con trỏ cấp 2 thuộc kiểu struct Node;
3. Position P: là con trỏ cấp 1 thuộc kiểu struct Node
4. ElementType X: là biến thuộc kiểu ElementType

Các hàm định nghĩa trên Else:

Các biến:

1. L (List): Danh sách liên kết L (con trỏ cấp 1 kiểu struct Node)
2. X (ElementType): Phần tử X thuộc kiểu ElementType
3. P (Position): Vị trí P trong danh sách liên kết (Thực chất P cũng là con trỏ cấp 1 kiểu struct Node)
4. pL (\*List): Con trỏ…trỏ đến danh sách liên kết L (con trỏ cấp 2 kiểu struct Node)

void makenullList(&L): Khởi tạo danh sách liên kết rỗng L (L->Next = Null)

int member(x, L): Kiểm tra sự có mặt của phần tử X trong danh sách liên kết L

return 1: có; return 0: không có

void append(x, &L): Nối phần tử x vào danh sách liên kết L (thêm cuối)

void addFirst(x, &L): Thêm phần tử x vào đầu danh sách liên kết L (thêm đầu)

void deleteList(p, &L): Xóa phần tử ở vị trí p trong danh sách liên kết L

Position locate(x, L): Tìm vị trí xuất hiện đầu tiên của phần tử x trong danh sách liên kết L

Return P: Nếu tồn tại P->Element = x; return NULL: Nếu không tồn tại P thỏa mãn

**---------- Một số lưu ý khi gõ code**

